The Construction and Evaluation of Statistical Models of Melody and Harmony

Raymond Peter Whorley

Doctor of Philosophy
Department of Computing
Goldsmiths, University of London
November 2013

Declaration of Originality

I Raymond Peter Whorley of Goldsmiths, University of London, being a candidate for Doctor of Philosophy, hereby declare that this thesis and the research presented in it are my own work.

Signed

Date

Acknowledgements

I am grateful to my supervisors Christophe Rhodes, Geraint Wiggins and Marcus Pearce for their unstinting support throughout the ups and downs of my research. I would also like to thank colleagues at Goldsmiths and Queen Mary, University of London, for engendering a welcoming and academically stimulating atmosphere. A thorough reading by my examiners, François Pachet and Nicolas Gold, has resulted in significant improvements to this thesis. Finally, I acknowledge the financial support of Goldsmiths, in the form of a departmental studentship (Department of Computing) and conference expenses.

Abstract

This research is concerned with the development of representational and modelling techniques employed in the creation of statistical models of melody and four-part harmony. Previous work has demonstrated the utility of *multiple viewpoint systems*, along with techniques such as *Prediction by Partial Match*, in the construction of cognitive models of melodic perception. *Primitive* viewpoints represent surface and underlying musical attributes, while *linked* viewpoints model combinations of such attributes. A viewpoint selection algorithm optimises multiple viewpoint systems by minimising the information theoretic measure *cross-entropy.* Many more linked viewpoints are used in this research than have previously been available, and the results show that many new viewpoints are incorporated into optimised systems. A significant aspect of this work is the proposal and implementation of a set of novel extensions of the multiple viewpoint framework for four-part harmony. Statistical models are constructed with the aim that given a soprano part, alto, tenor and bass parts are added in a stylistically suitable way. Version 1 is as closely related to the modelling of melody as possible (chord replacing note), and is a baseline for gauging expected improvements as the framework is extended and generalised. Three versions of the framework have been implemented, and their performances compared and contrasted. The results indicate that the baseline version has been improved upon. Time complexity issues are discussed in detail, and selected viewpoints are examined from a music theoretic point of view for insights into why they perform well. Finally, melodies and harmonisations of given melodies are generated using the best performing models. The quality of the music suggests that, in spite of the improvements achieved so far, the models are still unable to fully capture the musical style of a corpus. Another six versions of the framework are described, which are expected to contribute further improvements.

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

1.1 Preamble

There have been many attempts by researchers to model musical composition, or aspects of it, using AI techniques. The more successful attempts at modelling harmony have generally incorporated explicit rules which encapsulate musical knowledge to one degree or another. In expert systems (*e.g.*, Ebcioğlu, 1988), the use of such rules is the dominant paradigm; and in genetic algorithms (*e.g.*, Wiggins et al., 1999) musical knowledge is part and parcel of the fitness function. There are problems associated with this sort of approach, however. Firstly, for any given style of harmonisation, there are many general rules and lots of exceptions to those rules, some of which might not yet have been enumerated. Formulating a theory of a style by creating a model in this way is therefore extremely time consuming, and the resulting theory is very likely to be incomplete. Secondly, in order to formulate theories of several different harmonic styles (*e.g.*, those of Tallis, Bach and Mozart), it is necessary to come up with different sets of general rules and exceptions (which multiplies the time expended).

Machine learning has the potential to circumvent the above problems. The idea is to write a program which allows the computer to learn for itself how to harmonise in a particular style, by creating a statistical model of harmony from a corpus of existing music in that style. Providing that the representational, machine learning and modelling techniques are good enough, the resulting model is then a theory of that style, containing structure equivalent to rules and exceptions. Present techniques, however, are not sufficiently well developed to produce convincingly good models; although Allan and Williams (2005) have certainly demonstrated their potential.

A means of representing music which, when combined with machine learning and modelling techniques, shows particular promise, is called *multiple viewpoint systems* (Conklin, 1990). This framework not only represents basic musical attributes like note duration and pitch, but also derived attributes such as intervals. The use of these derived attributes necessarily introduces a certain amount of low level musical knowledge, which researchers (*e.g.*, Dixon and Cambouropoulos, 2000) have found to be beneficial.

It is conceivable that these or similar attributes are used by the human brain in inducing models from auditory input which aid the cognition of music. Pearce (2005) has successfully used this framework to produce cognitive models of melodic expectancy, and these models have been developed to carry out phrase segmentation (Potter et al., 2007). Further evidence that this approach has validity from a cognitive point of view can be found by analogy with the work of Shanahan (2005). In his research on cognitive robotics, Shanahan (2005) uses the global workspace model of information flow within a brain-like architecture to create a series of "conscious" states from a multitude of "unconscious" parallel processes. Multiple viewpoint systems combine the predictions of multiple statistical models (analogous to the "unconscious" parallel processes) to produce an overall "conscious" prediction. The multiple viewpoint framework is considered to be an ideal representation scheme for this research.

1.2 Motivations and Aims

In this section some background information is presented, outlining motivations for four distinct computational activities relating to music. It is then possible to place the research, and its motivations and aims, into some sort of context.

1.2.1 Background

A review of early attempts to develop computer programs for the composition of music leads Pearce et al. (2002, p. 119) to identify four distinct activities: "algorithmic composition, the design of compositional tools, the computational modelling of musical styles and the computational modelling of music cognition." Each of these activities has its own particular motivations and applicable methodologies for program development and evaluation; these motivations and methodologies will be summarised later. First, however, a taxonomy of *artificial intelligence* or *AI*, also from Pearce et al. (2002), will be presented, since the computational modelling of music cognition belongs firmly in one of its branches. It is useful to distinguish between three different types of AI (Bundy, 1990). *Basic AI* is an engineering science which deals with computational techniques relevant to the simulation of intelligence; *cognitive science* or *computational psychology* is a natural science which models the intelligence of living beings; and *applied AI* is an engineering discipline which designs and builds products incorporating AI techniques.

The motivations for algorithmic composition, which essentially extends a composer's own techniques, are purely artistic in nature. In this case, methodological issues are of no concern for program development or evaluation, although they may apply to certain aesthetic considerations, as for example in live coding.

The design of compositional tools which are of benefit to composers in general, is its own motivation. The methodologies of software engineering are appropriate to the development and evaluation of such tools. Pressman (2000) considers that there are three major phases to software engineering:

1. The *definition* phase, which can be further broken down into three tasks. The first, *system* or *information engineering*, is concerned with the data to be processed and the configuration of the elements of the system on which the software will be run. The second is *software project planning*; and the third, *requirements analysis*, is concerned with what the software is supposed to do, what interfaces are required and the establishment of criteria for the evaluation of the software. The functionality of the software is defined in detail as an important part of this task.

2. The *development* phase, which again is split into three tasks. The first, *software design*, is concerned with the selection of data structures and the specification of a software architecture. The latter describes, for example, how proposed software components interact with each other. The second, *code generation*, is the implementation of the software design in a programming language; and the third is *software testing.* A variety of tests are carried out: *internal tests* check that low-level software components are working properly; *unit tests* ensure overall conformity with the requirements; *application tests* reveal how well the software performs for each of the identified use cases; and *stress tests* reveal how the software copes with extreme scenarios.

3. The *support* phase deals with bug fixes, adaptation to new hardware or changing requirements, the provision of enhanced functionality, and so on.

The motivation for the computational modelling of musical styles "is to propose and evaluate hypotheses concerning the important stylistic properties of a corpus of musical compositions" (Pearce et al., 2002, p. 130). This activity spans two of the five research areas in computational musicology described by Volk et al. (2011), namely *music theory and analysis* and *historical musicology* (specifically, in the latter case, the development of musical style). The computational approach allows the use of methodologies which are more scientific than the *speculative* methodology normally employed in musicology. This type of computer modelling "requires that all assumptions included in the theory (self-evident or otherwise) are explicitly and formally stated" (Pearce et al., 2002, p. 130); also, "the implemented model may be evaluated through comparison of the compositions it generates with the human-composed pieces which the theory is intended to describe" (Pearce et al., 2002, pp. 130–1). Meredith (1996) notes that hypotheses can be disproved by the generation of stylistically aberrant compositions or the failure to generate stylistically typical compositions, and proposes practical methods for carrying out this type of evaluation. There is a problem with this approach, however, which is that hypotheses may be corroborated (though not proved) or disproved (*i.e.*, they are either provisionally true or definitely false). In practice, a completely comprehensive

Domain	**Activity**	**Motivation**
Composition	Algorithmic composition	Expansion of compositional repertoire
Software engineering	Design of compositional tools	Development of tools for composers
Computational musicology	Computational modelling of musical styles	Proposal and evaluation of theories of musical styles
Cognitive science	Computational modelling of music cognition	Proposal and evaluation of cognitive theories of musical composition

Table 1.1: Motivations for developing computer programs which compose music (adapted from Pearce et al., 2002).

hypothesis might be unattainable, in which case it would be appropriate to determine which of a number of false hypotheses best captures the style. Such comparisons are not a feature of Meredith's framework; but in §2.4 of the current document, this issue is addressed for statistical models in terms of hypotheses being relatively more or less probable rather than provisionally true or definitely false.

The motivation for the computational modelling of music cognition is to "help us understand the underlying cognitive processes involved in human composition" (Pearce et al., 2002, p. 134). There are three methodological issues to be considered in this case. Firstly, cognitive hypotheses must be clearly stated. Following Marr (1982) and McClamrock (1991), Pearce et al. (2002, p. 125) state that one must "specify the kinds of question (computational, algorithmic/representational or implementational) that the research will address;" and "specify a level of organisational abstraction which is the prime focus of the research." It is suggested that in the first instance questions are asked at the computational level. Secondly, experimental data relevant to music cognition must be obtained in order to formulate, support or refute hypotheses. Sloboda (1985) lists four sources of such data relating to composition: composers' manuscripts; composers' thoughts about their work methods; observation of composers at work; and observation of musicians improvising (the latter two sources are particularly amenable to objective analysis). Thirdly, "the evaluation of the theory [is] based on a detailed empirical analysis of the similarities and differences between its behaviour and the human behaviour it is intended to explain" (Pearce et al., 2002, p. 137). One method of evaluating a cognitive model of composition is to give the same compositional task to the computer model and to one or more human composers; a comparison of compositional behaviour throughout the task would then be made. A different approach has been proposed by Pearce and Wiggins (2001), which is described in detail in §2.4.

Finally, a very brief summary of the four identified activities, the domains to which they belong and their motivations, is given in Table 1.1.

1.2.2 Motivations

The primary motivation of this research is the proposal and evaluation of theories of musical styles. The basic idea is to develop representations, modelling techniques and machine learning techniques, applicable to melody and four-part harmony, to the extent that statistical models induced from different musical corpora are capable of producing melodies and harmonisations which are stylistically characteristic of each individual corpus. The computational approach employed allows the use of scientific methodologies.

A secondary motivation is to develop representations and techniques which can be adapted to induce statistical models of melody and harmony which may provide insights into the cognitive processes involved in the harmonisation of melodies, and in melodic and harmonic expectancy. It is expected that many of the developments applicable to models capable of generating melody and harmony will also be applicable to the computational modelling of the cognition of melody and harmony.

1.2.3 Aims

Following on from the success of Pearce (2005) in using multiple viewpoint systems to model melodic expectancy in humans, multiple viewpoint systems are the modelling technique of choice for this research. Viewpoints represent basic musical attributes such as pitch or derived ones such as interval, either individually or in combination. In the latter case they are called *linked* viewpoints. So far, this technique has mostly (although by no means exclusively) been used to model monodic sequences, and the pool of available viewpoints has been quite limited. Harmony, however, is more complex; for example, four-part harmony consists of four interrelated sequences. The use of more complex models and a much larger pool of viewpoints may well, therefore, be beneficial. Consequently, the main aims of the research are as follows:

1. to investigate the effect of a large pool of viewpoints on melodic models;
2. to propose ways in which the multiple viewpoint framework may be developed, such that the complexities of harmony can be adequately addressed;
3. to analyse the time complexity of software implementing these developments;
4. to design and carry out experiments to determine the best performing of these developments;
5. and to analyse the best performing viewpoints from a music-theoretic point of view, searching for regularities which confirm, conflict with and possibly transcend the commonly agreed rules of harmony and melodic construction, as well as seeking inspiration for new or improved viewpoints.

1.3 Original Contributions

This research directly contributes to the disciplines of basic artificial intelligence and computational musicology, while making indirect contributions to cognitive science and applied artificial intelligence (see §1.2.1).

1.3.1 Basic AI

Basic AI is concerned with computational techniques relevant to the simulation of intelligence. From this point of view, we can regard the machine learning of musical style from a corpus (which exhibits apparently intelligent behaviour) as a means of testing more generally applicable computational techniques.

Pearce (2005) introduced a viewpoint (feature) selection algorithm based on forward stepwise selection which evaluated all available viewpoints at each iteration. In this research there is a very much larger pool of available viewpoints (see §1.3.2 below), necessitating modification of the selection algorithm to make it more time efficient (see §3.4.5). A procedure similar to that described by Pickens and Iliopoulos (2005) for Markov random field induction has been developed: see Algorithms 3.1 to 3.7 in §3.4.5.4. The modified algorithm is unlikely to find the globally best multiple viewpoint system; but this was also the case for the original algorithm. This contribution allows viewpoint selection involving large numbers of viewpoints (not only musical ones) to be carried out within a reasonable timescale.

A great deal of guidance has been given in Chapter 4 about the construction of derived and linked viewpoint domains[1], based on the principle that between them, the members must be able to predict all of, and only, the members of the basic domain. This guidance culminates in the formal presentation of Algorithms 4.1 to 4.4 (see §§4.5.2 and 4.5.3), for use in the construction of domains for the most complex viewpoints used in this research. An unreliable domain construction procedure can result in overall prediction probabilities being incorrect, hence the importance of this contribution.

Three versions of the multiple viewpoint framework for harmony have been expounded and implemented as software. In principle, these developments are also applicable to any set of interrelated sequences, such as time-stamped economic or financial data. Version 1 is not a contribution as such, because it makes use of vertical viewpoint elements in exactly the same way as Conklin (2002); it is a baseline model for purposes of comparison. An empirical analysis of the time complexity of this version (as implemented) is carried out, however (see §4.6), which is a contribution. Although version 2 draws its inspiration from previous work (Allan, 2002; Hild et al., 1992; Phon-Amnuaisuk and Wiggins, 1999), the use of subtasks is new to the multiple viewpoint framework. Specifically, this version is able to predict or generate harmony one or more parts at a time, in any order. Version 3 is novel, in that it allows the use of different viewpoints

[1] A viewpoint domain is the set of valid elements (or symbols, or values) for a viewpoint.

in different voice-parts. A further six versions of the multiple viewpoint framework for harmony have been developed, but not yet implemented (see Chapter 10). The analysis inherent in the proposal of these novel additional versions is a contribution.

In §6.2.3 and §6.3.3 it is shown that the selection of specialist multiple viewpoint systems to individually predict musical attributes results in better overall harmonic models than the selection of a single system to predict all of the attributes. A similar, but very small, effect is noted with respect to melodic modelling in §5.3.3. It is highly likely that this result is applicable beyond the modelling of music.

Finally, although version 1 performs least well in conjunction with the corpus used during viewpoint selection (see §1.3.2), it appears that its more general models are better able to scale up to larger corpora (which may deviate somewhat from the characteristics of the original corpus) than those of versions 2 and 3. This can be an advantage, since viewpoint selection with a large corpus is extremely time-consuming. It is considered highly likely that this result too is applicable beyond the realms of music modelling.

1.3.2 Computational Musicology and Cognitive Science

The interdisciplinary field of computational musicology is one in which computational techniques are employed in pursuit of answers to musicological questions (Volk et al., 2011), while cognitive science seeks to model the intelligence of living beings. Inasmuch as the research described here makes progress in the statistical modelling of musical style, it is also potentially useful for the cognitive modelling of music (see, *e.g.*, Pearce 2005); therefore these parts of the research contribute directly to computational musicology and indirectly to cognitive science.

Many new atomic (or *primitive*) viewpoint types are introduced in §3.2.4.2. In addition, whereas there were only a limited number of ways in which such viewpoints could be linked in, for example, Pearce (2005), in this research any primitive viewpoint may be linked with any other, provided that such links are able to predict at least one attribute. We see in Chapter 7 that many of the better performing linked viewpoints with respect to the modelling of melody are new to this research, thereby demonstrating their contribution. Some of them are new linked viewpoints comprising existing primitive viewpoints, while others contain new primitive viewpoints. Chapter 8 provides evidence that new viewpoints are also amongst the better performing in relation to harmonic modelling, further demonstrating their contribution.

Taking the Cartesian product of the pitches seen in the soprano, alto, tenor and bass parts in the corpus produces 157,320 chords. The use of a domain of this size would result in exceedingly slow run times, and in any case a vast number of such pitch combinations would never be seen in music. Utilising only elements seen in the corpus and test data works well for melody (Pearce, 2005) but is severely limiting for harmony, where a means of taking account of chords as yet unseen would be of benefit. The novel solution described in §4.3.1 is based on the notion that a chord seen in one key (on a degree of scale basis) should be applicable to any key. Seen chords are transposed up and down a semitone at a time until one of the parts goes out of its range. Chords produced in this way which are not currently in the *augmented* domain are added to it. This contribution allows certain derived viewpoints to make use of chords not seen in the corpus (on an absolute pitch basis) when harmonising melodies.

In previous research, the *long-term model* (LTM, derived from the corpus) and *short-term model* (STM, derived from a single piece of music) have been combined (using a weighting technique) by amalgamating the viewpoint probability distributions within each of these models first, and then combining the two resulting distributions. Pearce (2005) proposed two possible alternative methods, but did not empirically compare them. The first effects a pair-wise combination of the distributions of identical viewpoints in the LTM and STM first, and then combines the resulting distributions. The second combines all viewpoint distributions at once, irrespective of whether they are in the LTM or STM. A comparison in §5.2.4 demonstrates that the original scheme is best, thereby making a contribution.

It is demonstrated in §6.3.1.1 that a better performance is achieved by predicting the bass part first followed by the inner parts together than by starting with the prediction of alto or tenor. This reflects the usual human approach to harmonisation. It is interesting to note that this heuristic, almost universally followed during harmonisation, therefore has an information theoretic explanation for its success. With certain provisos, a comparison in §6.4.3 shows that version 2 (with bass predicted first) performs better than version 1. This is further vindication of the bass first approach to harmonisation.

In §6.5.1 and §6.6.1 it is shown that version 3 performs better than versions 1 and 2. Furthermore, a worthwhile performance enhancement is attainable by creating a hybrid overall model comprising the better of version 2 and 3 subtask models. In Chapter 9, further quantitative evidence with respect to *probability thresholds* supports the conclusion that the baseline version 1 has been improved upon. Finally, the proposed viewpoint-based method for carrying out information theoretic music analysis outlined in Chapter 11 is a contribution.

1.3.3 Applied AI

Applied AI is concerned with the designing and building of products incorporating AI techniques. It is a field relevant to this research, but not specifically addressed by it in this thesis. At the very least, the direct contributions to basic AI and computational musicology could be utilised in the construction of systems designed to assist with the process of composition. It is also possible that such contributions could be adapted for use in music information retrieval systems and real-time improvisational or accompanying systems (*e.g.*, improving or adding more variety to the auto-harmonisation feature found in some MIDI keyboards). They may therefore also be considered indirect contributions to applied AI.

1.4 Thesis Overview

Chapter 2 This chapter presents a review of various techniques that are applicable to the computational modelling of music, as well as summarising previous research which employed such techniques for this purpose, with particular emphasis on the statistical modelling of melody and harmony. Related issues such as corpora, the representation of musical structure and evaluation are also discussed.

Chapter 3 The central ideas of the research are set out in detail in this chapter. The work is primarily concerned with comparing several different novel developments (versions 1, 2 and 3) of the multiple viewpoint framework originated by Conklin and Cleary (1988), as applied to harmony, with the objective of finding the best performing of them. In addition, it is concerned with the investigation of how models of melody could be improved by having access to a much larger pool of viewpoints than had been available in previous research. The viewpoint (feature) selection procedure introduced by Pearce (2005) is modified to avoid run times becoming excessively long when using this larger pool of viewpoints.

Chapter 4 This chapter investigates the related topics of viewpoint domains and time complexity. Guidance is given on how to reliably construct domains, culminating in a formal domain construction procedure for the most complex (version 3) viewpoints; and an empirical analysis of the time complexity of version 1 of the framework (as implemented as software) is carried out.

Chapter 5 An empirical analysis of the prediction performance of version 0 (melody only) is presented in this chapter. Different types of model are directly compared after the automatic selection of their multiple viewpoint systems. The better performing of the models are later used as the basis of the harmonic model comparisons. Significantly, the vast majority of the viewpoints selected for the best of the various types of model are new to this research.

Chapter 6 This chapter presents a similar analysis for harmonic versions 1, 2 and 3, demonstrating that the performance of these models increases with their complexity. On the other hand, the simplest model shows the biggest increase in performance when making use of an enlarged corpus which does not completely share the characteristics of the original.

Chapter 7 In this chapter we examine version 0 (melodic) multiple viewpoint systems and speculate on why certain viewpoints are selected from a music theoretic point of view. Metrical and phrase boundary regularities play an important role.

Chapter 8 In this chapter we investigate version 1, 2 and 3 multiple viewpoint systems in a similar way. We examine the more complex version 3 viewpoints as they evolve during the selection process. In this and the previous chapter we see many instances of predictions agreeing with intuitive or music theoretic expectations.

Chapter 9 The generation of melody and harmony by means of a modified random sampling technique is the subject of this chapter. Further quantitative evidence and an analysis of harmony produced by the best of the version 1, 2 and 3 models support the conclusion that the baseline version 1 has been improved upon. The models are not yet good enough to consistently generate high quality music in the style of the corpus, however.

Chapter 10 In this chapter, ideas for improving versions 0 to 3 and developing a further six increasingly complex versions of the multiple viewpoint framework for harmony are discussed in detail.

Chapter 11 This chapter presents a review of the thesis, a statement of the contributions made by this research and some general thoughts on the future direction of the research.

1.5 Publications

Content from the following peer reviewed papers and articles, which were written and published after the commencement of this research, appears without reference in this thesis:

> Whorley, R. P., Wiggins, G. A., and Pearce, M. T. (2007). Systematic evaluation and improvement of statistical models of harmony. In A. Cardoso and G. A. Wiggins, editors, *Proceedings of the 4th International Joint Workshop on Computational Creativity*, pages 81–88. London.
>
> Whorley, R. P., Pearce, M. T., Wiggins, G. A. (2008). Computational modelling of the cognition of harmonic movement. In K. Miyazaki, Y. Hiraga, M. Adachi, Y. Nakajima, M. Tsuzaki, editors, *Abstracts of the 10th International Conference on Music Perception and Cognition*, page 84. Sapporo, Japan.
>
> Whorley, R. P., Wiggins, G. A., Rhodes, C. S., and Pearce, M. T. (2010). Development of techniques for the computational modelling of harmony. In D. Ventura, A. Pease, R. Pérez y Pérez, G. Ritchie, A. Veale, editors, *Proceedings of the International Conference on Computational Creativity (ICCC-X)*,

pages 11–15. Lisbon, Portugal.

Whorley, R. P., Rhodes, C. S., Wiggins, G. A., and Pearce, M. T. (2013). Harmonising melodies: Why do we add the bass line first? In M. L. Maher, A. Veale, R. Saunders, O. Bown, editors, *Proceedings of the Fourth International Conference on Computational Creativity*, pages 79–86. Sydney, Australia.

Whorley, R. P., Wiggins, G. A., Rhodes, C. S., and Pearce, M. T. (2013). Multiple viewpoint systems: Time complexity and the construction of domains for complex musical viewpoints in the harmonization problem. To appear in *Journal of New Music Research*.

Chapter 2

Literature Review

2.1 Introduction

This chapter contains a review of previous research into the computational modelling of music, with emphasis on melodic modelling and the automatic harmonisation problem, which can be characterised as the task of finding a harmonisation (obeying the rules of harmony) for a given melody (Pachet and Roy, 1998). In §2.2, a digest of various computational techniques which could be of use in the current research is presented, including constraint satisfaction, genetic algorithms, finite context grammars, multiple viewpoint systems and graphical models. A discussion of corpora and the representation of musical structure (including the multiple viewpoint representation) appears in §2.3, following which the evaluation of computational models of music is discussed in §2.4. Finally, a summary of previous research on the computational modelling of music is presented in §2.5, with particular emphasis on the statistical modelling of melody and harmony. Please note that a more general discussion on the computational modelling of creativity is given in Appendix A.

2.2 Computational Methods

2.2.1 Constraint Satisfaction

Ovans and Davison (1992, p. 77) state the *constraint satisfaction problem* (CSP) as follows: "given a finite set of variables $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$ whose elements range respectively over the finite (and not necessarily numeric) domains $D_1, D_2, \ldots, D_n$, find a value for each variable such that a finite set of constraints is satisfied." Many solutions in solution space $D_1 \times D_2 \times \ldots \times D_n$ are likely to be invalid. A constraint, such as "a green square must not follow a yellow triangle," is a relation on a subset of $\mathcal{X}$ which contributes to the weeding out of invalid solutions. Rule-based *expert systems* make use of constraint satisfaction techniques.

CSPs can be solved by the depth-first search algorithm, which makes use of *back-*

tracking to escape from dead-ends in the search tree. Consistency techniques improve search efficiency by *early pruning* of the search space, thereby reducing the amount of backtracking. One such technique, *arc consistency*, applies to connected nodes in a *constraint graph* (Mackworth, 1977). Nodes in a constraint graph represent variables, while the connecting arcs represent constraints.

2.2.2 Genetic Algorithms

Holland (1975) first proposed the use of *genetic algorithms*, an overview of which (Wiggins et al., 1999) is summarised here. A genetic algorithm (GA) comprises the following:

- a *representation* for chromosomes, which are candidate solutions;
- an *initial population* of chromosomes;
- a set of *operators*, which generates new chromosomes from existing ones;
- a *fitness function*, which evaluates the chromosomes;
- and a *selection method*, which ensures that the fittest chromosomes are most likely to survive.

One of the simpler GAs has what is known as *steady state reproduction.* Firstly, an initial population of chromosomes is created, and they are all evaluated by the fitness function. An operator, which may be applied to one or more chromosomes, is randomly chosen. One or more *parent* chromosomes are selected, as appropriate, on the basis of the fitness evaluation, and the operator generates one or more child chromosomes. If a new chromosome is evaluated as being fitter than the least fit chromosome in the population, it replaces that chromosome. Another operator is randomly selected, and so on, until some pre-determined *stopping criterion* is reached. Note that a very different replacement strategy is implemented in a *generational* GA; in this case, a complete population is generated at each iteration.

Each operator has associated with it an *operator probability*, which governs the likelihood of application; they may also have behaviour modifying *parameters.* The two primary types of operator are *crossover* and *mutation.* Crossover is the exchange of information between two or more chromosomes, and a mutation changes one or more parts of a single chromosome.

When a suitable fitness function cannot be found (or relied upon), an *interactive* GA (or IGA) may be appropriate; in this case human judgement replaces the fitness function.

2.2.3 Finite Context Models

Finite context models are concerned with sequences of, for example, letters, words, musical notes or indeed almost anything. A generic sequence element will be referred to as an *event*. The mathematical treatment here generally follows that of Pearce (2005). First of all, some basic notation must be introduced: e_i is the i^{th} event in a sequence, and e_1^j represents the sequence $(e_1, \ldots, e_j)$. Each event is assumed to be of type τ, and to belong to a finite alphabet $[\tau]$.

2.2.3.1 Markov Models

The simplest type of finite context model is the *Markov model*. A *Markov chain* (*e.g.*, Bishop, 2006), widely used in natural language processing, is a sequence of random variables (representing, *e.g.*, letters of the English alphabet), where it is assumed that the probability of a particular symbol (the *prediction*) appearing is dependent (or conditional) upon a finite number of previous symbols (the *context*). This is an approximation of the probability of a symbol being dependent upon its entire history; it can therefore be expected that the longer the context, the more accurate the *transition probabilities*. A subsequence of symbols consisting only of a context and a prediction is called an *N-gram* (*e.g.*, Jurafsky and Martin, 2000), hence Markov models are alternatively known as N-gram models. The size of the context is known as the *order* of the Markov model; therefore for example a 2-gram (or *bigram*) is first-order, and an N-gram with no context, a *unigram*, is zeroth-order. Generally, an N-gram has a context of $n-1$ events; formally then, the *Markov assumption* is:

$$p(e_i|e_1^{i-1}) \approx p(e_i|e_{i-n+1}^{i-1}).$$

See Figure 2.1 and Figure 2.2 in §2.2.5 for graphical representations of first- and second-order Markov models respectively.

Conklin (1990) notes that the effectiveness of finite context models can be increased by using very long contexts, and describes some ways of achieving this. One way which does not increase model complexity too much is to use a *threaded context* (Andreae, 1977), where N-grams comprise symbols which are a fixed number of symbols apart.[1] An alternative to the Markov model is the hidden Markov model, which is described later.

Parameter Estimation A widely used, simple and effective technique for determining transition probabilities or parameters is *maximum likelihood estimation* (Manning and Schütze, 1999). For Markov models, this involves counting the number of times each N-gram context appears in a training corpus, and the number of times a particular symbol follows a particular context. The probability of a symbol appearing given the previous context is found by dividing the latter count by the former. For example, for the sequence of letters “abracadabra,” the first-order transition probability of prediction

[1] This definition is modified later with respect to threaded viewpoints.

"b" given the context "a" is:

$$P(b|a) = \frac{c(ab)}{c(a)} = 0.5$$

where $c(ab)$ is the frequency count of "ab" (the final "a" is not a context). In general:

$$p(e_i|e_{i-n+1}^{i-1}) = \frac{c(e_i|e_{i-n+1}^{i-1})}{\sum_{e\in[\tau]} c(e|e_{i-n+1}^{i-1})}.$$

Prediction and Generation Once a model has been induced from the training corpus, it can be used to *predict* existing data, which means assigning each symbol in the sequence of data a probability according to the model. The probability of the entire sequence is the product of the individual symbol conditional probabilities:

$$p(e_1^j) = \prod_{i=1}^{j} p(e_i|e_{i-n+1}^{i-1}).$$

Strictly speaking, for this equation to be true, padding symbols must appear before the start of the sequence to provide sufficient context. As an alternative, a zeroth-order model can be used to predict the first event, a first-order model to predict the second, and so on, until sufficient context is available to use a particular n^{th}-order model to predict the rest of the sequence.

It is also possible to *generate* novel sequences by *random sampling* of the model's probability distributions. Suppose, for example, that two symbols can follow a particular context with probabilities of 0.3 and 0.7 respectively. A random number between 0 and 1 is generated; if it is 0.3 or less the former symbol is chosen, and if it is greater than 0.3 the latter is chosen. This new symbol then becomes part of the context for the next event to be chosen, and so on. Other more sophisticated *Markov Chain Monte Carlo* sampling techniques can also be used (Bishop, 2006).

Smoothing Techniques It is often the case that contexts occurring while, for example, generating a harmony to a previously unseen melody, cannot be found in a model. Similarly, while predicting test data, it is often the case that even if a context is matched, the prediction does not appear in the associated probability distribution. The higher the order of the model, the more frequently these problems will occur. Conklin and Cleary (1988) and Allan (2002) describe the following simple *back-off* method (Katz, 1987), which has worked well with text. A number of models of different order are created. When searching for a particular context, the highest order model is checked first. If the context is not found, the second-highest order model is checked, and so on until the (progressively smaller) context is found. At this point, a symbol can be generated by, for example, random sampling from the probability distribution. If the model is being used to predict test data, however, and the required prediction is not in the probability distribution, then one possible naïve solution is to carry on checking lower-order models until the prediction is found (the previously checked models are discarded). There is a problem with this method, however, as described in the following paragraph.

Conklin (1990) notes that probability distributions must be complete (*i.e.*, they must cover all possible events). One reason for this is easy to see. Let us imagine that a harmonisation is being generated. The highest order model is checked first, and the context is matched. There may be, for example, three events in the distribution (the probabilities of which sum to one), and one of these is chosen (*e.g.*, by random sampling). Now let us imagine that probabilities are being assigned to events in test data, and that precisely the same context is found. On this occasion, however, the next event in the test data is not present in the distribution; therefore it is necessary to try a lower-order model. The event, and its associated probability, is found in this distribution; but it is now clear that with respect to the original context, the sum of the probabilities is greater than one.

A very simple way of completing probability distributions is to use *additive smoothing* (*e.g.*, Nivre, 2000), in which a small number (usually either 0.5 or 1) is added to all counts[2] associated with all contexts in all models. A much better back-off smoothing method is *Prediction by Partial Match*, or *PPM* (Cleary and Witten, 1984), which uses *full blending* to determine a complete distribution. Predictions of the maximum model order are assigned a proportion of the probability mass, with the rest, called the *escape probability*, being passed to the next-largest order, and so on. The escape probability is dependent upon the particular escape method used; Witten and Bell (1989) review methods A, B, C, P, X and XC (Pearce, 2005, additionally reviews methods D and AX). As the models are traversed during the construction of the probability distribution, events predicted by higher-order models are seen to be predicted again by lower-order models. There is a good reason to suppose that events already predicted should be excluded from consideration at lower orders; that is, order i predictions are a subset of those of order $i-1$. In practice, *exclusion* is sometimes used and sometimes not. The escape probability passed on from the zeroth-order model is shared between events unseen in the training corpus (assuming exclusion is used), and this completes the distribution. Formally, most escape methods are specific instances of the following general equation (Kneser and Ney, 1995):

$$p(e_i|e_{i-n+1}^{i-1}) \quad = \quad \begin{cases} \alpha(e_i|e_{i-n+1}^{i-1}) & \text{if } c(e_i|e_{i-n+1}^{i-1}) > 0 \\ \gamma(e_{i-n+1}^{i-1})p(e_i|e_{i-n+2}^{i-1}) & \text{if } c(e_i|e_{i-n+1}^{i-1}) = 0 \end{cases}$$

where $\alpha()$ is the probability estimate and $\gamma()$ is the escape probability. A variant of PPM, called PPM*, employs models of unlimited order (Cleary and Teahan, 1997). A heuristic which works reasonably well is to begin back-off with the shortest context having only one prediction. In the absence of such a context, back-off begins with

[2] Importantly, including all zero counts; therefore all possible events must be identified in advance.

the longest context. Bunton (1997) improves on this with an information-theoretic "percolating" state selection procedure.

Interpolated smoothing is more complicated than back-off smoothing, but Chen and Goodman (1999) find the former to be better for low counts. Pearce (2005) notes that the method is a recursive weighted combination of i^{th}- and $(i-1)^{\text{th}}$-order models, and gives the following equation:

$$p(e_i|e_{i-n+1}^{i-1}) = \alpha(e_i|e_{i-n+1}^{i-1}) + \gamma(e_{i-n+1}^{i-1})p(e_i|e_{i-n+2}^{i-1})$$

where again, $\alpha()$ is the probability estimate and $\gamma()$ is the escape probability.

2.2.3.2 Hidden Markov Models

A *hidden Markov model*, or *HMM* (*e.g.*, Manning and Schütze, 1999), is a Markov chain consisting of a sequence of hidden states, where (it is assumed that) the probability of a particular hidden state appearing is dependent upon a finite number of previous hidden states. In addition to state transition probabilities, there are a number of *emission probabilities* associated with each state giving rise to observed events. The process of computing the probability of an observation sequence, given a model, is known as *decoding*. This can be achieved using any combination of the *forward procedure* and the *backward procedure*. The *Viterbi algorithm* (Viterbi, 1967) is used to find the most likely sequence of hidden states, given an observation sequence and a model. Model parameters are estimated from a training observation sequence using a special case of the *Expectation Maximisation* (EM) method (Dempster et al., 1977), which is called the *Baum-Welch* or *forward-backward algorithm* (Baum and Petrie, 1966). A detailed account of these algorithms can be found in Manning and Schütze (1999). See Figure 2.3 in §2.2.5 for a graphical representation of a first-order HMM.

2.2.3.3 Hierarchical Hidden Markov Models

Weiland et al. (2005) describe *hierarchical hidden Markov models* (Fine et al., 1998), and a way of improving their performance, as follows. Hierarchical hidden Markov models (HHMMs) comprise internal, production and end states, which are organised on an arbitrary number of hierarchical levels in the form of a tree. Internal states contain probabilistic information about transitions to sibling and child states; production states are childless, but are able to emit symbols (with particular emission probabilities); and end states force a return to the level above. Note that the structure of HHMMs must be determined in advance of training, and they are not subsequently able to self-adapt (Fine et al., 1998).

Unfortunately, the algorithms originally developed for these models[3] (Fine et al.,

[3] The algorithms are for calculating the likelihood of a sequence; finding the most probable state sequence; and estimating the parameters of a model.

1998), which are generalisations of those used for HMMs, have a time complexity of $O(NT^3)$, where N is the number of states and T is the size of the training corpus. Wierstra (2004) has improved on this, however, with an implementation having a time complexity of $O(N^2T)$. The idea is to transform an HHMM into its flat equivalent (*i.e.*, an HMM), and use this whenever possible. HHMMs have two possible forms: maximally self-referential (MaxSR) and minimally self-referential (MinSR). The former allows internal states to have self-referential loops, but the latter does not (note that both forms allow such loops on production states). In order be transformed into an HMM, an HHMM must be in minimally self-referential form; therefore a MaxSR HHMM must first be converted to a MinSR HHMM.

2.2.4 Multiple Viewpoint Systems

Informally, *multiple viewpoint systems* (Conklin, 1990; Conklin and Witten, 1995) comprise more than one finite context model (in practice, N-gram models), the predictions of which are combined to give an overall prediction. The following formal description, including the example application to the card game Eleusis (the rules of which are unimportant), is summarised from Conklin (1990). The nomenclature and mathematical treatment generally follows that of Pearce (2005). First of all, however, some additional basic notation must be introduced. Event space $\mathcal{E}$ is the set of representable events, and S^* is the set of valid sequences derived from elements of a set S. Furthermore, $c :: e$ denotes the addition of an event to an existing sequence c.

A *type* τ is an attribute or property of an event. *Basic* types are the more concrete properties used to define the event space. In the case of Eleusis, an event is a turn of a card, which is completely described by the tuple $\langle rank, suit \rangle$ (*e.g.*, $\langle 3, \clubsuit \rangle$ or $\langle K, \diamondsuit \rangle$). *Derived* types are more abstract, such as the difference in rank (mod 13) between two successive cards (`rankint`) or the colour of a card (`colour`), and they are derived from one or more basic types. Each type τ has an associated partial function Ψ_τ which maps event subsequences to an element of type τ; for basic types, Ψ_τ is a *projection function.* Usually, only the most recent one or two events in the sequence are involved in the mapping; for example, $\Psi_{\texttt{rankint}}$ maps $\langle 2, \spadesuit \rangle$, $\langle 7, \heartsuit \rangle$ to the `rankint` element 5 (note that $\Psi_{\texttt{rankint}}$ is undefined for the first event in a sequence, in which case it returns $\perp$). *Long-range* derived types are exceptions to the rule that only the most recent events are used by Ψ_τ; here, non-adjacent events can be used in the mapping. For example, type `tri` (threaded rank interval) is the difference in rank between $card_n$ and $card_{n-2}$ (note that this is really a long-range type). A true *threaded* type is defined only at certain positions in a sequence; these are positions where a Boolean *test* type is true (Conklin and Anagnostopoulou, 2001). A threaded type includes a measure of the distance between adjacent threaded events, and so is a *product type* (see below). Note that basic and derived types (excluding threaded types) are also known as *primitive* types.

$[\tau]$ denotes the set of valid type τ elements, called the *syntactic domain* of τ (*e.g.*,

$[\texttt{rankint}] = \{0, \ldots, 12\}$ and $[\texttt{colour}] = \{red, black\}$); therefore $[\tau]^*$ denotes the set of valid type τ sequences. A *viewpoint* consists of a partial function $\Psi_\tau : \mathcal{E}^* \rightharpoonup [\tau]$ and a finite context model of sequences in $[\tau]^*$. A *linked viewpoint* models interactions or dependencies by utilising a *product type* $\tau_1 \otimes \ldots \otimes \tau_n$ between n primitive types (*e.g.*, $\texttt{rankint} \otimes \texttt{colour}$), where $[\tau] = [\tau_1] \times \ldots \times [\tau_n]$ (*e.g.*, $[\texttt{rankint} \otimes \texttt{colour}] = \{\langle 0, red\rangle, \langle 0, black\rangle, \ldots, \langle 12, red\rangle, \langle 12, black\rangle\}$). For $\tau = \tau_1 \otimes \ldots \otimes \tau_n$, $\Psi_\tau(e_1^j)$ is a tuple $\langle \Psi_{\tau_1}(e_1^j), \ldots, \Psi_{\tau_n}(e_1^j)\rangle$, unless any of the constituents of the tuple is undefined, in which case $\Psi_\tau(e_1^j)$ is undefined. Note that event space $\mathcal{E} = [\texttt{rank}] \times [\texttt{suit}]$. More generally, for n basic types: $\mathcal{E} = [\tau_1] \times \ldots \times [\tau_n]$.

A multiple viewpoint system comprises more than one viewpoint, and is a subset of the power set of the set of primitive types. For example, for the set of primitive types

$$\{\texttt{rank}, \texttt{suit}, \texttt{rankint}\},$$

the power set is (using the above product type notation)

$$\{\emptyset, \texttt{rank}, \texttt{suit}, \texttt{rankint}, \texttt{rank} \otimes \texttt{suit}, \texttt{rank} \otimes \\ \texttt{rankint}, \texttt{suit} \otimes \texttt{rankint}, \texttt{rank} \otimes \texttt{suit} \otimes \texttt{rankint}\};$$

therefore, for example, $\{\texttt{suit}, \texttt{rankint}\}$ and $\{\texttt{rank}, \texttt{suit} \otimes \texttt{rankint}\}$ are both multiple viewpoint systems.

The *type set* of a type τ is represented by $\langle\tau\rangle$. This set enumerates the basic types that a viewpoint modelling τ is able to predict (*i.e.*, the basic types from which it is derived); for example, $\langle\texttt{rankint}\rangle = \{rank\}$. For a linked viewpoint comprising n primitive types (Pearce, 2005):

$$\langle\tau\rangle = \bigcup_{k=1}^{n} \langle\tau_k\rangle.$$

The *semantic domain* of a type τ is denoted by the set $[\![\tau]\!]$, and $[\![\cdot]\!]_\tau$ is a function from $[\tau]$ to $[\![\tau]\!]$. For example, $[\![\texttt{faced}]\!] = \{is \ldots, is\ not\ a\ royal\ card\ (J, Q, K)\}$ and $[\texttt{faced}] = \{true, false\}$; therefore, for instance, $[\![\cdot]\!]_{\texttt{faced}}$ maps *false* to *is not a royal card (J, Q, K)*. For a linked viewpoint comprising n primitive types, $[\![\tau]\!] = [\![\tau_1]\!]$ and $\ldots$ and $[\![\tau_n]\!]$.

Formally, for each viewpoint, the event sequence $c :: e$ in $\mathcal{E}^*$ must be converted to a sequence in $[\tau]^*$ using the function $\Phi_\tau : \mathcal{E}^* \to [\tau]^*$, which is defined as follows:

$$\begin{aligned} \Phi_\tau(()) &= (), \\ \Phi_\tau(e_1^j) &= \begin{cases} \Phi_\tau(e_1^{j-1}) :: \Psi_\tau(e_1^j) & \text{if } \Psi_\tau(e_1^j) \text{ defined} \\ \Phi_\tau(e_1^{j-1}) & \text{otherwise.} \end{cases} \end{aligned}$$

In practice, for the sake of efficiency, the set of primitive type sequences so constructed (in this case including $\perp$) are stored in a *solution array* (Ebcioğlu, 1988). Part of an Eleusis solution array (*i.e.*, containing only the example types above) is shown in Table 2.1.

Type	Event number												
	1	2	3	4	5	6	7	8	9	10	11	12	13
`rank`	2	3	4	9	6	7	8	5	10	J	Q	J	A
`suit`	♣	♢	♣	♡	♠	♢	♣	♡	♠	♢	♢	♡	♢
`faced`	F	F	F	F	F	F	F	F	F	T	T	T	F
`colour`	B	R	B	R	B	R	B	R	B	R	R	R	R
`rankint`	$\perp$	1	1	5	10	1	1	10	5	1	1	12	3
`tri`	$\perp$	$\perp$	2	6	2	11	2	11	2	6	2	0	2

Table 2.1: Part of the solution array for the first thirteen cards in a game of Eleusis (adapted from Conklin, 1990).

Product types can easily be modelled from the relevant primitive type sequences. Bearing in mind that undefined elements are ignored, care must be taken during parameter estimation (see §2.2.3) to ensure that productions are counted accurately.

A finite context model of arbitrary highest order $\hbar$ is created for each viewpoint, exemplified by type τ (the order does not have to be the same for each viewpoint). Prediction sets comprising all possible viewpoint elements $[\tau]$ are created using PPM (see §2.2.3). For derived types, it is then necessary to convert these to prediction sets comprising the basic type or types in $\langle\tau\rangle$. This is achieved by using the partial function Ψ_τ in reverse on each of the viewpoint elements, bearing in mind that more than one basic viewpoint element could result from each such application of Ψ_τ, since it is not a bijection. In such cases, the probability of the viewpoint element is divided equally between the possible basic viewpoint elements. The completion of these basic type prediction sets must be ensured (*i.e.*, all possible elements must be represented).

The overall structure of a multiple viewpoint system can vary, but Conklin and Witten (1995) use one consisting of a short-term model and a long-term model. The predictions of all the viewpoints in the short-term model are combined using weighted linear combinations (Hamburger, 1986); similarly for all those in the long-term model. These two predictions are then combined using a Dempster-Schafer scheme (Garvey et al., 1981) to give a final prediction.

The method for combining viewpoint predictions (within either the short-term or the long-term model) will now be examined in detail. In order to minimise computational complexity, prediction proceeds in stages; one for each of the basic attributes which make up an event. At any stage, only those viewpoints capable of predicting the relevant attribute are *activated*. Other attributes will already have been instantiated in any previous stages; therefore in practice only basic viewpoint elements which *unify* with these attributes appear in prediction sets. If the prediction sets are sorted such that the j^{th} element of each set is the same (although very likely with different probabilities), the combined probability of this element is given by a weighted arithmetic mean. For N viewpoints:

$$p(j) = \frac{\sum_{i=1}^{N} p_i(j) w_i}{\sum_{i=1}^{N} w_i}.$$

The idea is to give more weight to viewpoints which are capable of predicting with more certainty. A prediction set in which the probability distribution is completely uniform will predict with a great deal of uncertainty; the less uniform the distribution, the greater the certainty. The uncertainty of a distribution $[\tau]$ is given by the Shannon entropy function:

$$H([\tau]) = -\sum_{j=1}^{|[\tau]|} p(j) \log_2 p(j).$$

Maximum uncertainty is given by:

$$H_{max} = \log_2 |[\tau]| .$$

What Conklin (1990) calls *relative entropy*, Re, is a measure of the uncertainty of a distribution with respect to maximum uncertainty. Lower Re indicates greater certainty, which warrants greater weighting. It is defined as:

$$Re([\tau]) = \begin{cases} H([\tau])/H_{max} & H_{max} > 0 \\ 1 & \text{otherwise.} \end{cases}$$

Note that this is not the same as *Kullback-Leibler* (KL) *divergence* (Kullback and Leibler, 1951), which is also known as relative entropy (Bishop, 2006). Finally, an exponential bias $b \in N$ is introduced into the weighting function which favours distributions with low Re:

$$w_i = Re([\tau])^{-b}.$$

Pearce (2005) introduces a new method for combining viewpoint predictions; namely, a weighted geometric mean rather than a weighted arithmetic mean. For N viewpoints:

$$p(j) = \frac{1}{R} \left(\prod_{i=1}^{N} p_i(j)^{w_i} \right)^{\frac{1}{\sum_{i=1}^{N} w_i}}$$

where R is a normalisation constant. The weights can be calculated as a function of Re in precisely the same way as for the arithmetic mean. Tax et al. (2000) explored the issue of combination with respect to multiple classifier systems. They found that, for classifiers using independent data representations, an unweighted geometric combination scheme performed better than an unweighted arithmetic one. Pearce (2005) predicts and demonstrates that, since weighting improves the performance of arithmetic combination, weighting will similarly improve the performance of geometric combination.

Pearce (2005) also introduces a viewpoint (or feature) selection algorithm which constructs multiple viewpoint systems according to objective criteria. Prior to this, viewpoints were chosen on the basis of human judgement, resulting in a limited number of systems for comparison. The basis of the algorithm is *forward stepwise selection*; the system is gradually built up, starting with the empty set (*N.B.*, *backward stepwise elimination* is a possible alternative). At each iteration, all single primitive or linked viewpoint deletions are considered. If any deletions are found to improve the model, the viewpoint making the biggest improvement is removed from the system. The algorithm continues to look for improvements in this way until there are no deletions which further improve the system. At this point, all single viewpoint additions are considered, with the one resulting in the largest improvement making its way into the system. The algorithm then reverts to looking for deletions. When neither additions nor deletions are able to improve the model, the system is complete (although not necessarily globally optimal). See, for example, Aha and Bankert (1996) for the application of stepwise selection to machine learning.

2.2.5 Graphical Models

Graphical models are "a family of techniques which exploit a duality between graph structures and probability models" (Smyth, 1997, p. 1261). Given a set of random variables $\mathbf{x} = \{x_1, \ldots, x_N\}$ with joint distribution $p(\mathbf{x})$, conditional independence in $p(\mathbf{x})$ is represented by means of an annotated graph. *Nodes* (or *vertices*) represent the random variables in $p(\mathbf{x})$; and the *links* (or *edges*, or *arcs*) between them indicate the independence structure, which is exploited in order to achieve tractability (Smyth, 1997). Three useful properties of probabilistic graphical models are:

> 1. They provide a simple way to visualise the structure of a probabilistic model and can be used to design and motivate new models.
> 2. Insights into the properties of the model, including conditional independence properties, can be obtained by inspection of the graph.
> 3. Complex computations, required to perform inference and learning in sophisticated models, can be expressed in terms of graphical manipulations, in which underlying mathematical expressions are carried along implicitly.
>
> (Bishop, 2006, pp. 359–60)

Smyth (1997, pp. 1262–1263) notes that graphical models can be subdivided into two main types: *Bayesian networks* or *directed graphical models*, also known as "belief networks, recursive graphical models, [...] causal networks, directed Markov networks, and probabilistic (causal) networks;" and *Markov random fields* or *undirected graphical models*, otherwise known as "Markov networks, Boltzmann machines, and log-linear models." Bayesian networks are particularly useful in fields where there are definite cause-effect relationships, such as statistics and artificial intelligence. Markov random fields are more useful for correlational relationships, such as are found in image processing and statistical physics.

2.2.5.1 Bayesian Networks

Links in directed graphs have an arrowhead at one end, signifying that the node near the arrowhead (the *child*) is dependent upon the node to which it is linked (the *parent*). A node may have more than one parent. Note that there must be no *directed cycles* in a Bayesian network; therefore the specific graphs under consideration here are *directed acyclic graphs* (DAGs). Local conditional probabilities are used in the factorisation of $p(\mathbf{x})$, so the joint distribution is

$$p(\mathbf{x}) = \prod_{n=1}^{N} p(x_n|\mathrm{pa}_n)$$

where pa_n denotes the set of parents of x_n (Bishop, 2006).

Murphy (2002) gives a detailed account of *dynamic Bayesian networks* (DBNs), which are Bayesian networks modelling sequential data unfolding with time, where Z_t is a collection of random variables at time t. A DBN is the pair $(B_1, B_{\rightarrow})$, where B_1 is a Bayesian network defining the prior $P(Z_1)$, and $B_{\rightarrow}$ is a two-slice temporal Bayesian network defining $P(Z_t|Z_{t-1})$:

$$P(Z_t|Z_{t-1}) = \prod_{i=1}^{N} P(Z_t^i|\mathrm{pa}(Z_t^i))$$

where Z_t^i is the i^{th} random variable at time t. Parents $\mathrm{pa}(Z_t^i)$ may be at time t or $t-1$, meaning that the model is assumed to be 1^{st}-order.

Many well known probabilistic models can be expressed within this framework (Cemgil, 2006; Murphy, 2002), and some of them are used here as examples. A 1^{st}-order Markov model is represented as shown in Figure 2.1, with

$$p(\mathbf{x}) = p(x_1)p(x_2|x_1)p(x_3|x_2)p(x_4|x_3).$$

A 2^{nd}-order Markov model is represented as shown in Figure 2.2, with

$$p(\mathbf{x}) = p(x_1)p(x_2|x_1)p(x_3|x_2, x_1)p(x_4|x_3, x_2).$$

Although DBNs are, strictly speaking, 1^{st}-order models, Murphy (2002, p. 15) notes that "this is mostly for notational simplicity: there is no fundamental reason why we

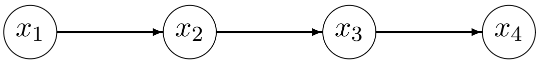


Figure 2.1: A first-order Markov model (adapted from Cemgil, 2006).

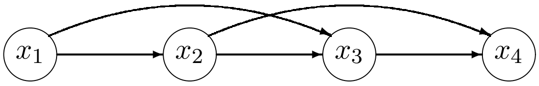


Figure 2.2: A second-order Markov model (adapted from Cemgil, 2006).

cannot allow arcs to skip across slices."

A 1st-order hidden Markov model (HMM) is represented as shown in Figure 2.3, with

$$p(\mathbf{h}, \mathbf{x}) = p(h_1)p(h_2|h_1)p(h_3|h_2)p(h_4|h_3)p(x_1|h_1)p(x_2|h_2)p(x_3|h_3)p(x_4|h_4).$$

Inference is the calculation of posterior distributions for unknown variables, given a model and nodes that are fixed at particular values; and *maximum a posteriori* (MAP) identification is the finding of the most probable state of a set of unknown variables. Inference in a first-order HMM is normally carried out by the *forward-backward* algorithm, and the MAP problem is solved by the *Viterbi* algorithm; but since an HMM can be represented as a graphical model, standard graphical model algorithms can also be used. In fact the graphical model algorithms, being completely general, are also applicable to arbitrary-order HMMs (Smyth, 1997). Extrapolating further, it is clear that very complex probabilistic models can be represented as dynamic Bayesian networks, and they can all be handled by the graphical model algorithms (subject to the usual constraints of time and space complexity); see §2.2.5.4 below.

Graphical model inference techniques often make use of undirected graphs; fortunately, it is easy to convert a directed graph into an undirected one, although the latter may well be a less efficient representation. The conversion method is explained in §2.2.5.3 below.

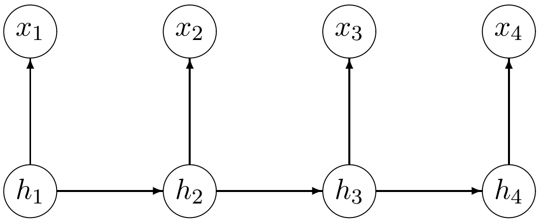


Figure 2.3: A first-order hidden Markov model (adapted from Cemgil, 2006).

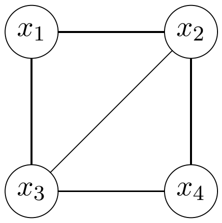


Figure 2.4: Nodes x_1 and x_2 form a clique, and nodes x_2, x_3 and x_4 form a maximal clique (adapted from Bishop, 2006).

2.2.5.2 Markov Random Fields

This is a brief overview of the treatment of Markov random fields given by Bishop (2006). Links in undirected graphs do not have arrowheads; they are correlational links rather than causal ones. This means that the way joint distributions are factorised is different. In order to explain how it is done, the concept of a *clique* must first be introduced. A clique is simply a subset of nodes that is fully connected. A *maximal clique* is one to which another node cannot be added without it ceasing to be a clique; see Figure 2.4.

Let C be a maximal clique, and $\mathbf{x}_C$ be the set of variables it contains. The joint distribution is the product of *potential functions* $\psi_C(\mathbf{x}_C)$ over the maximal cliques:

$$p(\mathbf{x}) = \frac{1}{Z}\prod_C \psi_C(\mathbf{x}_C)$$

where Z is a normalisation constant sometimes called the *partition function*:

$$Z = \sum_{\mathbf{x}}\prod_C \psi_C(\mathbf{x}_C).$$

By insisting that $\psi_C(\mathbf{x}_C) \geq 0$, we ensure that $p(\mathbf{x}) \geq 0$. In general, the potential functions cannot necessarily be said to have any particular probabilistic interpretation; however, if for example an undirected graph has been constructed from a directed graph, the potential functions may well have such an interpretation.

2.2.5.3 Converting Bayesian Networks to Markov Random Fields

Bishop (2006) notes that it is sometimes useful to convert a directed graph to an undirected one so that exact inference techniques associated with undirected graphs can be used. The conversion process he describes is very simple. For each node in a directed graph, undirected links are added such that the parents of a node are connected to each other; the arrowheads are then removed from the original links. All maximal clique potentials are initialised to 1, and then they are multiplied by the relevant joint distribution factors from the directed graph. In all cases $Z = 1$. A simple example is shown

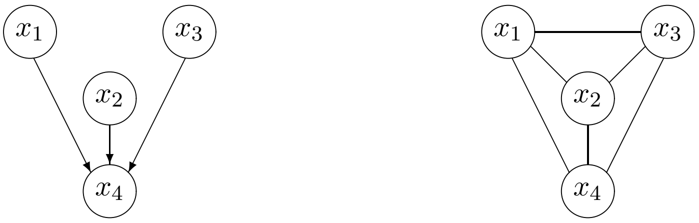


Figure 2.5: Conversion of a directed graph to an undirected graph (adapted from Bishop, 2006).

in Figure 2.5, where for the directed graph

$$p(\mathbf{x}) = p(x_1)p(x_2)p(x_3)p(x_4|x_1, x_2, x_3)$$

and for the undirected graph

$$p(\mathbf{x}) = \psi_{1,2,3,4}(x_1, x_2, x_3, x_4) = p(x_1)p(x_2)p(x_3)p(x_4|x_1, x_2, x_3).$$

Note that the undirected graph in this example shows no conditional independence structure. In general, "the two types of graph can express different conditional independence properties" (Bishop, 2006, p. 392).

Some directed graphs result in undirected graphs which look exactly the same, except that the arrowheads are removed from the links (*i.e.*, no additional links are required). The first-order Markov model graph shown in Figure 2.1 is a case in point; each node has precisely one parent, except the first, which has none, so no additional links are required. The first-order hidden Markov model shown in Figure 2.3 is a similar case. In Figure 2.2, however, some of the nodes have two parents; but in each case, the parents are already linked, resulting once again in an undirected graph with no additional links.

2.2.5.4 Graphical Model Algorithms

This section provides a very brief overview of relevant algorithms, many of which are expressed in terms of the passing of local *messages*. For exact inference in tree-structured graphs, the problem of evaluating local marginals over nodes or subsets of nodes (inference) is addressed by the *sum-product* algorithm (a generalisation of the forward-backward algorithm), and the most probable state (MAP) is found by the *max-sum* algorithm (a generalisation of the Viterbi algorithm). For arbitrary graph topologies, there is an exact inference procedure called the *junction tree* algorithm. See, for example, Bishop (2006) for more information.

It is instructive to note at this point that both directed and undirected graphs can be represented by *factor graphs*, which have additional nodes for the factors, as exemplified

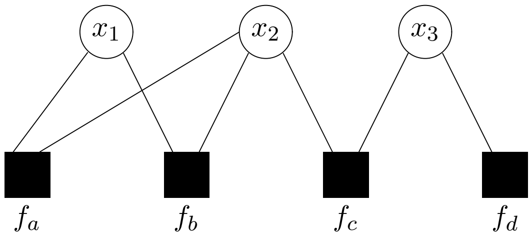


Figure 2.6: A factor graph (adapted from Bishop, 2006).

by Figure 2.6, where

$$p(\mathbf{x}) = f_a(x_1, x_2) f_b(x_1, x_2) f_c(x_2, x_3) f_d(x_3).$$

It turns out that these graphs are useful for deriving the sum-product algorithm.

In practice, it is possible for undirected graphs to have many loops, necessitating techniques for approximate inference such as *variational* methods, (*iterative*) *sampling* or *Monte Carlo* methods, and *loopy belief propagation*. Note that recent work has started to go beyond the inference problem, focusing on learning the graph structure itself from data (Friedman and Koller, 2003). This requires a search of the space of possible structures and a measure with which to evaluate each structure.

2.3 The Corpus and the Representation of Musical Structure

2.3.1 The Corpus

Corpora are widely used in statistical natural language processing. They are large bodies of text, often marked up with linguistic (*e.g.*, part of speech) information, which implies a certain amount of pre-processing. “The general issue is whether the corpus is a *representative sample* of the population of interest” (Manning and Schütze, 1999, p. 119). As far as music is concerned, there are often not many suitable pieces of a particular style; therefore it is best to include as many as possible in the corpus.

Conklin and Cleary (1988) use two corpora, each comprising one piece of music. A single monodic Gregorian chant in the Dorian mode is used to model melody, and a sixteenth century work, *Oculus Non Vidit* by Lasso, is used to model two-part polyphony. These pieces are used because the music is simple, regular and yet flexible; and they follow well established rules, with which generated pieces can be inspected for conformance. It is recognised that the small quantity of data results in sampling problems, however.

In later work on the modelling of melody, Conklin (1990) chooses to use chorale melodies that were harmonised by J. S. Bach due to the fact that they are uncomplicated, plentiful and melodically well formed. Allan (2002) also uses three hundred and eighty-four chorales harmonised by J. S. Bach in his research on the modelling of harmony. He notes that although Bach's harmonisations range from simple to complex, they seem to share a coherent musical style. The data is divided into two corpora; one for major key and the other for minor key chorales. Each corpus is further randomly divided into a training set (40%) and three test sets (20% each); one test set is used exclusively to evaluate the finished model.

Ponsford et al. (1999), on the other hand, prefer to use eighty-four seventeenth century French *sarabandes*, written by various composers. The choice is made on the basis of dance music of the period being strongly harmonic, with sarabandes being less simple (and therefore more interesting) than some other dance forms.

2.3.2 Musical Structure

A piece of music is played and listened to from beginning to end over a period of time. Clearly what is happening as "now" moves through the piece is of the utmost importance; but what happens now is understood in the context of what has previously occurred, and also sets up expectations as to what might happen next. These expectations can be fulfilled or thwarted (Narmour, 1992); therefore our understanding of what we are hearing now can be modified in retrospect. Not only is there an appreciation of the local structure of a piece, but also of its overall structure; especially after repeated hearings have made it familiar. This suggests that there might be some sort of grammar which is able to describe musical structure, and with which we are able to understand large scale musical works.

A method of musical analysis which goes beyond merely describing chord progressions was developed by Schenker (1979). Smoliar (1980, p. 111) notes that:

> According to Schenker's theories, an entire tonal composition could arise from a series of [...] elaborations (which he called *diminutions*), compounded ultimately upon a single note — the "tone" of the composition's tonality.

In performing a Schenkerian analysis, this idea is turned on its head. Starting from a complete composition, a series of *reductions* is made in which more and more of the musical detail is removed at each stage. This is a largely subjective process which is intended to reveal ever higher levels of musical structure, until eventually an irreducible *Ursatz* is reached.

Smoliar (1980) has developed a computational tool to help music theorists perform Schenkerian analyses. He notes that Schenker's theories are compatible with the idea of transformational grammars. As a result, he represents musical structure in the form of a tree, in a similar way to which the parsing of a sentence can be represented by a tree. Transformations modify the structure of a tree; in other words, "a transformation is simply a means for taking one tree and rewriting it as another tree"(Smoliar, 1980,

p. 111). Trees comprise musical *events*, which have three possible forms:

1. a single note;
2. a *SEQuence* of events, occurring in a designated order;
3. a *SIMultaneity* of events, all beginning concurrently.

(Smoliar, 1980, p. 111)

Note that this idea was later taken up by Conklin (2002); see §2.5.3.2.

Lerdahl and Jackendoff (1983) have developed a generative theory of tonal music which combines cognitive psychology with generative linguistics. It is a formal musical grammar which describes four hierarchical components of musical intuition. These components are: *grouping structure*, which deals with the segmentation of music into, for example, motifs, phrases and sections; *metrical structure*, which describes the relationship between strong and weak beats at various hierarchical levels; *time-span reduction*, which finds the relative structural importance of pitches at different hierarchical levels; and *prolongational reduction*, which deals with, for example, hierarchies of musical tension and relaxation. The reductions represent musical structure in the form of a tree (*cf.* Smoliar, 1980). The theory has three types of rule: *well-formedness rules*, which enumerate possible structural analyses; *preference rules*, which choose from the possible analyses those which are most likely to coincide with the intuition of experienced listeners; and *transformational rules*, which take account of exceptions to the strict hierarchies described by the previous types of rule (*e.g.*, elisions, which are slight overlaps between phrases).

2.3.3 File Formats

MIDI (Musical Instrument Digital Interface) files are commonly created and played back by *sequencers*; computer programs developed for this purpose. Music can most easily be put into this format by connecting a keyboard to a sequencer via a MIDI interface. Different instrumental (or other) sounds can be made to play back concurrently (*e.g.*, through a synthesiser module) by inputting music into different channels, and assigning a different instrumental sound to each channel. A MIDI file consists of a list of events, each with its own starting time (measured in *ticks*), and each of which is described by a *message*. Events may have general effects, such as specifying a channel's *program* (instrumental sound), reverb, panning (stereo position) or volume; but most often they are specific commands such as *Note On* (which includes pitch) or *Note Off*. It is these latter commands which are particularly relevant for our purposes. See Rothstein (1992) for further information. Ponsford et al. (1999) create MIDI files of the sarabandes in their corpus as a first step, after which the data is pre-processed and annotated (see §2.3.4.3).

Allan (2002) uses text files from Bach (1998). Allan prefers text files because they are easy to read (*cf.* MIDI), easy to annotate with additional data, and easy to process.

A typical file begins with global information such as the key, time signature and tempo. The remainder of the file is a representation of a chorale using combinations of note name and octave such as "C#2," which is the $C\sharp$ above "C#1." This pitch information is arranged in four columns (one for each voice: soprano, alto, tenor and bass), and an arbitrary number of rows, which are spaced according to the time interval between the beginnings of notes. Taking the file encoding the chorale *Ach bleib' bei uns, Herr Jesu Christ* (*bch001.txt*) as an example (see Figure 2.7), chords generally appear at crotchet intervals, with quaver movement indicated by the presence of pitch information between these rows. Harmonic function symbols (*e.g.*, "T," representing a tonic triad) are also shown at crotchet intervals, and bars and phrases are numbered.

Pearce (2005) obtains his corpora of folk and hymn music in a different text file representation: the `**kern` format (Huron, 1997). Figure 2.8 shows a `**kern` encoding of the beginning of the same chorale, *Ach bleib' bei uns, Herr Jesu Christ* (Bach 2000, *0253-01.krn*), for purposes of comparison. This encoding precisely reflects what is in the score, including clefs, key signatures and the text, in addition to note pitches and durations. It is not annotated with harmonic function symbols.

2.3.4 Data Structures

This section focuses on the multiple viewpoint representation as applied to music, and the chord representation work of Paiement et al. (2005a). There is also a brief discussion of some other ideas on the representation of music as data structures.

2.3.4.1 Multiple Viewpoint Representation

Viewpoint names are based upon those in Pearce and Wiggins (2006), as they are generally more intelligible than names used in previous research; the latter are given in parentheses.

Conklin and Witten (1995) describe the use of multiple viewpoint systems (see §2.2.3) for the prediction and generation of chorale melodies. Events are musical notes, and event space $\mathcal{E}$ is $[\texttt{Pitch}] \times [\texttt{KeySig}] \times [\texttt{BarLength}] \times [\texttt{Fermata}] \times [\texttt{Onset}] \times [\texttt{Duration}]$. Type `Pitch` (previously `cpitch` and `pitch`) is represented by its MIDI value, and `KeySig` (previously `keysig`) is described in terms of the number of sharps (positive integer) or flats (negative integer). Type `BarLength` (previously `barlength` and `timesig`) is represented by the number of semiquavers in a bar; `Onset` (previously `onset` and `st`) is the start-time of an event, measured in semiquavers, assuming time zero on the first beat of the first bar (even if the melody begins later in the bar); and `Duration` (previously `dur` and `duration`) is also measured in semiquavers. Type `Fermata` (previously `fermata`) distinguishes between events which are associated with a fermata and those which are not. It should be noted that rests are not events in this formalism.

```
Choralname = bch001
Anzahl Stimmen = 4
Tonart = A-dur
Takt = 4/4
Tempo = 100
Notentextausgabe in 16tel-Schritten:

PHRASE  TAKT     SOPRAN  ALT   TENOR  BASS     HARMONIK

     1            C#2    A 1    E 1   A -1     T



           1      C#2    A 1    E 1    A 0     T

                  D 2

                  E 2    H 1    E 1    G#0     D3



                  C#2    A 1    E 1    A 0     T



                  A 1    A 1    E 1    F#0     S3

                                D 1

           2      H 1    G 1    D 1    G 0     SS

                         F#1

                  C#2    E 1    A 0    A 0     T



                  D 2    F#1    A 0    D 0     S



     2            C#2    E 1    A 0   A -1     T
```

Figure 2.7: Beginning of file *bch001.txt*, which encodes the chorale *Ach bleib' bei uns, Herr Jesu Christ.*

```
!!!COM: Bach, Johann Sebastian
!!!OPR: Joh. Seb. Bachs vierstimmige Choralgesnge
!!!XEN: Four-part Chorales
!!!OTL: 1. Ach bleib' bei uns, Herr Jesu Christ
!!!OMV: 1
!!!SCT: BWV 253
!!!SCA: Thematisch-systematisches Verzeichnis der musikalischen Werke
Johann Sebastian Bach: Bach-Werke-Verzeichnis (Schmieder)
!!!YOR: Bach Gesellschaft Edition xxxix
!!!EED: Steven Rasmussen
!!!ENC: Steven Rasmussen
!!!CDT: 1685/3//-1750/7/28/
!!!OCY: Deutschland
!!!YEC: Copyright (c) 1994, 2000 Center for Computer Assisted Research
in the Humanities
!!!YEM: Rights to all derivative editions reserved
!!!YEM: Refer to licensing agreement for further details
!!!YEN: United States of America
**kern  **silbe **kern  **kern  **kern
*I:[SOPRANO]    *I:[SOPRANO]    *I:[ALTO]     *I:[TENORE]     *I:[BASSO]
*clefG2 *       *clefG2 *clefGv2        *clefF4
*k[f#c#g#]      *       *k[f#c#g#]      *k[f#c#g#]      *k[f#c#g#]
*A:     *       *A:     *A:     *A:
*M4/4   *       *M4/4   *M4/4   *M4/4
=0-     =0-     =0-     =0-     =0-
4cc#\   Ach     4a/     4e\     4AA/
2ryy    .       2ryy    2ryy    2ryy
4ryy    .       4ryy    4ryy    4ryy
=1      =1      =1      =1      =1
8cc#\L  bleib'  4a/     4e\     4A\
8dd\J   .       .       .       .
4ee\    bei     4b\     4e\     4G#\
4cc#\   uns,    4a/     4e\     4A\
4a/     Herr    4a/     8e\L    4F#\
.       .       .       8d\J    .
=2      =2      =2      =2      =2
4b\     Je-     8gn/L   4d\     4Gn\
.       .       8f#/J   .       .
4cc#\   -su     4e/     4A/     4A\
4dd;\   Christ, 4f#;/   4A;/    4D;\
4cc#\   weil    4e/     4A/     4AA/
```

Figure 2.8: Beginning of file *0253-01.krn*, a **kern encoding of the chorale *Ach bleib' bei uns, Herr Jesu Christ.*

Pearce (2005) extends and generalises the event space in order to represent a wider range of melodies:

$$\begin{aligned}\mathcal{E} = {} & [\texttt{Onset}] \times [\texttt{RestLength}] \times [\texttt{Duration}] \times [\texttt{BarLength}] \\ & \times [\texttt{Pulses}] \times [\texttt{Pitch}] \times [\texttt{KeySig}] \times [\texttt{Mode}] \times [\texttt{Phrase}].\end{aligned}$$

The basic time unit has been reduced from a semiquaver to one ninety-sixth of a semibreve, which allows semiquaver triplets and demisemiquavers. With this in mind, the definitions of types `Onset`, `Duration` and `BarLength` have effectively been changed. The time signatures of the chorale melodies, either $\frac{3}{4}$ or $\frac{4}{4}$, are easily distinguished by bar length; but this is not generally the case. In order to discriminate between, for example, $\frac{3}{4}$ and $\frac{6}{8}$, type `Pulses` (previously `pulses`) has been introduced, which is based on the upper digit of the time signature. Types `Pitch` and `KeySig` are completely unchanged. In the formulation of Conklin and Witten (1995), `RestLength` (previously `deltast`) is a derived type; Pearce (2005), however, makes it a basic type. It captures the notion of a rest, in a rather indirect way, as the time interval between the end of the previous event and the beginning of the current one (if there is no rest, the time interval is zero). Type `Mode` (previously `mode`) captures the distinction between major and minor keys; $[\texttt{Mode}] = \{0, 9\}$, where $[\![0]\!]_{\texttt{Mode}}$ = major, and $[\![9]\!]_{\texttt{Mode}}$ = minor.[4] Finally, type `Phrase` (previously `phrase`) models phrase boundaries, which are important from the point of view of melodic (and more generally, musical) structure; $[\texttt{Phrase}] = \{-1, 0, 1\}$, where $[\![1]\!]_{\texttt{Phrase}}$ = first event in phrase, and $[\![-1]\!]_{\texttt{Phrase}}$ = last event in phrase. This type effectively makes type `Fermata` redundant, since a fermata marks the end of a phrase in the chorale melodies.

Besides the basic types making up the event space, Conklin and Witten (1995) describe fifteen derived types, including four threaded types. The first of the non-threaded derived types, `RestLength`, has been promoted to a basic type by Pearce (2005) as described above. Type `IOI` (inter-onset interval, previously known as `ioi` and `gis221`) is the time period between the start of two consecutive events:

$$\Psi_{\texttt{IOI}}(e_1^j) \quad = \quad \begin{cases} \perp & \text{if } j = 1 \\ \Psi_{\texttt{Onset}}(e_1^j) - \Psi_{\texttt{Onset}}(e_1^{j-1}) & \text{otherwise.} \end{cases}$$

Two definitions of type `PositionInBar` (previously `posinbar`) have been given in previous research. The first is the number of time units after the beginning of the current bar that an event starts (Conklin and Witten, 1995). The second, from Pearce (2005), distinguished here by renaming as `SeqPositionInBar`, is sequential position in bar, where $[\![1]\!]_{\texttt{SeqPositionInBar}}$ = first event, $[\![2]\!]_{\texttt{SeqPositionInBar}}$ = second event, and so on. Boolean test type (Conklin and Anagnostopoulou, 2001) `FirstInBar` (first in bar, previously known as `fib`) distinguishes between events on the first beat of a bar[5] (*i.e.*, those with

[4] Other modes, such as Lydian, can if necessary be represented by other integers.

[5] Pearce (2005) says first event of a bar rather than first beat; but since the use of rests is disallowed

a PositionInBar value of 0) and all other events:

$$\Psi_{\texttt{FirstInBar}}(e_1^j) \quad = \quad \begin{cases} T & \text{if } \Psi_{\texttt{PositionInBar}}(e_1^j) = 0 \\ F & \text{otherwise.} \end{cases}$$

Type Interval (chromatic pitch interval, previously known as cpint and seqint) is the difference in pitch between two consecutive events, measured in semitones:

$$\Psi_{\texttt{Interval}}(e_1^j) \quad = \quad \begin{cases} \perp & \text{if } j = 1 \\ \Psi_{\texttt{Pitch}}(e_1^j) - \Psi_{\texttt{Pitch}}(e_1^{j-1}) & \text{otherwise.} \end{cases}$$

Type Contour (previously contour) indicates whether an event is higher than, lower than, or at the same pitch as the previous event:

$$\Psi_{\texttt{Contour}}(e_1^j) \quad = \quad \begin{cases} -1 & \text{if } \Psi_{\texttt{Pitch}}(e_1^j) < \Psi_{\texttt{Pitch}}(e_1^{j-1}) \\ 0 & \text{if } \Psi_{\texttt{Pitch}}(e_1^j) = \Psi_{\texttt{Pitch}}(e_1^{j-1}) \\ 1 & \text{if } \Psi_{\texttt{Pitch}}(e_1^j) > \Psi_{\texttt{Pitch}}(e_1^{j-1}). \end{cases}$$

In the original work of Conklin (1990), type Tonic (then known as referent) represented the tonic of the major key with a given key signature; this of course meant that for a minor key the referent was the mediant, which is clearly not ideal. Pearce (2005) gives an improved definition of Tonic, which now represents the tonic irrespective of whether the key is major or minor:

$$\Psi_{\texttt{Tonic}}(e_1^j) \quad = \quad \begin{cases} (\Psi_{\texttt{Mode}}(e_1^j) + 7\Psi_{\texttt{KeySig}}(e_1^j)) \bmod 12 & \text{if } \Psi_{\texttt{KeySig}}(e_1^j) > 0 \\ (\Psi_{\texttt{Mode}}(e_1^j) - 5\Psi_{\texttt{KeySig}}(e_1^j)) \bmod 12 & \text{if } \Psi_{\texttt{KeySig}}(e_1^j) < 0 \\ \Psi_{\texttt{Mode}}(e_1^j) & \text{otherwise.} \end{cases}$$

Type ScaleDegree (chromatic pitch interval from the tonic, previously known as cpintfref and intfref) is the pitch interval (mod 12) between an event and the referent:

$$\Psi_{\texttt{ScaleDegree}}(e_1^j) = (\Psi_{\texttt{Pitch}}(e_1^j) - \Psi_{\texttt{Tonic}}(e_1^j)) \bmod 12.$$

As a result of the improved definition of Tonic, Boolean valued type InScale (previously inscale) now distinguishes between events which are in the major or harmonic minor scale (as appropriate) with the referent as tonic, and those which are not. Type IntFirstInBar (previously called IntFirstBar, cpintfib and intfib) is the chromatic pitch interval from the event on the first beat of the current bar; if there is no event on the first beat, IntFirstInBar is undefined. Types IntFirstInPiece (chromatic pitch interval from first event in piece, previously called IntFirstPiece, cpintfip and intfip) and IntFirstInPhrase (chromatic pitch interval from first event in phrase, previously called IntFirstPhrase, cpintfiph and intphbeg) are self-explanatory.

in this research, the two definitions are effectively equivalent.

Pearce (2005) introduces a number of other non-threaded derived types. The first of these, `PitchClass` (chromatic pitch-class, previously known as `cpitch-class`), encapsulates the notion of octave equivalence by enforcing a congruence relation on `Pitch` (Conklin, 1990): $i \equiv j$ iff $(i - j) \bmod 12 = 0$; therefore $[\texttt{PitchClass}] = \{0, 1, \ldots, 11\}$. Type `IntervalClass` (chromatic pitch-class interval, previously known as `cpcint`) is derived by similarly applying a congruence relation to `Interval`. Type `DurRatio` (previously `dur-ratio`) is the ratio of the duration of an event to the duration of the immediately preceding event:

$$\Psi_{\texttt{DurRatio}}(e_1^j) = \begin{cases} \bot & \text{if } j = 1 \\ \frac{\Psi_{\texttt{Duration}}(e_1^j)}{\Psi_{\texttt{Duration}}(e_1^{j-1})} & \text{otherwise.} \end{cases}$$

The *tactus* is the main beat occurring at a shorter time-period than the bar; for example, in music with a time signature of $\frac{3}{4}$, there are three tactus pulses per bar separated by a time interval of a crotchet. Boolean test type `Tactus` (previously `tactus`) indicates whether or not an event occurs on a tactus pulse:[6]

$$\Psi_{\texttt{Tactus}}(e_1^j) = \begin{cases} T & \text{if } \Psi_{\texttt{Onset}}(e_1^j) \bmod \frac{\Psi_{\texttt{BarLength}}(e_1^j)}{\Psi_{\texttt{Pulses}}(e_1^j)} = 0 \\ F & \text{otherwise.} \end{cases}$$

Finally, test types `FirstInPhrase` (first event in phrase, previously known as `fiph`) and `LastInPhrase` (last event in phrase, previously known as `liph`) are trivially derived from type `Phrase`.

The symbol $\ominus$ is introduced here to indicate threading, where A $\ominus$ B means A threaded B. The first of the threaded types is `Interval` $\ominus$ `FirstInBar` (interval threaded at first in bar, previously known as `ThreadBar` and `thrbar`), which is defined only for notes with a `PositionInBar` value of 0. It is a combination of interval with the first note in the previous bar and the difference in start-time of the notes, the *timescale*, bearing in mind that the first note of the preceding bar is not necessarily at its beginning. Similarly, type `Interval` $\ominus$ `FirstInPhrase` (interval threaded at first in phrase, previously known as `ThreadInitPhr`, `thrfiph` and `thrph`) is defined only for the first event in a phrase, and is a combination of chromatic pitch interval with the first note in the previous phrase and its timescale. Pearce (2005) adds type `Interval` $\ominus$ `LastInPhrase` (interval threaded at last in phrase, previously known as `ThreadFinalPhr` and `thrliph`), which requires no further explanation. Type `Interval` $\ominus$ `Tactus` (interval threaded at tactus, previously known as `ThreadTactus` and `thrtactus`) is a generalisation and modification of what was originally called `thrqu` (threaded quarter note); it is defined for notes occurring on tactus beats, and is a combination of chromatic pitch interval with the previous such note and its timescale. Type `PhraseLength` (length of phrase, previously called `phraselength` and `lphrase`) is the number of time units from the start

[6] This test type does not currently indicate the correct tactus pulses for compound time signatures.

of the first note of the phrase to the start of its last note (which has a `LastInPhrase` value of *true*). See Table 2.2 for a summary of basic and derived types, including their syntactic domain $[\tau]$.

2.3.4.2 Chord Representations for Probabilistic Graphical Models

Paiement et al. (2005b, 2006) are primarily interested in modelling jazz music, and describe several chord (and melodic) representations for use in probabilistic graphical models. The first chord representation is designed to facilitate psychoacoustic comparisons between chords, which are viewed as individual timbres (Vassilakis, 1999). For each note in a chord, an approximation of the perceived loudness of individual fundamental and harmonic frequencies (adjusted to well-tempered tuning) is calculated over the frequency range 30 Hz to 20 kHz. To each well-tempered pitch in this range is assigned the maximum loudness[7] at that pitch of the notes in the chord, giving a distributed representation of that chord. This can be simplified by adding the loudnesses for each pitch class (C, C♯, and so on), giving the representation $\mathbf{v}_j = \{v_j(0), \ldots, v_j(11)\}$ for each chord $\mathcal{X}_j$. The Euclidean distance between perceptually similar chords tends to be small in this continuous chord space.

Having introduced the continuous distributed representation, the next step (producing the second chord representation) "is to convert the Euclidean distances between chord representations into probabilities of substitution between chords. Chords can then be represented as individual discrete events" (Paiement et al., 2005a, p. 11). The probability of substituting chord $\mathcal{X}_i$ for $\mathcal{X}_j$ is:

$$p_{i,j} = \frac{\phi_{i,j}}{\sum_{1 \leq j \leq s} \phi_{i,j}}$$

where $\phi_{i,j} = \exp\{-\lambda \|\mathbf{v}_i - \mathbf{v}_j\|^2\}$, s is the number of different chords in the corpus and $0 \leq \lambda < \infty$ (to be optimised).

In order to model interactions between melody and harmony, a means of representing melody is required. Since both the duration and metrical position of a note is related to its perceptual importance, "a 12-dimensional continuous vector representing the relative importance of each pitch class over a given period of time t" (Paiement et al., 2005a, p. 14) is proposed. A bar of time length t can be divided into say eight time-steps of equal length. A pitch class is deemed to have a perceptual importance equal to the number of time-steps during which it is heard in time t modified by a metrical weighting (*e.g.*, in a four-beat bar, the first, second, third and fourth beats are assigned weightings of 5, 3, 4 and 3 respectively, with smaller intermediate weights).

After an intermediate step where root progressions are modelled given the melody, we arrive at the final model for modelling melody and harmony, for which a third chord representation is introduced. Here, a chord comprises a root component (equal to the

[7] The loudnesses are not added to allow for the masking effect (Moore, 1982).

τ	$[\![\cdot]\!]_\tau$	$[\tau]$	Derived from
Onset	start-time of note	Z^*	Onset
RestLength	duration of rest	Z^*	RestLength
Duration	duration of note	Z^+	Duration
BarLength	number of time units in a bar	Z^*	BarLength
Pulses	upper digit of time signature	Z^*	Pulses
Pitch	chromatic pitch	Z	Pitch
KeySig	number of flats or sharps	$\{-7,\ldots,7\}$	KeySig
Mode	major or minor key	$\{0,9\}$	Mode
Phrase	note at start or end of phrase	$\{-1,0,1\}$	Phrase
IOI	difference in start-time	Z^+	Onset
PositionInBar	position of note in the bar	Z^*	Onset
SeqPositionInBar	sequential position in the bar	Z^+	Onset
Interval	sequential pitch interval	Z	Pitch
Contour	descending, level, ascending	$\{-1,0,1\}$	Pitch
Tonic	tonic of relevant scale	$\{0,\ldots,11\}$	KeySig, Mode
ScaleDegree	Interval from referent	$\{0,\ldots,11\}$	Pitch
InScale	note in relevant scale, or not	$\{T,F\}$	Pitch
IntFirstInBar	Interval from first in bar	[Interval]	Pitch
IntFirstInPiece	Interval from first in piece	[Interval]	Pitch
IntFirstInPhrase	Interval from first in phrase	[Interval]	Pitch
PitchClass	chromatic pitch-class	$\{0,\ldots,11\}$	Pitch
IntervalClass	sequential pitch-class interval	$\{0,\ldots,11\}$	Pitch
DurRatio	sequential duration ratio	Q^+	Duration
FirstInBar	note on first beat of bar, or not	$\{T,F\}$	Onset
Tactus	note on tactus pulse, or not	$\{T,F\}$	Onset
FirstInPhrase	first note in phrase, or not	$\{T,F\}$	Phrase
LastInPhrase	last note in phrase, or not	$\{T,F\}$	Phrase
Interval $\ominus$ FirstInBar	Interval at first in bar	[Interval] $\times Z^+$	Pitch, Onset
Interval $\ominus$ FirstInPhrase	Interval at first in phrase	[Interval] $\times Z^+$	Pitch, Onset
Interval $\ominus$ LastInPhrase	Interval at last in phrase	[Interval] $\times Z^+$	Pitch, Onset
Interval $\ominus$ Tactus	Interval at tactus beats	[Interval] $\times Z^+$	Pitch, Onset
PhraseLength	length of phrase	Z^+	Phrase, Onset

Table 2.2: Basic and derived types for melodies (adapted from Conklin and Witten, 1995, with modifications and additions from Pearce, 2005). Note that $Z^+ = \{1,2,3,\ldots\}$, $Z^* = \{0,1,2,\ldots\}$, $Z = \{\ldots,-2,-1,0,1,2,\ldots\}$ and $Q^+ = \{\text{positive rational numbers}\}$.

pitch class of the root) and six structural components (3rd, 5th, 7th, 9th, 11th and 13th) which have up to four possible values, depending on whether a component is major, minor, perfect, diminished, augmented, and so on.

2.3.4.3 Other Data Structures

The *Common Hierarchical Abstract Representation for Music*, or *CHARM* (Smaill et al., 1993), is a logical, flexible representation scheme for musical structure. It describes performed music in terms of discrete notes and higher-level structures. A note, or *event*, is represented by the following tuple:

$$\texttt{event}(\texttt{Identifier}, \texttt{Pitch}, \texttt{Time}, \texttt{Duration}, \texttt{Amplitude}, \texttt{Timbre})$$

while higher-level structure is represented by *constituents*, defined as:

$$\texttt{constituent}(\texttt{Identifier}, \texttt{Properties}, \texttt{Definition}, \texttt{Particle\_list}, \texttt{Description})$$

where `Particle_list` is a list of events and/or lower-level constituents.

Ponsford et al. (1999) use a representation which is neutral with respect to key. Pitch is defined as the tuple (scale degree, modifier), where scale degree $\in \{1, 2, 3, 4, 5, 6, 7\}$ and modifier $\in \{\texttt{common}, \texttt{major}, \texttt{minor}, \texttt{raised}, \texttt{lowered}\}$. An event is defined, in a CHARM compliant way, as:

$$\texttt{event}(\texttt{NoteID}, \texttt{Pitch}, \texttt{OnsetTime}, \texttt{Duration}).$$

For the higher-level structure representing harmonies, however, a simpler representation than CHARM is employed, exemplified by $((2, \texttt{common})(5, \texttt{common})(7, \texttt{minor}))$.

As the corpus is initially in the form of MIDI files, a certain amount of pre-processing must be done in order to make it usable. First of all, each piece is converted to a list of events, as defined above. After this, pitches are converted to scale degree; note lengths are rounded; harmony is sampled at quaver intervals; chords are converted to root position form; and identical quaver-length harmonies are elided to produce crotchet length harmonies. In addition, the corpus is automatically annotated with symbols marking boundaries of pieces, phrases and bars.

In order to represent both the distribution of notes within a chord and its harmonic function, Allan and Williams (2005) use symbols such as "0:4:9:16/T." The figures refer to the number of semitones that each voice (soprano, alto, tenor and bass) is lower than the soprano. In this example, "T" means that it is a tonic chord. The representation distinguishes between notes continued from the previous beat and repeated notes. Movement off the beat in the lower three voices is represented by symbols such as "0,0,2,2/0,0,2,2/0,0,0,0" (see Figure 2.9). The three groups of figures describe movement in the alto, tenor and bass parts respectively, relative to the note sounding at the

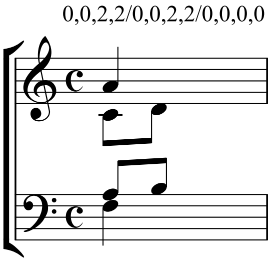


Figure 2.9: Hidden state representation for movement off the beat (adapted from Allan and Williams, 2005).

beginning of the beat. Each group contains four figures, one for each quarter of a beat. The first figure, corresponding to the start of the beat, is always zero. In this example, no movement occurs until halfway through the beat, when both the alto and the tenor are raised by two semitones. Since there is no further movement, the final figure in the alto and tenor groups is also two.

2.4 The Evaluation of Computational Models of Music

One way of evaluating a computational model of music is to evaluate compositions generated by it. Pearce and Wiggins (2001) address this specific issue, believing "a common means of evaluation to be fundamental if we are to judge musical theories from other communities in our research programme." Their framework allows music to be composed by any computational means, and enables objective evaluation of this music with respect to specified compositional aims. There are four distinct parts to the framework. Firstly, the compositional aims must be unambiguously specified. Secondly, a model of a musical style or genre, known in this context as a *critic*, is created from a relevant corpus of musical examples using a justifiably suitable machine learning technique. Thirdly, the computer produces compositions which are acceptable to the critic. Finally, the compositions are evaluated with reference to the compositional aims by means of carefully designed experiments involving human subjects.

Pearce (2005) describes a method for the evaluation of creativity, which fits nicely into the above framework, called the *consensual assessment technique*, or *CAT* (Amabile, 1996). The end-products of the creative task, in this case pieces of music, must be suitable for rating by expert judges, and should be presented to each judge in a different random order. Each judge independently provides subjective ratings of various aspects of the compositions, such as originality. A statistical assessment of inter-judge rating consistency is then carried out; providing the consistency is high, correlation of ratings with, for example, objective features of the compositions is permissible.

Conklin and Witten (1995) develop an evaluation technique based on information theory. If we define $P_m(e_i|c_{i,m})$ as the probability of the i^{th} musical event given its context for a particular model m, then the minimum length of the compressed code is

$$-\log_2 P_m(e_i|c_{i,m}) \tag{2.1}$$

bits. This is known as the *pointwise entropy*, which can be thought of as a measure of surprise (Manning and Schütze, 1999); the greater the pointwise entropy, the greater the surprise. Assuming that there are a total of n sequential events, then an approximation to *cross-entropy* is given by

$$-\frac{1}{n}\sum_{i=1}^{n}\log_2 P_m(e_i|c_{i,m}). \tag{2.2}$$

This is a useful measure, because it gives a "per symbol" value; it can therefore be used to compare sequences of any length. All else being equal, the model which assigns the lowest cross-entropy to a test sequence is the best descriptor of the data (Allan, 2002). This is equivalent to saying that the model which assigns the highest geometric mean of the conditional probabilities in a test sequence describes the data best.

2.5 The Computational Modelling of Music

2.5.1 Summary of Previous Work

2.5.1.1 Constraint-based Methods

Ebcioğlu (1988) describes a rule-based expert system, called CHORAL, for the four-part harmonisation of chorale melodies in the style of J. S. Bach. A substantial amount of complex musical knowledge is represented using first-order logic. The predicates are grouped in such a way that the music is observed from *multiple viewpoints*;[8] each of these viewpoints is described in terms of a rich set of logical primitives, covering many musical attributes. Amongst the viewpoints are one for the chord skeleton, and one which observes the individual voices.

Each viewpoint is represented by a *solution array*, which is filled step by step in co-ordination with the other viewpoints by using *generate-and-test*. Candidate assignments to the next element of the solution array are generated by means of *production rules*, and then candidates not meeting all of the *constraints* are discarded. The *worth* of each of the remaining candidates is calculated by adding the weights of applicable *heuristics*, which are weighted according to their importance; the candidate with the highest worth is added to the solution array. A ranked list of alternative candidates is retained, however, in case at some future point in the process it is not possible to make an assignment to a solution array; in this eventuality *backtracking* to an earlier step occurs, and the highest

[8] The multiple viewpoints here are different from those introduced by Conklin and Cleary (1988), not least because no statistical modelling is involved, although they seem to be the inspiration for their terminology.

ranked of the remaining candidates for this step replaces the originally chosen element.

The *chord skeleton* viewpoint produces a sequence of chords with no duration information. Associated with each chord is the key (which can change during the chorale) and the scale degree. Attributes such as the pitch of a voice in any chord in the generated sequence can be referenced. Rules governing for example modulation and cadences are encoded in this viewpoint. The *fill-in* viewpoint chooses the actual notes (based upon the chord skeleton), including passing notes, suspensions and so on. The rules deliberately allow clashes of passing notes, since these are a part of Bach's harmonic style. The *time-slice* viewpoint represents the chorale as a sequence of quaver-length time-slices. Attributes such as the pitch of a voice at any time-slice, and whether or not there is a note onset, can be referenced. Harmonic constraints such as the avoidance of consecutive fifths are found in this viewpoint. The *melodic string* and *merged melodic string* viewpoints observe the melodic aspects of the voices; the latter is similar to the former except that repeated notes are merged. Voice leading rules are encoded in this viewpoint. Finally, there is a *Schenkerian analysis* viewpoint (inspired by Schenker, 1979 and Lerdahl and Jackendoff, 1983), which produces a hierarchical voice-leading analysis; this viewpoint does not contribute to the creation of the harmony, however.

Ebcioğlu (1988) assesses the competence of the expert system as approaching that of a talented music student who has studied the chorales harmonised by Bach, but concedes that this method is probably not a good cognitive model of composition.

Ovans and Davison (1992) describe an expert assistant for first species counterpoint which makes use of constraint satisfaction techniques. The interactive system displays a melody, and then determines the set of allowable counterpoint notes for each melody note. The user, who might typically be a music student, composes a counterpoint to the given melody by making choices from the sets of notes (not necessarily in chronological order). Such choices generally further constrain subsequent choices. At any time, at the request of the user, the expert assistant can complete the composition.

Notes of equal length are used in first species counterpoint, requiring only pitch to be represented; constraints are expressed in terms of the chosen representation (MIDI, in this case). Nine rules govern valid note combinations (Mann, 1965), each of which comprises one or more constraints. Ovans and Davison (1992) note that although the system prevents invalid solutions by efficiently representing and propagating constraints, it is not as proficient as the user at finding good (*i.e.*, musical) solutions.

Pachet and Roy (1998) define the automatic harmonisation problem as the automatic harmonisation (typically in four parts) of a given melody, such that the rules of harmony are obeyed. Stated in these terms, a constraint-based solution to this problem seems quite natural. They describe three categories of harmonic constraint: constraints on successive notes of a part, constraints on the constituent notes of a chord, and constraints on sequences of chords, which are identified by the scale degree of their root and their inversion. Their focus is on the exploitation of part-whole relations in the formulation of this problem, for example the relationship between note and chord. They have used object-oriented techniques to build a structured representation of music, called the MusES system (Pachet et al., 1996), with which knowledge of harmony can be expressed.

Comparisons with previous attempts to solve this problem using flat representations are difficult, because of the widely differing processor speeds, memory sizes and general architectures of the computers running the various implementations. Pachet and Roy (1998) have, nevertheless, concluded that all else being equal, their system is the fastest. Irrespective of all this, it is clear that the primary advantage of such a structured approach is that it makes the process of building the knowledge base much faster and easier.

Phon-Amnuaisuk and Wiggins (1999) stress the need for separation of the knowledge base from the inference engine in such systems, since this facilitates extendibility and maintainability. They also describe the organisation of knowledge in terms of *object-level knowledge*, which is a lower level of knowledge; *meta-level knowledge*, which is a higher level of knowledge used to reason about the object-level; *domain knowledge*, which is knowledge of a particular area such as music; and *control knowledge*, which enables the system to utilise the domain knowledge. Their rule-based system for four-part harmonisation employs a structured approach. Cadences are dealt with first, followed by pre-cadence sections, mid-phrase sections and finally phrase beginnings. Backtracking is used to retreat from dead-ends in the search.

The issue under investigation here is the relative performance of this system compared with a system implementing a *genetic algorithm* (see "Evolutionary Methods" below) into which the same amount of musical knowledge has been encoded. An objective evaluation of the harmonisations clearly demonstrates the superiority of the rule-based approach. The reason given for this superiority is additional implicit knowledge in the form of a structured search mechanism (cadences first, and so on). Phon-Amnuaisuk and Wiggins (1999) suggest improving on this by means of what they call the *explicitly structured knowledge paradigm*, in which musical domain knowledge is split into *musical knowledge* and *musical processes.* Both of these dimensions would have a hierarchical structure (*cf.* Pachet and Roy, 1998). See Figure 2.10 for an example of a musical process.

2.5.1.2 Evolutionary Methods

Phon-Amnuaisuk et al. (1999) use a genetic algorithm to generate homophonic[9] four-part harmony for given melodies. Musical knowledge resides in three different areas of the system. Firstly, there is a knowledge-rich CHARM compliant representation (see §2.3.4.3) for the chromosomes, each of which consists of five vertically aligned sequences. The first four of these contain information about the four parts (SATB). Within these sequences, each note is represented by a tuple of integers, such as $[0, 0, 3]$, which rep-

[9] Consisting of block chords only, with no extra-chord movement such as passing notes.

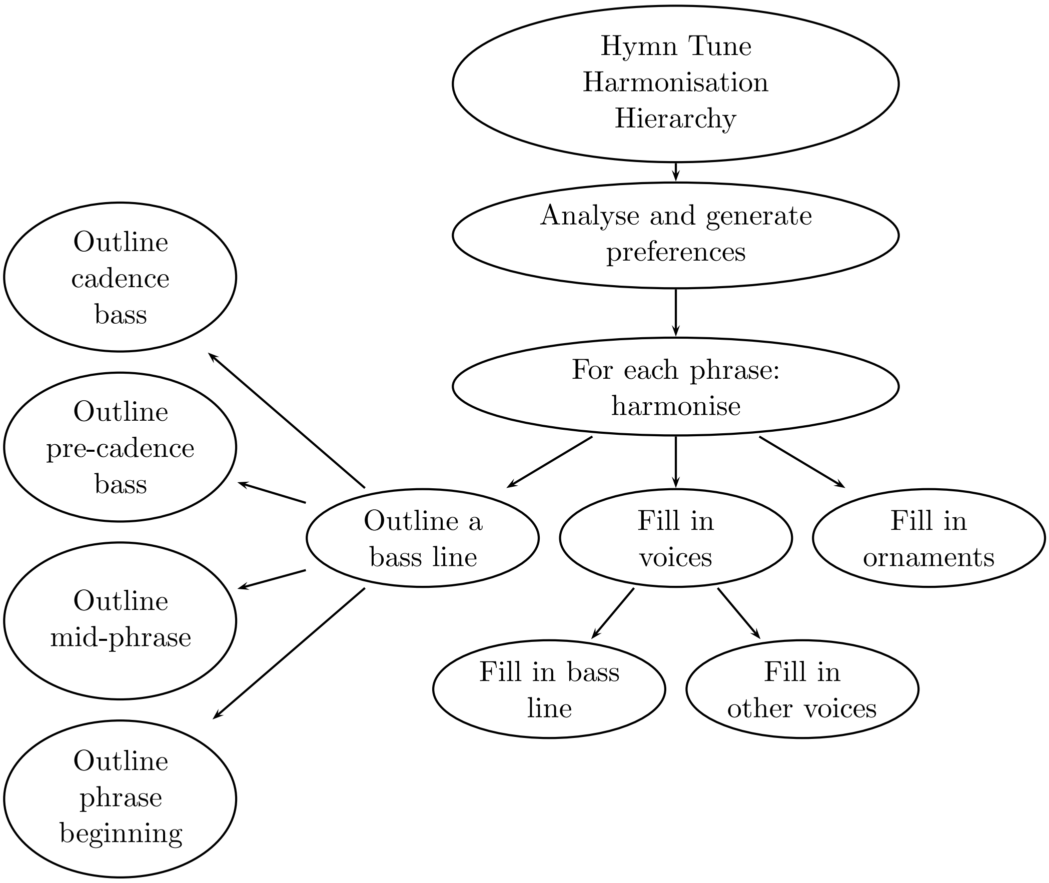


Figure 2.10: The hierarchical structure of a harmonisation process (adapted from Phon-Amnuaisuk and Wiggins, 1999).

resent scale degree, chromatic alteration and octave respectively. The fifth sequence contains integers representing note duration. Secondly, there is musical knowledge in the six crossover and mutation operators. These are: *Splice*, which results in two chromosomes undergoing a crossover; *Perturb*, a mutation of the three lowest parts (ATB) such that one or more parts are transposed by up to a tone; *Swap*, a mutation effected by swapping two of the three lowest parts; *Rechord*, a mutation resulting in a different harmonisation of a particular soprano note; *PhraseStart*, a mutation of the first down beat of each phrase, resulting in tonic root position (I) chords; and *PhraseEnd*, a mutation which results in a root position chord appearing at the end of each phrase. Thirdly, musical knowledge is incorporated into the fitness function; chromosomes are penalised for violating a range of harmonic constraints, such as the avoidance of parallel octaves and fifths.

The results show, however, that even after 300 generations it is not possible to satisfy all of the constraints. This is explained in terms of a solution converging to a local optimum in the search space which is globally sub-optimal. The context dependent nature of harmony requires that in order to escape from such a sub-optimal solution, many simultaneous changes need to be made to reduce the fitness penalty; this is unlikely to happen. The system is also unable to produce a coherent overall harmonic structure.

A genetic algorithm to generate instrumental jazz solos has also been developed, with encouraging results considering the simplicity of the encoded rules (Wiggins et al., 1999).

2.5.1.3 Connectionist Methods

Hild et al. (1992) have created a neural network system called HARMONET, which is capable of producing four-part harmonisations of chorale melodies in the style of J. S. Bach. It was separately trained, using error backpropagation, on two sets of twenty Bach chorale harmonisations in major and minor keys respectively. A key feature of the architecture of this system is that the overall harmonisation task is decomposed into a number of subtasks.

The first and most important subtask is the construction of the harmonic skeleton, which comprises a chord symbol for each crotchet beat (quavers and semiquavers are viewed as ornamentation, and are dealt with in a later subtask). The chord symbols encompass attributes such as "chord inversion" and "characteristic dissonances." Individual harmonies are learned in relation to a fixed local context, or *window*, consisting of previous harmonic symbols; past, current and future melodic (or soprano) symbols; and symbols phr_t and str_t, which indicate position in phrase and stressed/not stressed respectively. The window, which moves a crotchet at a time through a chorale, is shown below (H_t is the symbol to be learned or generated):

$$\begin{array}{ccccc} & & s_{t-1} & s_t & s_{t+1} \\ H_{t-3} & H_{t-2} & H_{t-1} & H_t & \\ & & & phr_t & \\ & & & str_t. & \end{array}$$

In a more advanced system, three networks trained on different sized windows each generate a basic harmonic function at each crotchet beat. The harmonic function with most "votes" goes forward for modification by two further networks dealing with, for example, chord inversion and characteristic dissonances.

The second subtask, performed by conventional symbolic algorithms, is the creation of the chord skeleton, which is the filling in of the actual notes of each chord to produce (at this stage) homophonic harmony. This is done by generating a set of chords which satisfy the usual constraints (or rules) of harmony, and then testing the chords by a number of criteria in order to choose the best one.

The third and final subtask is the addition of ornamenting quavers (*e.g.*, passing notes). A network is trained using contexts described in terms of, for example, the various intervals between the notes of adjacent chords (*cf.* the use of derived types in the multiple viewpoint framework). The resulting generated harmonisations were evaluated by an audience of professional musicians, who gauged the quality to be similar to that produced by an improvising organist.

It is possible to change the style from Bach's to that of any other composer by using an appropriate training set, which makes this system far more flexible than a purely knowledge-based approach. The style of one composer might be more or less complex than that of another, however. Finding the appropriate level of complexity of the networks is a problem that has been considered by Hörnel and Ragg (1996). There are three specific issues: the degrees of freedom of the model, the size of the network, and input vector information redundancy. All of these issues are addressed by ENZO, a system implementing a genetic algorithm for network optimisation. Its fitness function can contain problem-specific (in this case musical) knowledge.

The classification performance of evolved HARMONET networks, based on a test set of ten major key Bach chorales, is slightly better than that of networks given standard training. The real benefit, however, is that the evolved networks are much smaller than the original ones. It is noticeable that evolved networks trained on J. S. Bach examples are larger and more complex than those trained on J. Pachelbel examples. Some input units representing essential elements of Bach's style are missing from the evolved Pachelbel networks.

2.5.1.4 Multi-agent Methods

Dixon (2000) describes a beat tracking system involving multiple competing *agents*. The input to the system is music in a digital audio format, from which note onset times (taken to be equivalent to musical *events*) are detected and recorded. Analysis of *inter-onset intervals* (IOI is here defined not only as the interval of time between successive onsets, but also as the time between non-adjacent onsets) within short time periods leads to the emergence of an estimate of the *inter-beat interval* (IBI), which is equivalent to the tempo of the music.

Having estimated the IBI, the actual beat locations are determined by multiple agents, each examining a different hypothesis with respect to beat frequency and phase. The hypothesis of an individual agent may change over time. Agents may be added or removed during their progress through the music; for example if an agent is unsure as to whether an event is on the beat or not, it is cloned. One agent accepts that the event is on the beat, and the other does not. On the other hand, if more than one agent is concurrently adhering to the same hypothesis, all but the one with the best beat prediction history are removed; an agent is also deleted if it has been unable to predict a beat location for a specified length of time. A confidence value for each agent (adjusted for tempo) is updated periodically on the basis of its beat prediction history; desirable features include regular spacing of beats and as few gaps in the sequence as possible. At the end of the process, the beat prediction history of the agent with the highest confidence value is chosen to be the output of the system.

The performance of the system is described thus: "The system successfully tracks the beat in most popular music, but makes some phase errors, mainly when presented with extremely complex rhythms or music with large tempo deviations. Even in these situations, the performance is quite robust, with the system recovering from its errors and resuming correct tracking after a short period" (Dixon, 2000, p. 787).

A similar system, but this time including musical knowledge and taking MIDI performance data as its input, is also described (Dixon and Cambouropoulos, 2000). It is noted that although previous systems perform well if the tempo is constant, they perform less well when the tempo varies. The intention is to show that the use of musical knowledge to judge the salience of musical events results in an improved performance when applied to music of varying tempo. Three attributes are used to determine relative salience: duration, dynamics and pitch. These attributes are combined in two different ways so that both an additive and a multiplicative salience function can be tested. Agents' confidence values are updated periodically by adding the adjusted salience values of all the events in their beat prediction history.

The results, for a set of thirteen Mozart piano sonatas played by a professional pianist, show that, although it is debatable as to which of the salience functions is better, the inclusion of low-level musical knowledge does indeed significantly improve the performance of the beat tracking system.

2.5.2 The Statistical Modelling of Melody

2.5.2.1 Finite Context Grammars

Markov Models Pinkerton (1956) creates 1st-order Markov models from a corpus of thirty-nine nursery rhyme melodies (transposed into C major), and uses them to generate new melodies. The melodies are divided into beats, which means that longer notes are split for modelling purposes. His alphabet comprises one symbol per degree of the major scale, plus one for a tied note or rest.[10] Metre is taken into account by creating transition tables for every beat of the bar.

Brooks Jr. et al. (1993) create Markov models of up to 7th-order[11] from a corpus of thirty-seven common metre hymn tunes transposed to the key of C. To avoid the added complexity of note durations, the melodies are split into quavers (the shortest note length in the corpus). This necessitates a means of distinguishing between the start of a note and its continuation: note starts are assigned chromatic pitches represented by the even numbers 02 to 98, while note continuations are assigned the odd numbers 03 to 99 (see also the penultimate paragraph of §2.5.3.2 and §3.4.4). An arbitrary number of rests (represented by 00) are encoded at the beginning of each hymn tune so that the same N-gram length can be used throughout the entire melody.

During the generation (or synthesis) of melodies, external constraints are imposed

[10] It is slightly odd that Pinkerton (1956) decides not to distinguish between tied notes and rests.

[11] Brooks Jr. et al. (1993) say 8th-order, but define order as N-gram length rather than context length.

to enforce a preselected metrical structure as well as ensuring that the final note is a C; a note not meeting such a constraint is discarded, and another generated in its place. It is sometimes necessary to abandon a synthesis and start again. Melodies are generated using unsmoothed models from 0^{th}- to 7^{th}-order inclusive. At very low orders (particularly 0^{th}-order), the melodies are not in the style of those in the corpus, having unnatural intervals; whereas at very high orders, long sequences from the corpus are reproduced. Intermediate orders produce the best results, in which melodies are more or less in the correct style, but are not recognisably composed of sequences from individual hymns in the corpus. There is a tendency to generate long ascending or descending sequences, however, which is not typical of the corpus; and the range of the melodies is unnaturally large due to the transposition of the corpus into C.

Hierarchical Hidden Markov Models Weiland et al. (2005) assert that hierarchical hidden Markov models (Fine et al., 1998) are good at capturing both local and large-scale dependencies in a sequence, whereas hidden Markov models are only capable of efficiently representing local dependencies. Bearing in mind that a piece of music can be analysed as a hierarchical structure (Lerdahl and Jackendoff, 1983; Schenker, 1979), their use of a hierarchical hidden Markov model (HHMM) to learn pitch and phrasing information from a corpus of twenty-five major key chorale melodies (Bach, 1998) seems reasonable. The melodies are transposed to C major, and note lengths are ignored. The training set comprises symbols for phrase beginning and end, end of chorale, and pitch (`C`, `C#`, `D`, and so on). These symbols also form the set of possible observation symbols for the HHMM.

A three-level MaxSR HHMM is used for this purpose. On the top level, as always, is the internal state *root*. At the phrase level, chorales are modelled as having an opening phrase, a final phrase, and at least one other phrase in between. The second level, therefore, comprises three internal states *start*, *body* (which has a self-referential loop) and *end*, and one production state *eof* (which emits the end of chorale symbol). On the third level, associated with each of the internal states on the level above, are one production state per observation symbol (except the end of chorale symbol) and end state *e*; therefore all emission probabilities are equal to 1. Training of the model determines the level three horizontal transition probabilities, resulting in a separate (and different) statistical model for each of the three second-level internal states.

Generated melodies show the same mostly stepwise motion as the original chorale melodies. The model is also trained on bass lines of Bach chorale harmonisations, resulting in generated output containing more leaps of a fourth or a fifth than in the generated melodies. It should be noted, however, that this is an extremely simple implementation, which is equivalent to three first-order N-gram models being used in a procedural way; it should be easy to improve on this without resorting to the complexity of hierarchical hidden Markov models.

2.5.2.2 Multiple Viewpoint Systems

Early Models Conklin and Cleary (1988) argue that the complexity of music makes it necessary to consider more than one sequence of symbols. They describe an early version of the multiple viewpoint framework. The viewpoints applied to the modelling of melody (in this case, monodic Gregorian chant) are:[12] `melody` (the absolute pitch of a note); `melint` (the interval between the preceding note and the current one, for example a major third); `pitchclass` (the notes A, B, C, *etc.*, irrespective of octave); and `duration` (the length of a note). Other viewpoints, for example for keeping track of themes or motifs, can also be envisaged. It is assumed that music comprises discrete events (notes and rests), each described by a combination of the primitives *pitch* and *duration*; a melody is a sequence of such events. Viewpoints are derived by mapping primitives in the training data (a single Gregorian chant) to viewpoint elements within viewpoint chains. When generating a new melody using the models created from these viewpoints, the individual viewpoint predictions are generally combined in the following way: the predictions of each viewpoint are converted into (pitch, duration) pairs, and then the probabilities of corresponding predictions are added (and presumably normalised), resulting in an overall probability for all possible (pitch, duration) pairs. An exception to this is that only the intersection of the `melint` and `pitchclass` prediction sets are used, as `melint` often predicts pitches not found in the corpus; this issue is discussed in detail in §4.2.1. Conklin and Cleary (1988) have also applied multiple viewpoints to the modelling of polyphony (see §2.5.3).

The multiple viewpoint system {`melody`, `duration`} constrains pitches to be chosen from amongst those in the training data. Variable order Markov models with $\hbar = 8$ produce music which is largely identical to sections of the modelled chant; $\hbar = 2$ results in a much more original melody, while maintaining the smoothness of the chant; and $\hbar = 0$ produces a random melody. System {`melint`, `duration`} enables pitches not found in the training data to be generated. $\hbar = 0$ produces a melody which includes many notes not in the mode of the modelled chant (*viz.* Dorian). In order to generate melodies from $\hbar = 0$ models which stay within the mode, `melint` and `pitchclass` are combined as described above.

More Highly Developed Models Conklin (1990) develops the above by describing a method, based on multiple viewpoints (see §2.2.4 and §2.3.4.1), which both constructs an overall theory for a genre by means of a long-term model, and captures structure within a particular work by using a short-term model. The probability distributions of the two models are weighted and then combined to produce an overall distribution. The system uses Prediction by Partial Match with escape method B (Witten and Bell, 1989), and also utilises a considerably extended version of the multiple viewpoint idea

[12] Viewpoint names are shown in typewriter font for consistency with preceding and following discussions involving viewpoints.

(*cf.* Conklin and Cleary, 1988). Viewpoint names below are based upon those found in Pearce and Wiggins (2006).

Using a training corpus of ninety-five chorale melodies and a test set of five, Conklin (1990) demonstrated experimentally that a long-term model using a single linked viewpoint comprising all of the basic types (except `Onset`, which was replaced by `RestLength`) was a poor predictor, although prediction was better with PPM ($\hbar = 2$) than without. A multiple viewpoint system comprising sixteen primitive and threaded viewpoints ($b = 0$ and $\hbar = 6$) fared a little better when using either a short-term or a long-term model; combining them such that the long-term model received four times the weighting further improved prediction. Giving b the value 64 for weighting both within and between the two models resulted in another improvement, as did the use of a multiple viewpoint system consisting largely of linked viewpoints (each containing two primitive viewpoints). The best performing system that was tried comprised six linked viewpoints and six primitive ones, with $b = 128$ and $\hbar = 6$.

Later, Conklin and Witten (1995) did some studies with a small training set which resulted in the fixing of finite context models at 2nd-order for short-term, and 3rd-order for long-term viewpoints. Eight different multiple viewpoint systems were then trained on a set of ninety-five chorale melodies, and tested using a further five such melodies (*i.e.*, the same data as used by Conklin 1990). The experiments focused on the prediction of the basic type `Pitch`. The systems were evaluated by calculating their cross-entropies (see §2.4), which ranged from 2.33 bits/pitch for system {`Pitch`} to 1.87 bits/pitch for system {`ScaleDegree`⊗`Interval`, `Interval`⊗`IOI`, `Pitch`, `ScaleDegree`⊗`FirstInBar`}. The latter system is therefore the best performing. It is interesting to note that people appear to perform only a little better at this task, with a cross-entropy of about 1.75 bits/pitch (Witten et al., 1994). This estimate of human performance must be treated with some caution, however, as it is based on only two chorale melodies.

Conklin and Witten (1995) have demonstrated the utility of presenting prediction results as a plot of pointwise entropy (see Equation 2.1) versus event number. In such an entropy profile, the peaks correspond to surprising events which are therefore hard to predict, while the troughs indicate particularly predictable events. This technique has been used in musical analysis (Potter et al., 2007).

Finally, the system generated some new melodies by means of random sampling, having been given the opening seven notes (as initial context) and the overall rhythmic structure of a number of existing chorale melodies. One of these was presented in the paper, and pronounced “reasonable;” but there was no objective evaluation of the generated melodies.

Conklin (2003) warns that while random sampling is the simplest generation method, it does not necessarily produce music with a sufficiently high probability to be considered typical of the style modelled. He describes two ways of modifying a piece p (preferably one which is already in the required style) in order to generate a new high probability piece. The first technique employs *Gibbs sampling* from the statistical model. An event is chosen at random, and all changes to that event which are permissible by the model are made, resulting in a set of slightly different pieces. One of these pieces is randomly chosen, one of its events is changed, and so on. The second technique uses *Metropolis sampling*. An event in p is chosen at random and replaced by another event, giving rise to p'. If the probability of p' is greater than that of p, it automatically replaces p; if not, there is a chance that p' will not replace p (replacement becomes less likely as the algorithm proceeds). The "winning" piece has one of its events changed at random, and so on.

Pattern Discovery Conklin and Anagnostopoulou (2001) used the multiple viewpoint representation for the discovery of patterns in a corpus of one hundred and eighty-five Bach chorales. A *viewpoint pattern* is a sequence of viewpoint elements of the same type; its *piece count* is the number of pieces in which it occurs, and its *total count*, or $\#(P)$, is the number of times that it appears in the corpus. The probability of the pattern is predicted by a Markov model, trained on the corpus, with $\hbar = 1$. Its expected count, $E(P)$, is found by multiplying its probability by the number of positions in the corpus at which it could possibly appear. A high *pattern score* could indicate something of musicological interest:

$$\frac{(\#(P) - E(P))^2}{E(P)}.$$

The set of viewpoint patterns occurring in a minimum of k pieces is enumerated (the viewpoint and integer k are chosen by the musicologist). The statistical significance of the patterns is tested by deriving a p-value from each pattern score; patterns with p-values above a certain threshold (say 0.01) are discarded. It is possible for significant patterns to be subsequences of longer significant patterns; only the longest such patterns are retained. For viewpoint `Interval` in the soprano, two-hundred and seventy-five longest significant patterns were found for $k = 2$, dropping to six for $k = 100$. The mean length of these patterns was greater at lower values of k. One of the interesting patterns discovered was a six note chromatically ascending figure occurring a total of nineteen times in the bass part of twelve chorales.

Conklin (2003) sees utility in combining Gibbs or Metropolis sampling with pattern discovery. The idea is to find repeated patterns in the piece of music being used as a basis for the generation of a new piece, so that as sampling proceeds, changes made to one pattern can also be made to its clones. This is likely to result in a generated piece with greater structural integrity than might otherwise be the case.

Recent Highly Developed Models Pearce (2005) developed the multiple viewpoint framework in his search for statistical models which best predict unseen melodies, which are capable of generating stylistically successful melodies, and which in particular are able to account for observed patterns of melodic expectation in people. A PPM model was implemented in such a way that different features of the model could be compared in a series of experiments. These features were: long- and short-term models; a number of different escape methods; order bound (including the unbounded PPM*); update exclusion; and interpolated smoothing. The viewpoint system employed for these experiments comprised {`Pitch`} only. The empirically best model, chosen as the basis for further experiments, combined, using a weighted arithmetic mean, a long-term model (LTM) and a short-term model (STM). The LTM was updated with new data following each event prediction; it used escape method C, unbounded order and interpolated smoothing. The STM employed escape method AX, unbounded order, update exclusion and interpolated smoothing. This model predicted the chorale melody data set (see below) with a cross-entropy of 2.34 bits/symbol. It should be noted that because of the nature of the experimental methodology, this model might not be globally optimal (*i.e.*, not all of the possible LTM and STM parameter combinations were tried).

The corpus for the next set of experiments contained 185 chorale melodies. Cross-entropy, determined by ten-fold cross-validation, was the measure used for evaluation. Having found Dempster-Shafer (Garvey et al., 1981) and rank-based viewpoint combination methods to perform relatively poorly in preliminary studies, the first experiment was designed to compare weighted arithmetic with weighted geometric combination (see §2.2.4) both within and between the short- and long-term models. Different values of bias b, drawn from the set $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 16, 32\}$, were also compared. The multiple viewpoint system used was {`ScaleDegree` ⊗ `Interval`, `Interval` ⊗ `IOI`, `Pitch`, `ScaleDegree` ⊗ `FirstInBar`}, modelling basic type `Pitch`, which was the best system found by Conklin and Witten (1995). Optimum performance (*i.e.*, lowest cross-entropy of 2.04 bits/symbol) was produced by employing a weighted geometric mean for all combinations, with a bias of two for viewpoint combination and a bias of seven for combining the LTM with the STM. The second experiment was designed to identify multiple viewpoint systems modelling basic type `Pitch` which further reduce cross-entropy. The viewpoint selection algorithm described in §2.2.4 was used to construct such a system, on the basis of minimum cross-entropy, from a pool of primitive viewpoints (see Table 2.2) and a limited number of linked viewpoints derived from previous research (see Table 2.3). The system that emerged was {`Interval`⊗`Duration`, `ScaleDegree`⊗`IntFirstInPiece`, `Pitch` ⊗ `Duration`, `ScaleDegree` ⊗ `FirstInBar`, `Interval` ⊖ `Tactus`, `ScaleDegree` ⊗ `Duration`, `Interval` ⊗ `DurRatio`, `IntFirstInPiece`, `Interval` ⊖ `FirstInPhrase`}, with a cross-entropy of 1.95 bits/symbol.

Pearce (2005) argues that it is not necessary to include Gestalt-like principles in a theory of melodic expectancy, as they have been in the *Implication-realisation* or *IR* theory (Narmour, 1990, 1992). He predicts that a statistical model created algorithmically from a suitable corpus of music would show melodic expectancy tendencies similar to those exhibited by humans. In order to demonstrate this, empirical data on melodic

τ_1		τ_2
Pitch	$\otimes$	Duration
Pitch	$\otimes$	IOI
Pitch	$\otimes$	DurRatio
Interval	$\otimes$	Duration
Interval	$\otimes$	IOI
Interval	$\otimes$	DurRatio
Contour	$\otimes$	Duration
Contour	$\otimes$	IOI
Contour	$\otimes$	DurRatio
ScaleDegree	$\otimes$	Duration
ScaleDegree	$\otimes$	IOI
ScaleDegree	$\otimes$	DurRatio
ScaleDegree	$\otimes$	FirstInBar
ScaleDegree	$\otimes$	IntFirstInPiece
ScaleDegree	$\otimes$	Interval
IntFirstInPhrase	$\otimes$	Contour
IntFirstInBar	$\otimes$	BarLength

Table 2.3: Product types (linked viewpoints) used in Pearce's research (adapted from Pearce, 2005).

expectancy in people is modelled by a multiple viewpoint system and then compared with a two-factor quantitative formulation of the IR theory (Schellenberg, 1997) supplemented with a tonality predictor. This latter combination was chosen as it is the simplest formulation still retaining its predictive capability with respect to data used in the first two of another series of experiments. The training corpus for these experiments comprised a total of nine hundred and three Canadian folk ballads, chorale melodies and German folk songs. Viewpoints were chosen by the viewpoint selection algorithm of §2.2.4, using regression coefficient as a performance measure (melodic expectancy observed in people was regressed on that shown by the model given the same stimuli).

The first experiment concerned eight musical intervals (the context) followed by twenty-five different chromatic continuation pitches. In the original experiment (Cuddy and Lunny, 1995), musically trained and untrained subjects rated the continuations on a scale from one to seven. An *analysis of variance* (ANOVA) revealed that there was a "significant interaction between context interval and continuation tone," but that musical training had no discernible effect; a mean of the ratings across all subjects was therefore taken. Pearce (2005) used a trained LTM to produce a probability distribution over the set of continuation pitches for each of the context intervals. The multiple viewpoint system which matched the human data best, {Interval $\otimes$ Duration, IntFirstInPiece, IntervalClass}, accounted for about 72% of the variance. The two-factor IR theory formulation, supplemented with the tonal region predictor described by Krumhansl (1995), fitted the data equally well.

The second experiment presented eight folk song excerpts (half major, half minor) followed by fifteen different diatonic continuation pitches (Schellenberg, 1996). The contexts were played in such a way that the metrical structure could be perceived. Other than the foregoing, the procedure was the same as that of the first experiment. This time, Pearce (2005) used a model comprising both an LTM and an STM. The system achieving the closest fit with the human data, {`IntFirstInBar`, `IntFirstInPiece`, `ScaleDegree` $\otimes$ `Interval`, `Pitch` $\otimes$ `IOI`}, accounted for about 83% of the variance. This was better than the 75% accounted for by the two-factor formulation supplemented with key profiles (Krumhansl and Kessler, 1982) to predict tonality. The difference in performance was statistically significant.

The third experiment involved selecting the expected pitch of each note in turn of two chorale melodies (Manzara et al., 1992) using a betting paradigm (Cover and King, 1978), resulting in the production of *entropy profiles* for the melodies. The multiple viewpoint system matching the human data most closely, {`IntFirstInPiece`, `ScaleDegree` $\otimes$ `DurRatio`, `Interval` $\ominus$ `FirstInPhrase`}, accounted for about 63% of the variance. This was far better than the 13% accounted for by the two-factor formulation supplemented with key profiles. It is interesting to note that, for example, the system {`ScaleDegree` $\otimes$ `Interval`, `Interval` $\otimes$ `Duration`, `IntFirstInBar`, `ScaleDegree`$\otimes$`FirstInBar`, `Interval` $\ominus$ `FirstInPhrase`}, although fitting the human data less closely, is better at this prediction task.

Finally, Pearce (2005) tests the hypothesis that models developed above can generate melodies, in a particular style, which are considered to be as successful, original and creative as existing ones. The following three models trained on the set of chorale melodies were chosen for evaluation: System A is the single viewpoint system {`Pitch`}; System B is the multiple viewpoint system {`IntFirstInPiece`, `ScaleDegree`$\otimes$ `DurRatio`, `Interval` $\ominus$ `FirstInPhrase`}; and System C is the multiple viewpoint system {`Interval` $\otimes$ `Duration`, `ScaleDegree` $\otimes$ `IntFirstInPiece`, `Pitch` $\otimes$ `Duration`, `ScaleDegree`$\otimes$`FirstInBar`, `Interval` $\ominus$ `Tactus`, `ScaleDegree`$\otimes$`Duration`, `Interval`$\otimes$ `DurRatio`, `IntFirstInPiece`, `Interval` $\ominus$ `FirstInPhrase`}. Melodies generated by these models using Metropolis sampling were evaluated by means of a development of the consensual assessment technique (see §2.4).

Seven melodies were generated by each of Systems A, B and C, in all cases using the same seven melodies from the chorale data set as initial states for Metropolis sampling of pitches (*N.B.*, note durations remained the same throughout this process). These twenty-one new melodies, and the original seven, were played in a random order from quantised MIDI files with metre and phrasing emphasised. Sixteen experts rated the stylistic success, originality and creativity of each of the melodies on a scale from one to seven, and indicated whether or not they recognised them. The originality and creativity ratings were found to be inconsistent, possibly because these concepts were not particularly relevant to the chorale style; therefore they were not analysed further.

τ_1		τ_2
Tessitura		
ScaleDegree	$\otimes$	Mode
ScaleDegree	$\otimes$	FirstInPhrase
ScaleDegree	$\otimes$	LastInPhrase
Interval	$\otimes$	InScale
Interval	$\otimes$	Tessitura

Table 2.4: Additional viewpoint types used in the development of System D (Pearce, 2005).

On the other hand, there was a high level of inter-expert consistency for the success ratings. An analysis of these ratings prompted Pearce (2005, p. 199) to conclude “that none of the systems are capable of consistently generating chorale melodies which are rated as equally successful stylistically as those in Dataset 2” (*i.e.*, the chorale data set).

In an attempt to construct a better multiple viewpoint system for the generation of melodies, Pearce (2005) added six more viewpoints arising from his analysis of the experts’ comments (see Table 2.4). New derived type Tessitura (initially tessitura) has the value 0 if the pitch is within one standard deviation of the mean, 1 if above this range and −1 otherwise. Application of the viewpoint selection algorithm described in §2.2.4 resulted in the system {Interval $\otimes$ Duration, ScaleDegree $\otimes$ IntFirstInPiece, Pitch $\otimes$ Duration, ScaleDegree $\otimes$ Mode, ScaleDegree $\otimes$ LastInPhrase, ScaleDegree $\otimes$ Duration, Interval $\otimes$ DurRatio, Interval $\otimes$ InScale, Interval $\ominus$ FirstInPhrase, IntFirstInPhrase}, known as System D. This model predicted the chorale data set with a cross-entropy of 1.91 bits/symbol (*cf.* 1.95 bits/symbol for System C). Only one melody generated by System D was presented, and it was not rated by the panel of experts; but it was considered by Pearce (2005) to be “much more coherent than the melodies generated by System C.”

2.5.3 The Statistical Modelling of Harmony

2.5.3.1 Finite Context Grammars

Markov Models Clement (1998) uses first-order Markov chains to demonstrate that two distinct (though artificial) styles of harmonic progression can be learned. The N-gram context is a single harmonic symbol; therefore the probability of a particular harmonic symbol appearing is dependent only upon the harmonic symbol that immediately precedes it. Bearing in mind that longer contexts should more accurately capture harmonic styles, Ponsford et al. (1999) improve on this by using N-grams of up to third-order to create statistical models of underlying harmonic movement from a corpus of seventeenth century French *sarabandes.* Again, the N-gram contexts, although larger, are solely harmonic; no account is taken of the melody.

Ponsford et al. (1999) experiment with different annotations of the corpus: piece boundaries only indicated; piece and phrase boundaries; piece, phrase and bar boundaries; and piece, numbered phrase and bar boundaries. When using an N-gram model to generate novel sequences of harmonic symbols by random sampling, a starting context of $n - 1$ dummy symbols is employed. Generated sequences not matching an annotated template are rejected; to save time, this matching is done on a phrase by phrase basis. They note that the generated pieces nearly always begin and end in the same key, and contain characteristic sequences of chords. Generally, phrases are well formed and end in appropriate cadences. The pieces as a whole are more convincing when numbered phrase-boundary annotations are used.

Biyikoğlu (2003) uses Markov models to study harmonic syntax and the relationship between melody and harmony. A harmonic analysis module takes as its input a corpus of 170 chorales (split into major and minor) harmonised by J. S. Bach. Its output, which is used to train separate second- and third-order Markov models, comprises sequences of chord symbols and annotation symbols (*start*, *end*, *bar* and *phrase*). Note that melody is completely removed from these training sequences. This being the case, in order to generate harmonic progressions for unseen melodies, it is first necessary to divide the melodies into segments of (usually) a crotchet in duration. A set of chord symbols is then assigned to each segment, such that each chord in the set contains the pitch class of at least one melody note appearing in the segment. The probability that one of the chord symbols in the set will be chosen to harmonise a segment is conditional only on the harmonic context.

Once the model has generated a harmonic symbol for every segment in the melody, the four-part harmonisation is completed by using a strategy similar to that described by Cope (2001). In general when attempting to fill in the actual notes of a chord, the notes of the previous chord should already have been found. A combination of the notes of the previous chord and the melody note and harmonic symbol of the current chord are compared with progressions in the corpus; when a match is found, the notes of the second of the matched pair of chords are copied to the current chord in the generated harmony. In practice, however, a complete four-part harmonisation cannot be achieved due to data sparseness. In order to work around this problem, a constraint-based element is introduced into the system in the form of voice leading rules.

Allan (2002) suggests that melody can be properly taken into consideration in statistical models by using Markov chains in which a context of melody notes is added to the historical harmonic context. He uses contexts of up to eight symbols and compares models with harmonic context only, melodic context only and both harmonic and melodic context. The models are evaluated by calculating the cross-entropy of a test set of chorales (see §2.4). In general, as Markov chain context size increases, cross-entropy increases for models with a single context length and decreases for smoothed models (back-off and additive smoothing), thereby demonstrating the superiority of smoothed

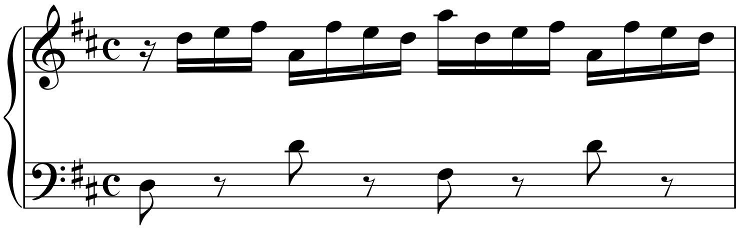


Figure 2.11: First bar of Prelude V from the first set of Bach's "Forty-eight."

models in the domain of harmony. Specifically, models using melodic context only (including the current melody note requiring harmonisation) perform better than models using harmonic context only; and smoothed models using both harmonic and melodic context perform best of all (mixed context models are not the best performing of those with a single context length, however). Unfortunately, the models are not used to generate harmonisations to previously unseen melodies.

Dictionary-based Models Assayag et al. (1999) describe a *dictionary-based* approach to the machine learning of music. The models they construct are similar to variable-order Markov models, but without smoothing. Initially, a *motif dictionary* is built by applying an incremental parsing algorithm (Ziv and Lempel, 1978) to a suitable representation of some existing music. A greater number of motifs can be extracted by running the algorithm again, starting on the second symbol of the data set, and so on. The motif dictionary, which includes the number of times each motif occurs in the data set, is then converted into the *continuation dictionary*, which comprises contexts and their associated continuation (prediction) probability distributions. The contexts range in length from single symbols to arbitrarily long sequences.

In order to create models of polyphonic music, a means of transforming the music into a sequence of discrete events must be found. Assayag et al. (1999) do a full expansion of the music (see Figure 2.12 in §2.5.3.2); but rather than breaking up notes completely, they have special symbols for note continuations (*e.g.*, '**a**' (in bold font) is a continuation of note 'a'). For two-part polyphony, an event is represented by the 3-tuple (higher note, lower note, duration), with rests indicated by dashes. Assuming that the duration of a semiquaver is represented by d_1 and ignoring octaves, the first half of Figure 2.11 can then be converted into the following sequence:

$$(-, d, d_1)(d, \mathbf{d}, d_1)(e, -, d_1)(f\sharp, -, d_1)(a, d, d_1)(f\sharp, \mathbf{d}, d_1)(e, -, d_1)(d, -, d_1).$$

This representation can be generalised to take account of any number of musical parts.

When generating music from such models, a maximum context length is chosen. Assuming that at least that number of tuples has already been generated, the system looks for the maximum sized context in the continuation dictionary. If it is there, the associated continuation probability distribution is randomly sampled, thereby generating the next tuple; if it is not, the context is shortened by one tuple at a time, until a match is found. Assayag et al. (1999) note that results have been inconsistent overall, but that subsequences demonstrate convincing harmony and counterpoint. Bearing in mind that the system has been developed to improvise a part (in real time) in response to a human performer, there is certainly scope to apply this approach to the harmonisation of melodies.

Hidden Markov Models Allan (2002) initially uses a first-order HMM with melody notes as observed events and harmonic symbols as hidden states. This was found to have approximately the same prediction performance (measured by cross-entropy) as the equivalent N-gram model, although smoothed high order N-gram models afford superior prediction performance. He notes, however, that because the Viterbi algorithm (Viterbi, 1967) can be used to find the globally most probable sequence of hidden states from an observed event sequence, even a simple HMM is effectively able to plan ahead while generating a harmonisation to a melody. A problem with finding the most probable solution is that the generated harmonisations are always more probable than Bach's own. A later, complete harmonisation model also uses HMMs. Following Hild et al. (1992), the overall harmonisation task is divided into three subtasks:[13]

1. Harmonic skeleton: each beat is assigned harmonic symbols using the initially developed HMM.
2. Chord skeleton: the notes of the chords are filled in using an HMM which has harmonic symbols as observed events and complete chords as hidden states.
3. Ornamentation: notes occurring off the beat are added by means of an HMM having observed events comprising a combination of the current harmonic symbol and notes on the current and following beat, and hidden states consisting of movement between those notes.

The complete harmonisation model was also evaluated by calculating cross-entropies. This showed that ornamentation was the least predictable task, followed by the chord skeleton task; the harmonic skeleton task was by far the most predictable (having the lowest cross-entropy). Allan and Williams (2005) note that the generated harmonisations could be better with respect to the flow of the individual parts.

2.5.3.2 Multiple Viewpoint Systems

Besides using them to model melody (§2.5.2.2), Conklin and Cleary (1988) use multiple viewpoints to model polyphony. The viewpoints `vint` (the interval between simultaneous events in different voices, taken at crotchet time intervals) and `voicerange` (the

[13] The first two subtasks are merged in later work (Allan and Williams, 2005).

allowable set of pitches for a particular voice) are used in addition to the previously described `melody`, `melint`, `pitchclass` and `duration` viewpoints. Note that `voicerange` is sometimes used in combination with `melint` such that only the intersection of the `melint` and `voicerange` prediction sets are used. The training data consists of a single sixteenth century piece of two-part polyphonic music. When generating polyphony, Conklin and Cleary (1988) add a number of events to each voice by hand in order to provide sufficient context.

The multiple viewpoint system {`melint`, `duration`, `pitchclass`} produces a fragment of music in which the two parts are independent, resulting in dissonance and the crossing of parts (*N.B.*, in this one case, the melodic model is used rather than the polyphonic one). One way to reduce (but in general, not eliminate) part crossing is to combine `melint` with `voicerange`. A fragment of three-part polyphony generated using this combination along with `duration` and `pitchclass` contains no part crossing at all; but only because the lowest note in the higher voice happens to coincide with the highest note in the lower voice in the training data. The fragment still exhibits dissonance, but this can be removed by using `vint`. The system {`vint`, `duration`, `pitchclass`, `voicerange`} produces reasonable two-part polyphony without dissonance, but the parts are not very smooth. In order to achieve flowing melodic lines, `melint` must also be included. For the final generation using the polyphonic model, the system {`vint`, `duration`, `melint`, `pitchclass`, `voicerange`} is used. In this case, the `vint` and `melint` prediction counts are added, and the result intersected with the `pitchclass` and `voicerange` prediction sets. This results in better melodic lines, but unfortunately also in a deterioration in the way the two voices fit together; it seems likely, therefore, that the intersection of the `vint` and `melint` prediction sets would result in the generation of better polyphony.

Conklin and Cleary (1988) also generate five bars of four-part polyphony using a simple hand-crafted model involving only the `vint`, `duration` and `pitchclass` viewpoints. Some dissonant intervals appear in the music, possibly because of the way the viewpoints are combined. There is also quite a lot of part crossing; the rhythm is very random; and there is little melodic flow, almost certainly due to the absence of `melint` from the model.

Conklin (2002) uses the multiple viewpoint representation for the discovery of vertical (*i.e.*, harmonic) patterns in a corpus of one hundred and eighty-five Bach chorales. This work extends that of Conklin and Anagnostopoulou (2001), where only patterns within individual voices are considered (see §3.3). Three types of music object are described, each of which has an *onset time* and a *duration* (*cf.* Smoliar, 1980):

1. The *Note* object type is defined as some suitable representation of absolute pitch, such as pitch class and octave.

2. The *Seq* object type is a sequence of objects of the same type (usually *Note* or

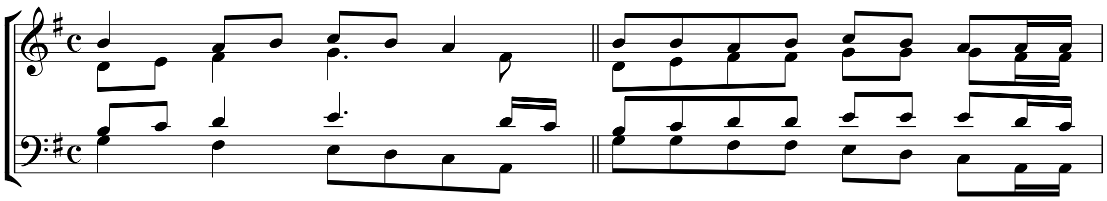


Figure 2.12: Excerpt from Bach chorale BWV 260: first bar original, second bar expanded (adapted from Conklin, 2002).

Sim), which have no temporal overlap. A sequence is enclosed in square brackets [ ].

3. The *Sim* object type is a simultaneity of objects of the same type (usually *Note* or *Seq*) having a common onset time. A simultaneity is enclosed in angle brackets $\langle\ \rangle$.

Formally, music objects are defined recursively as:

$$\begin{aligned} M &::= \mathit{Note} \mid \mathit{Seq}(M) \mid \mathit{Sim}(M) \\ \mathit{join} &: \mathit{Seq}(X) \times X \to \mathit{Seq}(X) \\ \mathit{layer} &: \mathit{Sim}(X) \times X \to \mathit{Sim}(X). \end{aligned}$$

The most basic way of representing the first bar of Figure 2.12 (octaves unspecified) is as a single *Sim* object containing four *Seq* objects:

$$\langle [B, A, B, C, B, A], [D, E, F\sharp, G, F\sharp], [B, C, D, E, D, C], [G, F\sharp, E, D, C, A] \rangle.$$

In order to discover vertical patterns, however, the music must be represented as a sequence of simultaneities. *Natural* partitioning does this by creating a new *Sim* object each time the four parts have a common onset time; therefore the first bar of Figure 2.12 becomes:

$$[\langle [B], [D, E], [B, C], [G] \rangle, \langle [A, B], [F\sharp], [D], [F\sharp] \rangle, \langle [C, B, A], [G, F\sharp], [E, D, C], [E, D, C, A] \rangle].$$

Unfortunately, this technique tends to under-partition harmony; therefore another, called *full expansion*, is preferred. Notes are split up such that for every individual onset time in the original music, there are now onsets in all four parts, as shown in the second bar of Figure 2.12. This is represented as:

$$[\langle B, D, B, G \rangle, \langle B, E, C, G \rangle, \langle A, F\sharp, D, F\sharp \rangle, \ldots, \langle A, G, E, C \rangle, \langle A, F\sharp, D, A \rangle, \langle A, F\sharp, C, A \rangle].$$

The tendency of this technique to over-partition is obviated to some extent by using threaded viewpoints at, for example, crotchet intervals.

The multiple viewpoint framework can be extended to model sequences of any of the object types defined above. Of particular interest here, however, are viewpoints associated with *Sim(Note)* objects. Such a viewpoint, used in this work, is pitch-class interval or `IntervalClass` (interval modulo 12), threaded at crotchet intervals. The vertical viewpoint elements comprise integers for each part, starting with the bass, enclosed in square brackets. The viewpoint sequence resulting from the second bar of Figure 2.12 is:

$$[[11, 3, 4, 10], [10, 2, 1, 3], [8, 0, 0, 9]].$$

If at least one of a set of viewpoint patterns occurs in x pieces in the corpus, then the *coverage* of that set is x pieces. A *shortest significant pattern* contains within its sequence no other significant patterns, whereas a *longest significant pattern* is not a subsequence of any other significant pattern. Shortest significant patterns, being the most general, appear in a large number of pieces.

For viewpoint `IntervalClass`, a set of thirty-two shortest significant patterns was discovered in the corpus, with a coverage of one hundred and forty-two pieces. Amongst the most significant of these patterns were various major and minor versions of the harmonic progression ii^{7}b – V – I or i, commonly found at cadences. The most significant pattern discovered (*i.e.*, the one with the smallest p-value) was $[[2, 10, 11, 0], [5, 9, 8, 10]]$, which had a coverage of thirty-six pieces. Conklin (2002, p. 41) acknowledges that a weakness of this technique "is that it will unavoidably sample accented non-harmonic tones."

An improvement to the full expansion idea (Assayag et al., 1999) makes use of symbols which indicate note continuations (or tied notes); quite simply, bold symbols represent continuations. The second bar of Figure 2.12 is then represented as:

$$[\langle B, D, B, G\rangle, \langle \mathbf{B}, E, C, \mathbf{G}\rangle, \langle A, F\sharp, D, F\sharp\rangle, \ldots, \langle A, \mathbf{G}, \mathbf{E}, C\rangle, \langle \mathbf{A}, F\sharp, D, A\rangle, \langle \mathbf{A}, \mathbf{F\sharp}, C, \mathbf{A}\rangle].$$

One disadvantage of this, however, is the proliferation of symbols. This can be avoided by using a single symbol (say +) to represent a continuation (Pinkerton, 1956); it is completely obvious which notes are being continued:

$$[\langle B, D, B, G\rangle, \langle +, E, C, +\rangle, \langle A, F\sharp, D, F\sharp\rangle, \ldots, \langle A, +, +, C\rangle, \langle +, F\sharp, D, A\rangle, \langle +, +, C, +\rangle].$$

Within the multiple viewpoint framework, a boolean test type could be used to distinguish between the start of a note and (if applicable) its continuation. Such a test type is introduced in §3.4.4.

In very recent work on subsumption in musical patterns, Bergeron and Conklin (2011) describe $\mathcal{VVP}$ (vertical viewpoint patterns), which comprise slices of polyphonic music expressed as viewpoints. The slices are created by full expansion and now contain

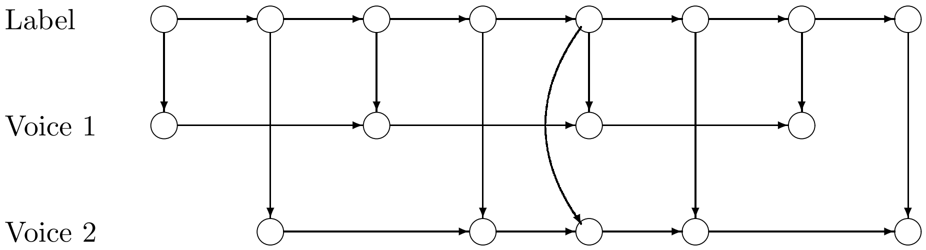


Figure 2.13: Bayesian network representation of a harmonic analysis model (adapted from Raphael and Stoddard, 2003).

continuation information. A relational pattern language called $\mathcal{R}$ is also developed, and rules governing the construction of well-formed $\mathcal{VVP}$ patterns ensure that they can be translated into $\mathcal{R}$. Bergeron and Conklin (2011, p. 8) show that "subsumption for $\mathcal{VVP}$ is sound and complete with respect to subsumption in $\mathcal{R}$." The identification of subsumption facilitates, for example, the tracking of pattern development in music.

2.5.3.3 Graphical Models

Bayesian Networks Raphael and Stoddard (2003) are interested in the harmonic analysis of MIDI files. Initially, they trained a first-order HMM (see Figure 2.3) to label contiguous regions of a piece of music with key, mode and functional chord. The hidden states were the labels, and the observed states were collections of pitches. Heuristics were used to substantially reduce the number of parameters. The authors were worried, however, by their assumption of the conditional independence of pitches, which ignores the existence of independent parts or voices in much music. As a result, they propose a probabilistic graphical model which "regards the data as a collection of voices where the evolution of each voice is conditionally independent of the others, given the harmonic state" (Raphael and Stoddard, 2003, p. 181). The Bayesian network representation of the model is reproduced in Figure 2.13. The assumption of conditional independence of the voices is almost certainly incorrect; but even when this simplifying assumption is made, the resulting model for only two voices is far more complex than the original HMM.

Paiement et al. (2005a,b, 2006) present probabilistic graphical models of jazz harmony as Bayesian networks, but convert them to Markov random fields (see §2.2.5.3) to carry out inference. Their first chord sequence model (without reference to melody) is constructed as "a binary tree structure suggested by the [metre] of the jazz standards in [their] database" (Paiement et al., 2005a, p. 6). The Bayesian network representation of the model is reproduced in Figure 2.14. Level 2 nodes represent discrete hidden variables which model local dependencies in addition to being conditioned by the global dependencies of level 1 (also discrete hidden variables). The black nodes of level 3 represent continuous observed variables with Gaussian distributions. The last eight bars of fifty-

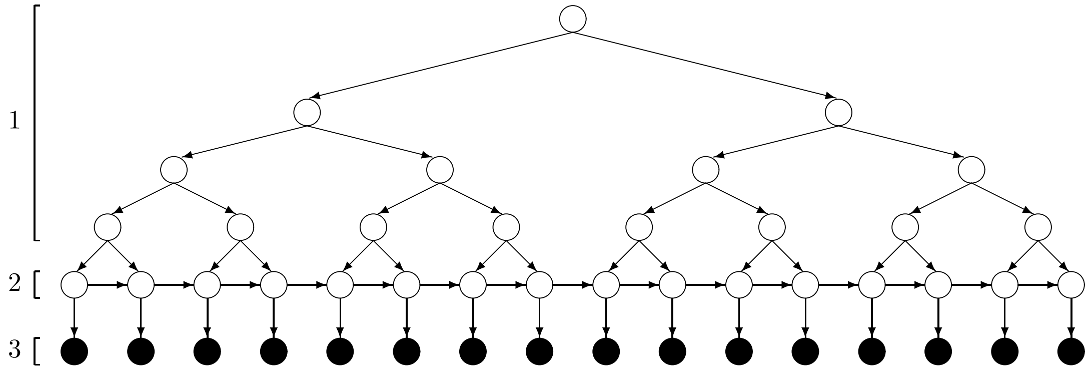


Figure 2.14: Bayesian network representation of a chord sequence model (adapted from Paiement et al., 2005a). White nodes represent discrete hidden variables and black nodes continuous observed (Gaussian) variables.

two jazz standards in MIDI format (all in the key of C) are used as a training set. Four beats to the bar and a (four note) chord change every two beats gives sixteen chords per extract (long chords are split into two-beat lengths). These chords are converted to the first (continuous distributed) chord representation described in §2.3.4.2 prior to training. Variations of the model in Figure 2.14 all have better prediction ability than a directly comparable HMM, and an example chord sequence generated by the Bayesian network model has a better overall musical structure than one generated by the HMM.

Their second chord sequence model employs the second (discrete substitution probability) chord representation described in §2.3.4.2. This model has the same structure as the one in Figure 2.14, except that there is an additional row of nodes in a fourth level (each node conditioned only by the corresponding level 3 node) and all nodes represent discrete variables, which are observed only in level 4. Hidden level 3 nodes effectively represent "first try" chords which may or may not be substituted to give the observed chords. Variations of the model again have a better prediction performance than a directly comparable HMM. The negative log-likelihoods are much lower for this model than for the first (although they cannot strictly be compared because of the different representations).

Next, they model root progressions given the melody, which is represented as described in §2.3.4.2 except that t is equal to the chord length (two beats). The model has the same structure as the one in Figure 2.14, except that there are two additional levels, each comprising a single row of nodes. Again, there are only vertical dependencies in levels 4 and 5, which respectively represent observed root notes (discretely modelled like the chords in the second chord sequence model above, giving substitution probabilities) and observed (continuous distributed) melodic variables. In this case, the corpus comprises eight bar extracts of forty-seven jazz melodies with root progressions. This model has a much better prediction ability than a directly comparable HMM.

Finally, they design a model which predicts chord symbols (not actual chords) given

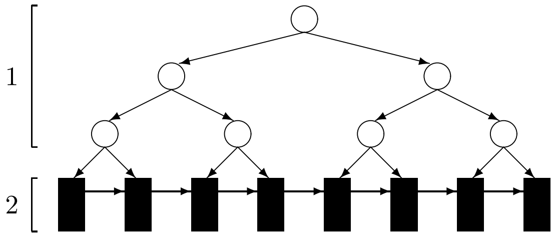


Figure 2.15: Bayesian network representation of a model which predicts chords given their root and the melody (adapted from Paiement et al., 2006). White nodes represent discrete hidden variables and black rectangular nodes represent subgraphs as shown in Figure 2.16

their root and the melody, which makes use of the third chord representation described in §2.3.4.2. The high level Bayesian network representation of this model is shown in Figure 2.15, where level 1 nodes represent discrete hidden variables, as before. The detail within the black rectangles in level 2 is revealed in Figure 2.16, where node H represents discrete hidden variables capturing local dependencies, and observed node R is assigned the pitch class of the root. The bottom row of nodes is the melodic representation used earlier, except that its components are now relative to the root rather than to C, and individual components have conditional probabilities. The next row up comprises the structural components of the chord, including node B, which is a 12-valued (pitch class) random variable representing the bass note; it is this row which we are interested in predicting. This time, the HMM (Figure 2.15 with the level 1 nodes removed) has a better prediction ability than the Bayesian network model. Paiement et al. (2006) speculate that with the roots given, the global dependencies are essentially redundant.

Markov Random Fields Bellgard and Tsang (1994) use a network of Boltzmann machines called an *effective Boltzmann machine* to harmonise chorale melodies (recalling that Boltzmann machines are a category of Markov random fields). The Boltzmann machine (BM) is trained on five chorale melody harmonisations, normalised to the same key, using window sizes of 5 and 3. Passing notes are removed from the corpus, and long notes are split such that all events are the same length. An event is represented by a vector (called a *scale*) of binary input/output (IO) units, which in turn represent pitches from G2 to F5, phrase start and end, and spacer (placed between chorales). The BM comprises a set of IO units and a set of hidden units, with a bidirectional weighted connection between each hidden unit and each IO unit. The energy of the BM, defined as minus one times the sum of the weights of activated pairs of units, is minimised during the learning phase by adjusting the weights according to the standard BM learning algorithm (Hinton and Sejnowski, 1986).

A melody to be harmonised is represented by a sequence of scales, each of which has a single pitch IO unit set to 1. The effective Boltzmann machine (EBM) consists of

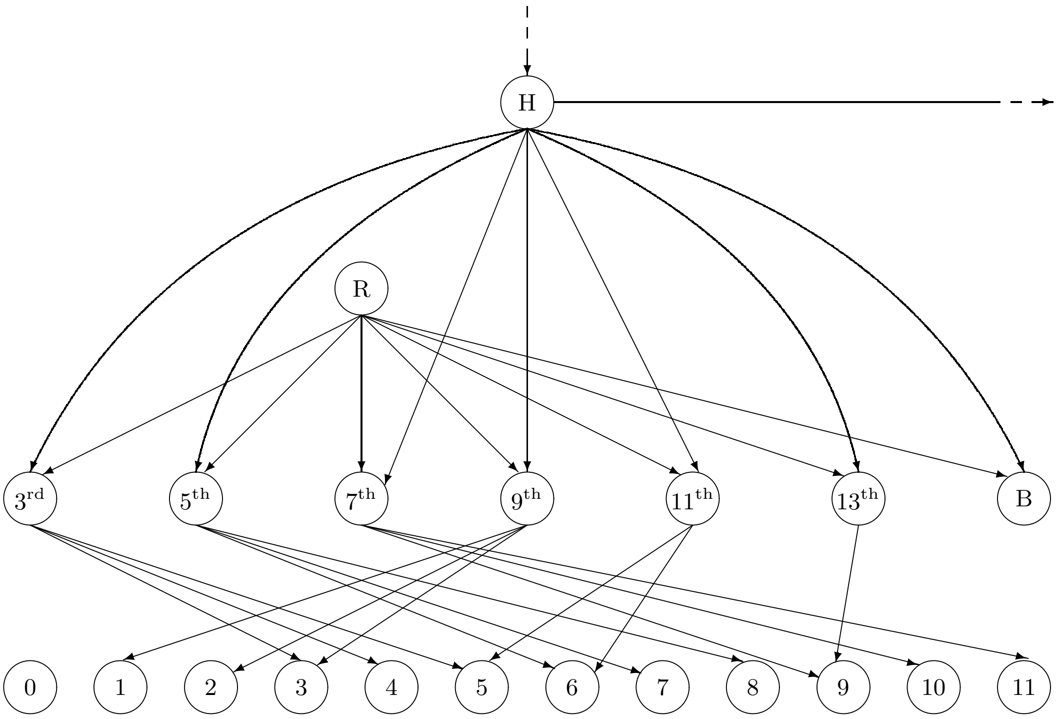


Figure 2.16: Subgraph of the Bayesian network representation of a model which predicts chords given their root and the melody, shown in Figure 2.15 (adapted from Paiement et al., 2006).

overlapping BMs (the number of BMs varies with the length of the melody). Assuming a window size of 5, one BM covers events 1 to 5, another events 2 to 6, and so on. The harmonisation is completed by minimising the energy of the entire EMB by means of simulated annealing (Kirkpatrick et al., 1983).

The EBM is able to correctly reproduce the harmony of a melody from the training set; indeed, it is able to do so given only the bass line, and also given only the first and last three chords. Given the first phrase of a melody not in the training set, however, it performs quite badly: two melody notes are left unharmonised; two chords contain five pitches (some of which are inappropriately spaced); the first chord contains only two pitches, a fourth apart; and the final chord is inappropriate for a phrase ending. To overcome these and other perceived problems, Bellgard and Tsang (1994) introduce three absolute constraints. Prohibited are: notes above a melody note, more than four pitches in a chord, and notes two semitones or less from a melody note. With these constraints in place, the EBM is able to produce much better harmony. It must be pointed out, however, that it is not entirely unheard of for the alto to go above the soprano in this type of music, as in the Praetorius (1571–1621) harmonisation of the old German melody *Es ist ein' Ros' entsprungen* (Vaughan Williams 1933, hymn no. 19). Worse, it is quite common for the alto to be within two semitones of the soprano (as, *e.g.*, in the chorale *O Wond'rous Love*, Bach 1938, No. 55). The likelihood is, then, that the harmony produced by the EBM will not be entirely typical of the style. It would be

Pitch class 0	●	○	○	●	○
Pitch class 1	○	○	●	○	●
Pitch class 2	○	●	○	●	○

Figure 2.17: A music sequence using three pitch classes (adapted from Pickens and Iliopoulos, 2005).

better to avoid the use of constraints such as these, and instead increase the size of the corpus.

Pickens and Iliopoulos (2005) use Markov random fields to gauge the similarity of polyphonic themes. They think of notes in terms of a graph of MIDI note number (representing pitch) against time; notes are considered to be either on (for some duration) or off, resulting in a representation that looks similar to a pianola roll. For the purpose at hand, the representation is further simplified by reducing the pitch information to twelve pitch classes, indicating only note onsets, and removing all duration information. The result is a matrix in which note onsets are shown as filled circles; some of the onsets may be simultaneous (see Figure 2.17). Finally, each position in the lattice is represented by a variable $n_{i,t}$, where each variable is either 1 (note onset) or 0 (otherwise). A model is developed which predicts the value of a variable from the values of other (usually nearby) variables.

Every variable $n_{i,t}$ has a *history* or *neighbourhood* $H_{i,t}$, which includes all variables up to and including time $t-1$ and usually some at time t:

$$H_{i,t} = \{n_{j,s} : s < t\} \cup \{n_{j,s} : s = t, j < i\}.$$

It is assumed that the probability of an onset of pitch class i at time t is completely determined by $H_{i,t}$. A dependency is expressed as a positive answer to the question: "*do onsets occur at all points in the lattice represented by a given subset of $H_{i,t}$?*" This subset S of $H_{i,t}$ is called the *support* of *feature function* f_S, which is defined as follows:

$$f_S(n_{i,t}, H_{i,t}) = n_{i,t} \prod_{n_{j,s} \in S} n_{j,s}$$

(*i.e.*, f_S is 1 *iff* all variables in S are 1). A *feature* is the union of a variable to be predicted with its support. Features are indexed relative to time t, but have absolute pitch class indices; for example, $\{n_{2,t}, n_{1,t}\}$, $\{n_{2,t}, n_{1,t}, n_{3,t-1}, n_{3,t-2}\}$ and $\{n_{2,t}, n_{0,t}, n_{2,t-2}, n_{0,t-2}\}$ are features in which $n_{2,t}$ is the variable to be predicted.

The field induction procedure (which closely follows Della Pietra et al., 1997) begins with a uniform structure containing individual onsets without dependencies; the weights of these atomic features are optimised. The set of candidate features at each stage is defined as the set of one onset extensions to the current structure, where the additional onset is not more than two simultaneities from the original. The optimum weight for each feature is calculated, and then the structure of the field is incrementally modified by adding the candidate feature which results in the greatest improvement in the log-likelihood of training data. The weights of all of the features in the structure are re-optimised, and then a new set of candidate features is assembled, and so on, until no further significant improvement to the log-likelihood can be obtained.

This approach can be applied to music information retrieval in the following way. A Markov random field model is created from a piece of music (the *query*), and then that model is used to predict each individual piece in a collection of music. If the model predicts a piece with a high probability (in other words, with a low cross-entropy), then the query and the piece are deemed to be similar, in that they could both have been sampled from the same distribution. Pieces are ranked according to this similarity measure.

2.6 Conclusions

We have presented a review of previous research relating to the computational modelling of music. In §2.2, various computational techniques that have been employed in such modelling (and which could be of use in the current research) were discussed, including constraint satisfaction, genetic algorithms, finite context grammars, multiple viewpoint systems and graphical models. A discussion of corpora and the representation of musical structure (including the multiple viewpoint representation) appeared in §2.3, following which the evaluation of computational models of music was discussed in §2.4. Finally, a summary of previous research on the computational modelling of music was presented in §2.5, including descriptions of work using constraint-based, evolutionary, connectionist, and multi-agent methods. Particular emphasis was placed on the statistical modelling of melody and harmony, with approaches using Markov models, dictionary-based models, HMMs, HHMMs, multiple viewpoint systems, Bayesian networks and Markov random fields being described.

The multiple viewpoint representation, in conjunction with variable order PPM models, is considered to be the ideal framework for the current research. It appears to have cognitive validity, as evidenced by the success of Pearce (2005) in using it to produce cognitive models of melodic expectancy. It is also expected to cope well with the complexity of four-part harmony in view of its ability to model underlying structure (with derived viewpoints) as well as surface structure (with basic viewpoints). Some modifications to the existing framework will be necessary, however. Finally, Conklin and Cleary (1988) suggest the intriguing possibility of implementing multiple viewpoint systems as neural networks, in order to exploit parallel computation and weight adaptation. This particular suggestion will remain, for the time being, a dream for the future.

Chapter 3

Central Ideas of the Research

3.1 Introduction

The ultimate goal of this research is to model the harmony of four-part non-homophonic music such as the chorale harmonisations of J. S. Bach. Music which is more complex than this must unfortunately be ruled out at this stage due to time limitations. It was originally intended that initial efforts would focus on the modelling of completely homophonic music; but the limited availability of such music for use as training data is problematic. Chants found in parish and cathedral psalters are very nearly homophonic, as are quite a number of the harmonisations found in, for example, Vaughan Williams (1933) and Nicholson et al. (1950). Changing harmony such that it becomes fully homophonic requires subjective judgement and the making of arbitrary decisions, however, which can contaminate the data. It was therefore decided that an unadulterated corpus of harmonised hymn tunes should be created, which will be described in detail in §3.5.

Having decided upon a suitable corpus, the problem facing us is how best to construct statistical models of harmony (and melody) which capture the musical style of that corpus. The solution is to use a machine learning approach in conjunction with the multiple viewpoint representation and Prediction by Partial Match. In high level terms, we take the corpus; run the viewpoint selection procedure detailed in Algorithms 3.1 to 3.7 (see §3.4.5.4); and use the resulting minimal cross-entropy multiple viewpoint systems to predict or generate harmony (or melody, as appropriate). Repeated use of a viewpoint domain construction procedure (see Chapter 4, especially Algorithms 4.1 to 4.4 for the most complex viewpoints) and the PPM algorithm (Cleary and Witten, 1984) is required throughout.

The current research is concerned with comparing several different novel developments of the multiple viewpoint framework, as applied to harmony; but the framework can be developed much further, and it is intended that these subsequent developments will form the basis of future research. Chapter 10 shows that as the framework is stretched, existing ideas such as backing off from an i^{th}-order model to an $(i-1)^{\text{th}}$-order model until a matching context is found, and similarly passing an escape probability down to models of ever-decreasing order for the purpose of calculating a complete prediction probability distribution, are radically rethought. It will be seen, for example, that differently shaped contexts of the same order are possible.

The current research also investigates how models of melody may be improved by having access to a much larger pool of viewpoints (especially linked viewpoints) than has been available in previous research.

There is a discussion about model structure (including representation) in §3.2, followed by a section on the modelling of melody in §3.3. The development of the multiple viewpoint framework for harmony is outlined in §3.4, including discussions on full expansion and viewpoint selection. The corpus is described in §3.5, and in §3.6 there is a brief outline of evaluation and the methodology to be employed in this research. Finally, in §3.7, some conclusions are drawn.

3.2 Model Structure

In the research described here, the word "model" can be applied at different levels of abstraction. In order to avoid confusion, this word is differently qualified at each level, resulting in the model taxonomy shown in the hierarchical structure of Figure 3.1. At the top of the structure is an *overall* model for the generation, analysis, perception or cognition of harmony. This overall model may consist of several *subtask* models (Allan, 2002; Hild et al., 1992; Phon-Amnuaisuk and Wiggins, 1999); for example, one such model might predict the bass part only. Each subtask model is a multiple viewpoint system in its own right. Each of these models may be further divided into one *long-term* and one *short-term* model (Conklin and Witten, 1995; Pearce, 2005, see §2.2.4) comprising a number of *viewpoint* models. A viewpoint model uses Prediction by Partial Match, and is made up of a number of different *N-gram* models of a particular linked viewpoint.[1] This hierarchical structure affords a convenient means of discussing different aspects of the research at the appropriate level of abstraction.

3.2.1 Overall Models

The focus of this research is an overall computational model which aims to harmonise melodies in ways which are stylistically characteristic of the corpus on which it was trained. Scientific methodologies guide its construction and evaluation. No attempt is made to construct any other type of overall model (*e.g.*, for harmonic analysis); but it is anticipated that the techniques developed during this research will be especially conducive to the creation of a future overall computational model for the cognition of harmonic movement. Cognitive science research methodologies will be brought to bear in the development of this future overall model. Note that prediction of test data means the assignment of probabilities to known events using the overall model.

[1] A primitive viewpoint is considered to be a special case of a linked viewpoint.

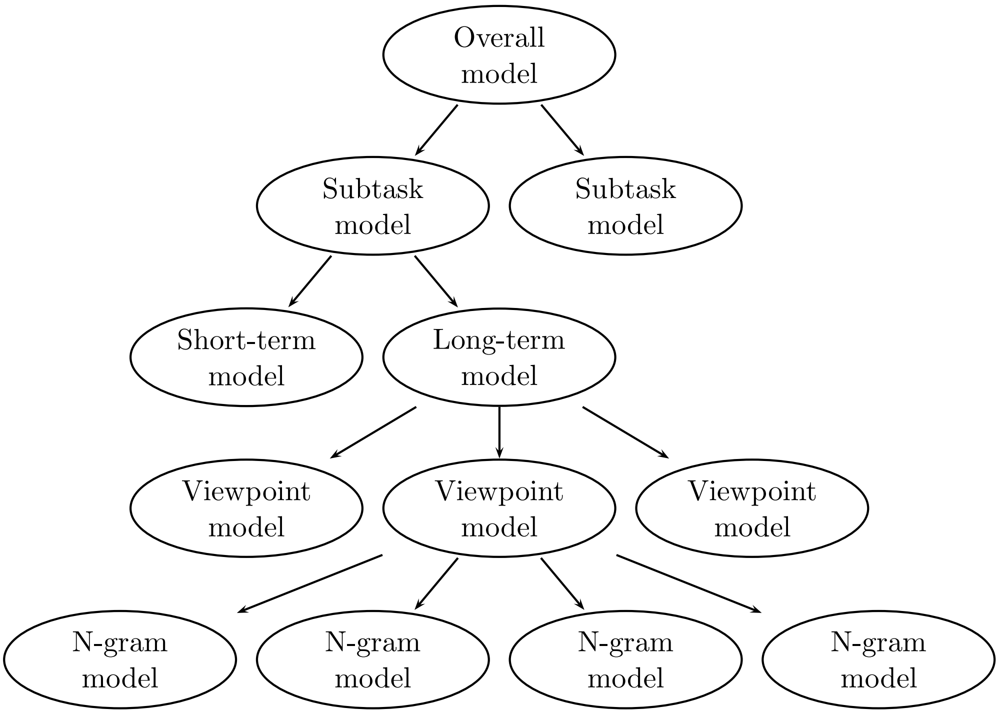


Figure 3.1: The hierarchical structure of a statistical model of harmony.

3.2.2 Subtask Models

For the purposes of the following discussion, it will be assumed in the first instance that an overall model for harmonising melodies in a particular style is being constructed. Allan (2002) follows Hild et al. (1992) in breaking up the harmonisation task into three subtasks, in the following order: "harmonic skeleton," "chord skeleton" and "ornamentation." In the "harmonic skeleton" subtask, harmonic function symbols are assigned to each beat; the actual notes of each chord are filled out in the "chord skeleton" subtask; and additional notes such as passing notes are supplied during "ornamentation." Wiggins (1998) suggests that the problem should be broken up even further, by for example choosing cadences first.

The time constraints of this research do not actually allow the division of the harmonisation task into more than a few subtasks; but here we hypothesise a subtask structure for future consideration which breaks up the task a great deal. The proposed structure approximately follows the one outlined in Figure 3.2. Harmonic function symbols are generated for all of the melody notes first, and then all of the bass notes are generated, followed by alto and tenor notes together. Each of these passes of the melody is broken up into three subtask models, which come into play in the following order: the first deals with cadences; the second is concerned with all chords except the cadence and a few pre-cadence chords in each phrase; and the third specialises in the pre-cadence chords, with the objective of knitting together the bulk of the harmony in each phrase with its cadence. Any "ornamentation" capability is likely to be integrated into the subtask

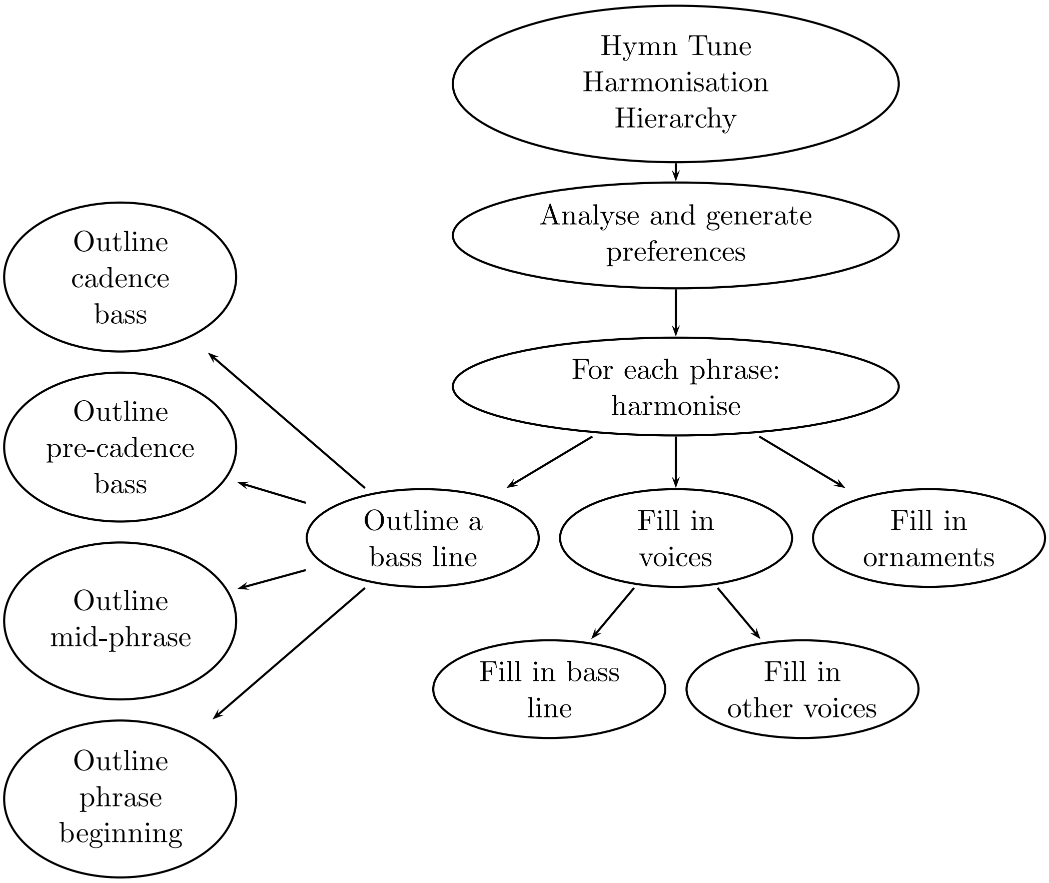


Figure 3.2: The hierarchical structure of a harmonisation process (adapted from Phon-Amnuaisuk and Wiggins, 1999).

models outlined above rather than making it a separate generation step.

Of course, this proposed structure is unlikely to be optimal; for example, generating the bass line before the harmonic function symbols may improve performance. Similarly, it may be better to generate all of the notes of a chord (and its associated harmonic function symbol) before moving on to the next one; also, perhaps for example the whole harmonic function symbol generation pass can be dispensed with. It would be possible to compare the performance of various different subtask structures.

There is one other subtask which must be added, or integrated into other subtasks, as a consequence of modelling fully expanded (see §2.5.3.1, §2.5.3.2 and §3.4.4) non-homophonic music. A given melody to be harmonised by the system will not be in expanded form; therefore in order to make use of the overall model for harmonisation induced from the expanded corpus, the melody must be expanded in some principled way.

Turning now to the construction of a future overall cognitive model of the perception of harmonic movement, it is almost certain that such a model will be different in terms of high level structure from a model which harmonises melodies. It is clear that certain structures can be immediately ruled out; for example, it would make no sense to predict a complete bass line before starting to predict harmonic function symbols. Predicting harmonic symbols at cadences before those at the beginning of the harmonic sequence would be equally nonsensical. In a cognitive model, prediction must be done in approximately chronological order.[2]

3.2.3 Long-term and Short-term Models

Each subtask model may comprise a *long-term model* (LTM) and a *short-term model* (STM) (the use of both is not mandatory). Conklin (1990) introduced the idea of using a combination of an LTM, which is a general model of a style derived from a corpus, and an STM, which is constructed as a single piece of music is being predicted or generated. The latter aims to capture musical structure particular to that piece. In this research, there is only one multiple viewpoint system per subtask model; *i.e.*, the LTM and STM comprise the same viewpoint models (this restriction is likely to be removed in future work, however).

Pearce (2005) proposes three ways of combining long- and short-term models, each of which employs a weighted arithmetic or geometric technique to combine prediction probability distributions (see §2.2.4). The first, here termed LS1 for convenience, consists in combining viewpoint distributions within each of the LTM and STM first, and then combining the two resulting distributions. This has been the method of choice in research to date. The second, LS2, effects a pair-wise combination of the distributions of identical viewpoints in the LTM and STM first, and then combines the resulting distributions. The third, LS3, combines all viewpoint distributions at once, irrespective of whether they are in the LTM or STM. These three methods are investigated in §5.2.4. Clearly, if, for example, only the LTM is used, only the first part of LS1 is relevant.

3.2.4 Viewpoint Models

3.2.4.1 Prediction by Partial Match

A *viewpoint model* is a weighted combination of various orders of N-gram model of a particular viewpoint type. Following previous related work (*e.g.*, Pearce, 2005), the N-gram models are combined by Prediction by Partial Match (Cleary and Witten, 1984). PPM makes use of a sequence of models, which we call a *back-off sequence*, for context matching and the construction of complete prediction probability distributions (*i.e.*, containing all possible predictions, however improbable). The back-off sequence begins with the highest order model, proceeds to the second-highest order, and so on. An *escape method* determines weights for prediction probabilities at each stage in this sequence, which are generally high for predictions appearing in high-order models, and vice-versa.

[2] Prediction is not necessarily completely chronological because of the phenomenon of retrospective listening (Narmour, 1992).

If necessary, a probability distribution is completed by backing off to a uniform distribution (also known as a -1^{th}-order model). In this research, we are using escape method C (see Witten and Bell, 1989, for a review of this and other escape methods). For any given model order, if r is the number of different symbols which follow a particular context, n is the total number of these symbols, and c_i is the number of times that the i^{th} symbol appears, then the prediction probability of the i^{th} symbol is

$$p = \frac{c_i}{n+r},$$

and the escape probability is

$$p = \frac{r}{n+r}.$$

For example, given the domain $\{\mathrm{A}, \mathrm{B}, \mathrm{C}\}$ and the sequence BABBAA, we calculate the probability of C appearing next in the sequence.[3] The counts and resulting probabilities for model orders 2 to -1 are shown in Table 3.1. There is no match with context AA in the second-order model, so we back off to the first-order model. Here, we are able to match context A, but not prediction C; therefore we note the escape probability of $\frac{1}{2}$, and move on to the 0^{th}-order model. Again, we are unable to match prediction C; so we note the escape probability of $\frac{1}{4}$. We are finally able to match prediction C, which has a probability of $\frac{1}{3}$, in the -1^{th}-order model; therefore, on the face of it, the probability of C appearing next in the sequence BABBAA is $\frac{1}{2} \times \frac{1}{4} \times \frac{1}{3} = \frac{1}{24}$. Notice, however, that by similar reasoning A and B each have a probability of $\frac{1}{4}$, which means that the sum of the probabilities in the distribution is $\frac{13}{24}$ rather than the required 1. Normalising the distribution gives C a probability of $\frac{1}{13}$.

A method known as *exclusion* (Cleary and Teahan, 1997) can be used in conjunction with escape to obtain more accurate estimates. The probabilities are calculated in the same way as before except that the counts of predictions already seen at a higher order are excluded from n. Using the same example, there is no match with context AA in the second-order model, so we back off to the first-order model. Here, we are able to match context A, but not prediction C; therefore we note the escape probability of $\frac{1}{2}$, and move on to the 0^{th}-order model. Again, we are unable to match prediction C; but this time, since A and B have already been seen in a higher-order model, $n = 0$ and the escape probability is 1. We are finally able to match prediction C in the -1^{th}-order model; at this stage the probability is shared equally between all previously unseen symbols, which means that the probability of a C is 1 in this model. The probability of C appearing next in the sequence BABBAA is therefore $\frac{1}{2} \times 1 \times 1 = \frac{1}{2}$. In simple terms, we know that A and B each have a probability of $\frac{1}{4}$ and that C is the only other symbol in the domain; it must therefore have a probability of $\frac{1}{2}$. Exclusion is used in this research.

Even when using exclusion, it is possible for prediction probability distributions to

[3] It should be borne in mind that the very limited statistics yielded by this short sequence will not produce realistic probabilities.

2^{nd}-order			1^{st}-order			0^{th}-order			-1^{th}-order		
Predictions	c	p	Pred'ns	c	p	Pred'ns	c	p	Pred'ns	c	p
AB → B	1	$\frac{1}{2}$	A → A	1	$\frac{1}{4}$	→ A	3	$\frac{3}{8}$	→ A	1	$\frac{1}{3}$
→ *Esc*	1	$\frac{1}{2}$	→ B	1	$\frac{1}{4}$	→ B	3	$\frac{3}{8}$	→ B	1	$\frac{1}{3}$
BA → A	1	$\frac{1}{4}$	→ *Esc*	2	$\frac{1}{2}$	→ *Esc*	2	$\frac{1}{4}$	→ C	1	$\frac{1}{3}$
→ B	1	$\frac{1}{4}$	B → A	2	$\frac{2}{5}$						
→ *Esc*	2	$\frac{1}{2}$	→ B	1	$\frac{1}{5}$						
BB → A	1	$\frac{1}{2}$	→ *Esc*	2	$\frac{2}{5}$						
→ *Esc*	1	$\frac{1}{2}$									

Table 3.1: PPM model (maximum 2^{nd}-order) using escape method C for the sequence BABBAA, assuming a domain of $\{\mathrm{A}, \mathrm{B}, \mathrm{C}\}$, in the style of Cleary and Teahan (1997).

have a total probability mass of < 1 due to left-over escape probability (*e.g.*, because there is no need to escape to the uniform distribution to complete the distribution). In all such cases the distributions are normalised. This was not done in earlier text compression work because it was not necessary to complete distributions. Within the multiple viewpoint framework, backing off only until the required prediction is found results in different sets of predictions in the respective distributions (once they have been converted into distributions over the domain of the basic type), which makes it impossible to properly combine the distributions so that the overall prediction probability can be found. This, fundamentally, is why distributions must be completed within this framework.

Escape method C has been chosen over other escape methods because it has been shown to perform well (Pearce, 2005) and is relatively easy to implement. PPM* and interpolated smoothing (which Pearce 2005 also found to perform well) are not implemented, as they have been thoroughly investigated and are not novel aspects of the work.

3.2.4.2 Viewpoint Types

The set of basic and derived viewpoint types to be used in this research include those in Table 2.2 (except for `RestLength`, `PitchClass`, `IntervalClass`, `Interval` $\ominus$ `LastInPhrase` and `PhraseLength`) plus `Tessitura` (Table 2.4). Viewpoint type `Pulses` has been redefined as the number of tactus pulses in a bar rather than as the upper digit of the time signature, which means that test type `Tactus` now indicates the correct tactus pulses for compound time signatures. In addition, there are fifteen new viewpoint types, as described below.

Basic type `Piece` marks the beginning and ending of a piece, which are important from the point of view of musical structure: $[\texttt{Piece}] = \{-1, 0, 1\}$, where $[\![1]\!]_{\texttt{Piece}}$ = first event in piece, and $[\![-1]\!]_{\texttt{Piece}}$ = last event in piece. Test types `FirstInPiece` (first event in piece) and `LastInPiece` (last event in piece) are trivially derived from type `Piece`.

Type TactusPositionInBar is the position in a bar of an event in terms of tactus beats; for example, if the onset of an event were five (completed) quavers into a bar having four crotchet tactus pulses, the TactusPositionInBar value would be 2.5. This viewpoint is defined as:

$$\Psi_{\texttt{TactusPositionInBar}}(e_1^j) = \frac{\Psi_{\texttt{PositionInBar}}(e_1^j) \times \Psi_{\texttt{Pulses}}(e_1^j)}{\Psi_{\texttt{BarLength}}(e_1^j)}.$$

Type Metre is a metrical level viewpoint, which assumes that there are either 2, 3 or 4 tactus pulses in a bar. The question is, which beats in a bar (if any) can be considered to be metrically equivalent? In a bar with 4 pulses, the first (having a TactusPositionInBar value of 0) is assigned a value of 3, the third a value of 2, and the second and fourth pulses a value of 1 (3 1 2 1). In a bar with 3 pulses, the first is given a value of 3, and the second and third pulses a value of 1 (3 1 1). In a bar with 2 pulses, the first is 3 and the second 1 (3 1). Of course, the values assigned to the different metrical positions can be debated, and some experimentation can be carried out in the future; but for this research, the values are considered to give a reasonable estimate of metrical equivalence and are fixed. Viewpoint Metre is formally defined as follows:

$$\Psi_{\texttt{Metre}}(e_1^j) \quad = \quad \begin{cases} 3 & \text{if } \Psi_{\texttt{TactusPositionInBar}}(e_1^j) = 0 \\ 2 & \text{if } \Psi_{\texttt{TactusPositionInBar}}(e_1^j) = 2 \text{ and } \Psi_{\texttt{Pulses}}(e_1^j) = 4 \\ 1 & \text{if } \Psi_{\texttt{TactusPositionInBar}}(e_1^j) = 1 \text{ or } 3 \\ 1 & \text{if } \Psi_{\texttt{TactusPositionInBar}}(e_1^j) = 2 \text{ and } \Psi_{\texttt{Pulses}}(e_1^j) = 3 \\ 0 & \text{otherwise.} \end{cases}$$

All but one of the remaining new viewpoint types are threaded and self-explanatory. The threaded types are Pitch $\ominus$ Tactus, ScaleDegree $\ominus$ Tactus, ScaleDegree $\ominus$ FirstInBar, ScaleDegree $\ominus$ FirstInPhrase, ScaleDegree $\ominus$ LastInPhrase, Contour $\ominus$ Tactus, Contour $\ominus$ FirstInBar, IOI $\ominus$ Tactus and IOI $\ominus$ FirstInBar. Finally, type Cont is a basic viewpoint introduced specifically for use in the modelling of harmony, which is described in detail in §3.4.4.

One of the major differences in this area from the work of Pearce (2005) is that linked viewpoint types are not restricted to those in Table 2.3 and Table 2.4; any primitive viewpoint may be linked with any other primitive viewpoint, provided such links are able to predict at least one attribute. Another difference is that whereas Conklin and Anagnostopoulou (2001) say that any viewpoint, including a linked one, can form the basis of a threaded viewpoint, in this research only primitive viewpoints are threaded; but they may be linked with unthreaded viewpoints, which gives added flexibility. For example, (Interval $\otimes$ ScaleDegree) $\ominus$ FirstInPhrase would previously have been equivalent to (Interval $\ominus$ FirstInPhrase) $\otimes$ (ScaleDegree $\ominus$ FirstInPhrase), where the pitch interval is from the from the previous first event in phrase. This implicitly rules out `Interval` $\otimes$ (`ScaleDegree` $\ominus$ `FirstInPhrase`), which, while still defined only at first in phrase, makes use of the immediately preceding (undefined) event in the formation of the pitch interval. Both of these options are allowable in this research. Finally, Conklin and Witten (1995) associate a timescale (in practice, `IOI`) with threaded viewpoints. Conklin and Anagnostopoulou (2001) state that the threaded viewpoint domain is the cross product of the base viewpoint and `IOI` domains. As Pearce (2005) points out, this is the same as the base viewpoint linked with `IOI` (bearing in mind, however, that it is defined at only certain places in the sequence). In this research, however, a timescale is not considered to be necessary and is omitted. One reason for this is that a maximum of three basic attributes are predicted (`Duration`, `Cont` and `Pitch`), with all other basic attributes given; it is therefore unnecessary for a timescale to be used in the prediction of, for example, phrase boundaries. Another reason is that whereas a timescale could be used to align threaded elements with those in a basic sequence, its use can be avoided by keeping track of alignment in a solution array.

3.2.4.3 Representation

In order to think about how linked viewpoints for the modelling of harmony may be constructed, it is useful in the first instance to visualise music in three dimensions (see Figure 3.3). This representation is an aid to conceptualisation which nicely illustrates some of the terminology introduced below (it is rather unwieldy, however; therefore a more compact representation is presented later). The soprano, alto, tenor and bass parts are shown going down the page, as is customary in scores. Going across the page are musical sequences in the time dimension. The various viewpoint types are shown going into the page (in Figure 3.3, only a small subset of viewpoints is shown); therefore each part can be seen as a *layer*.[4] The soprano line is given, and now the bass notes are being generated from left to right. Question marks indicate viewpoint elements which have not yet been generated. The layer in which prediction or generation is occurring at any particular time is referred to as the *prediction* layer, and the others are *support* layers which provide additional context (*cf.* Allan, 2002). Prediction could occur on more than one layer at a time; for example, generating alto and tenor notes together would match the common practice of human composers. For ease of explanation, however, further discussion will assume a single prediction layer unless otherwise stated.

For modelling melody alone, an example of a linked viewpoint is `Pitch`$\otimes$`ScaleDegree` (recall that linked viewpoints model interactions between primitive viewpoints; see §2.2.4). For the modelling of harmony, we need to differentiate between viewpoints in different layers; therefore with respect to the soprano layer, this viewpoint is designated $(\texttt{Pitch} \otimes \texttt{ScaleDegree})_S$. Linking within a single layer will be referred to as

[4] The term *layer* is more general than *part*, since additional layers for viewpoints common to or relying on all four parts (*e.g.*, harmonic function symbols) can be envisaged.

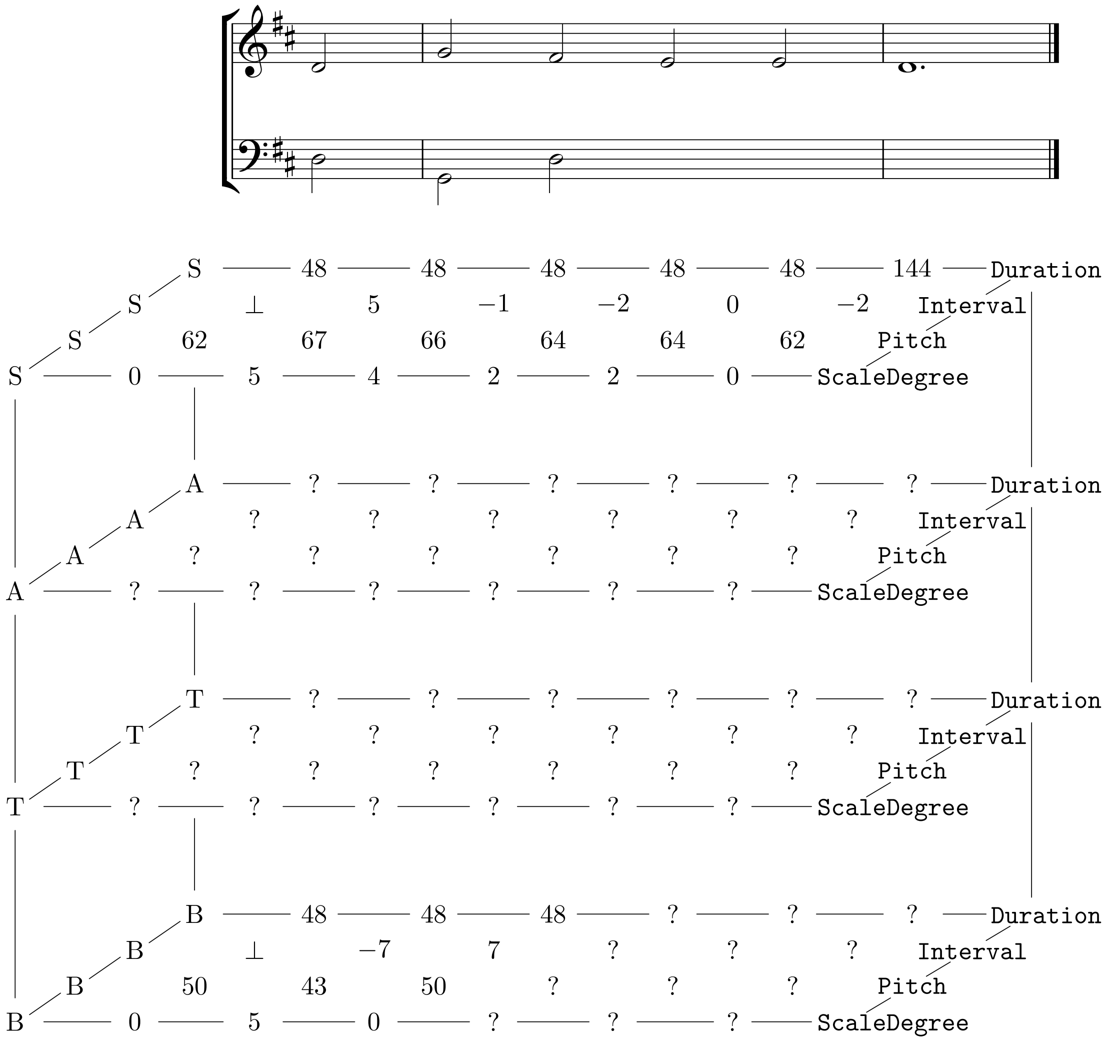

S 48 48 48 48 48 144 Duration
S ⊥ 5 −1 −2 0 −2 Interval
S 62 67 66 64 64 62 Pitch
S 0 5 4 2 2 0 ScaleDegree
A ? ? ? ? ? ? Duration
A ? ? ? ? ? ? Interval
A ? ? ? ? ? ? Pitch
A ? ? ? ? ? ? ScaleDegree
T ? ? ? ? ? ? Duration
T ? ? ? ? ? ? Interval
T ? ? ? ? ? ? Pitch
T ? ? ? ? ? ? ScaleDegree
B 48 48 48 ? ? ? Duration
B ⊥ −7 7 ? ? ? Interval
B 50 43 50 ? ? ? Pitch
B 0 5 0 ? ? ? ScaleDegree

Figure 3.3: A three-dimensional representation of a partial harmonisation of the final phrase of hymn tune *Tallis' Ordinal* (Vaughan Williams 1933, hymn no. 453).

intra-layer linking. We now introduce the concept of *inter-layer* linking; this is a necessary innovation, since clearly there are dependencies between the parts. Two or more intra-layer linked viewpoints (or primitive viewpoints on particular layers) may be linked together in a completely conventional way to produce an inter-layer linked viewpoint. The prediction layer may be explicitly indicated by placing a "p" after the S, A, T or B as appropriate, as exemplified by:

$$(\texttt{Pitch} \otimes \texttt{ScaleDegree})_S \otimes (\texttt{Interval} \otimes \texttt{ScaleDegree})_{Bp}.$$

An example of a multiple viewpoint system for the prediction of `Duration` and `Pitch` in the bass part, based on the primitive viewpoints shown in Figure 3.3, is:

$$\begin{aligned}\{(\texttt{Pitch})_{Bp}, (\texttt{Duration})_{Bp}, (\texttt{Duration} \otimes \texttt{Interval})_{Bp},\\ (\texttt{Interval})_S \otimes (\texttt{Duration} \otimes \texttt{ScaleDegree})_{Bp},\\ (\texttt{Pitch} \otimes \texttt{ScaleDegree})_S \otimes (\texttt{Interval} \otimes \texttt{ScaleDegree})_{Bp}\}.\end{aligned}$$

A different viewpoint representation adapted from Pearce (2005) is presented now (see Figure 3.4) for the following reasons:

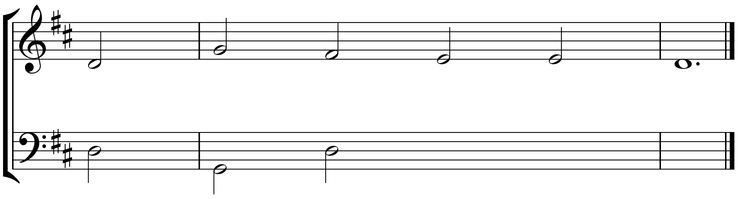
1. It flattens the three-dimensional representation into two dimensions, which makes it more comprehensible.

2. It clearly differentiates between basic (event space) viewpoints and derived viewpoints.

3. It clearly shows how intra-layer and inter-layer linked viewpoints are formed from primitive viewpoints (*i.e.*, basic and derived viewpoints).

Viewpoint names in the example multiple viewpoint system above are shown in bold font. Notice that in the sequence of inter-layer linked viewpoint elements, the ones yet to be predicted contain an intra-layer linked viewpoint which is already known. This is unusual in Markov modelling, and is discussed in the section on version 1 of the developing multiple viewpoint framework below.

As the multiple viewpoint framework is developed, it will be seen that contexts used in models become increasingly complex, leading to potential problems with context matching. It is necessary to be precise about how contexts are represented. If, for example, a context is represented as a compound symbol, inadvertent errors in the construction of the symbol (such as the swapping of two constituent symbols) could result in a match being missed, or in a false match. We therefore introduce a formal set notation for contexts, such that context matching can be determined by set equality. Each element in the set is represented in the following way: ⟨layer, relative chord position, intra-layer linked viewpoint, symbol tuple⟩. Note that a primitive viewpoint is treated as a special case of a linked viewpoint, and that the symbol tuple comprises



soprano

Duration	48	48	48	48	48	144
Pitch	62	67	66	64	64	62
Interval	⊥	5	−1	−2	0	−2
ScaleDegree	0	5	4	2	2	0
Pitch ⊗ ScaleDegree	⟨62, 0⟩	⟨67, 5⟩	⟨66, 4⟩	⟨64, 2⟩	⟨64, 2⟩	⟨62, 0⟩
Duration ⊗ ScaleDegree	⟨48, 0⟩	⟨48, 5⟩	⟨48, 4⟩	⟨48, 2⟩	⟨48, 2⟩	⟨144, 0⟩

bass

(**Duration**)$_{\mathbf{Bp}}$	48	48	48	?	?	?
(**Pitch**)$_{\mathbf{Bp}}$	50	43	50	?	?	?
Interval	⊥	−7	7	?	?	?
ScaleDegree	0	5	0	?	?	?
Interval ⊗ ScaleDegree	⊥	⟨−7, 5⟩	⟨7, 0⟩	⟨?, ?⟩	⟨?, ?⟩	⟨?, ?⟩
Duration ⊗ ScaleDegree	⟨48, 0⟩	⟨48, 5⟩	⟨48, 0⟩	⟨?, ?⟩	⟨?, ?⟩	⟨?, ?⟩
(**Duration ⊗ Interval**)$_{\mathbf{Bp}}$	⊥	⟨48, −7⟩	⟨48, 7⟩	⟨?, ?⟩	⟨?, ?⟩	⟨?, ?⟩

inter-layer

(**Interval**)$_{\mathbf{S}}$ ⊗(**Duration ⊗ ScaleDegree**)$_{\mathbf{Bp}}$	⊥	⟨⟨5⟩, ⟨48, 5⟩⟩	⟨⟨−1⟩, ⟨48, 0⟩⟩	⟨⟨−2⟩, ⟨?, ?⟩⟩	⟨⟨0⟩, ⟨?, ?⟩⟩	⟨⟨−2⟩, ⟨?, ?⟩⟩
(**Pitch ⊗ ScaleDegree**)$_{\mathbf{S}}$ ⊗(**Interval ⊗ ScaleDegree**)$_{\mathbf{Bp}}$	⊥	⟨⟨67, 5⟩, ⟨−7, 5⟩⟩	⟨⟨66, 4⟩, ⟨7, 0⟩⟩	⟨⟨64, 2⟩, ⟨?, ?⟩⟩	⟨⟨64, 2⟩, ⟨?, ?⟩⟩	⟨⟨62, 0⟩, ⟨?, ?⟩⟩

Figure 3.4: A better representation (adapted from Pearce, 2005) of a partial harmonisation of the final phrase of hymn tune *Tallis' Ordinal* (Vaughan Williams 1933, hymn no. 453).

as many symbols as there are primitive viewpoints in the intra-layer linked viewpoint. Relative chord position assumes prediction at chord 0. Consider the following simple N-gram, using only `Pitch` symbols from the bass layer, where the context is shown in blue and the prediction in red:

$$\begin{array}{cccc} 50 & 43 & 50 & ? \\ n-3 & n-2 & n-1 & n \end{array}$$

The context is formally represented as follows:

$$\{\langle \text{bass},\ -3,\ \texttt{Pitch},\ \langle 50\rangle\rangle,\ \langle \text{bass},\ -2,\ \texttt{Pitch},\ \langle 43\rangle\rangle,\ \langle \text{bass},\ -1,\ \texttt{Pitch},\ \langle 50\rangle\rangle\}.$$

3.3 The Modelling of Melody

A Common Lisp implementation of multiple viewpoints and PPM for the modelling of melody was undertaken first, partly as a stepping-stone towards implementation of the framework for modelling harmony, and partly to experiment with the unusually large pool of viewpoints in the modelling of melody. This implementation is called version 0, as it precedes harmonic versions beginning with 1.

Given a melodic data set, a multiple viewpoint system and some other input parameters, it is possible to predict attributes `Duration` and `Pitch` together, `Duration` alone or `Pitch` alone (all other attributes are assumed to be given). In the case of `Duration` and `Pitch` together, for each note in turn `Duration` is predicted first, followed by `Pitch`. This staged prediction technique was developed by Conklin (1990), motivated by the fact that there are computational complexity problems associated with the simultaneous prediction of attributes.

Overall models comprising both long- and short-term models, LTM only or STM only may be specified; there are no subtask models. The LTM has the option of remaining static throughout prediction or being updated after the prediction of every note. PPM using escape method C and exclusion has been implemented as described in §3.2.4.1. The maximum order of the N-gram models is an input parameter.

Two types of viewpoint prediction combination method have been implemented: weighted arithmetic (Conklin, 1990) and weighted geometric (Pearce, 2005). See §2.2.4 for more details. A parameter called a *bias* is used in the calculation of weights; a bias of 0 is equivalent to unweighted combination. Previous research has restricted bias to a small set of integer values; it is anticipated that a small increase in performance can be achieved by removing this restriction. At runtime, separate biases are specified for distribution combination within LTM and STM and for combining the LTM and STM. The biases can be separately automatically optimised for a given multiple viewpoint system. This is a static, off-line optimisation which selects the bias resulting in the lowest cross-entropy for a ten-fold cross-validation of the corpus. Limitations to this implementation mean that the same combination method and bias are used within the

LTM and STM, and the same combination method is used both within and between the LTM and STM. More flexibility in this area will be introduced in the future.

Output files are created during each run for record-keeping purposes. The files include the input parameters which produced the results. Discussion of viewpoint selection is deferred until §3.4.5. This implementation is also capable of generating novel melodies by random sampling of the models; this is described in detail in Chapter 9.

3.4 Development of the Multiple Viewpoint Framework for Harmony

Outlined below are three ways in which the multiple viewpoint framework has been developed in order to cope with the complexities of harmony, starting with the strictest possible application of viewpoints, and then gradually extending and generalising. A further six versions (making nine in all) are expounded in Chapter 10. These increasingly complex developments of the multiple viewpoint and PPM frameworks will be the subject of future research.

3.4.1 Version 1: Strict Application of Multiple Viewpoints and PPM

The starting point for the definition of the strictest possible application of viewpoints to harmony is the formation of *vertical viewpoint elements* as described in §2.5.3 (Conklin, 2002). An example of such an element (using the notation of Figure 3.4) is $\langle\langle 0\rangle,\ \langle 7\rangle,\ \langle 4\rangle,\ \langle 0\rangle\rangle$, where all of the values are from the domain of the same viewpoint (in this case `Scale-Degree`), and are from the soprano, alto, tenor and bass layers respectively. Similarly, a vertical element for linked viewpoint `Pitch` ⊗ `ScaleDegree` may, for example, be $\langle\langle 62,\ 0\rangle,\ \langle 57,\ 7\rangle,\ \langle 54,\ 4\rangle,\ \langle 50,\ 0\rangle\rangle$. There are two important points here: the same viewpoint is used on each layer, and all of the layers are represented in the vertical viewpoint element. This method reduces the entire set of parallel sequences to a single sequence, thus allowing an unchanged application of the multiple viewpoint framework, including its use of N-grams and PPM (*i.e.*, exactly the same as in melodic version 0).

In order to make better comparisons with later versions, however, it must be made clear that these vertical viewpoint elements are equivalent to inter-layer linked viewpoints as described in §3.2.4.3 above, where each layer uses the same intra-layer linked viewpoint. The linking of viewpoints is done in a completely conventional way. Note that, as usual, if at a point in the event sequence a derived viewpoint is undefined, a linked viewpoint containing that viewpoint is also undefined. This version has the option of predicting "blind" (*i.e.*, in exactly the same way as version 0)[5] or predicting with reference to a given melody. The latter is overwhelmingly the emphasis of this research, however, and we continue on the basis that there is a given melody. This means that

[5] This affords the opportunity to generate melodies while taking account of their harmonic basis.

the soprano is a compulsory support layer and the alto, tenor and bass are prediction layers. If we assume that the primitive viewpoint `ScaleDegree` is used on each layer, the name of the inter-layer linked viewpoint is:

$$(\texttt{ScaleDegree})_S \otimes (\texttt{ScaleDegree})_{Ap} \otimes (\texttt{ScaleDegree})_{Tp} \otimes (\texttt{ScaleDegree})_{Bp}.$$

The following example uses the primitive viewpoint `ScaleDegree` in every layer. The soprano is the support layer, and the alto, tenor and bass are prediction layers. Conventionally, a viewpoint predicts a viewpoint element of the same type(s), resulting in N-grams such as this, which links a known soprano element with alto, tenor and bass elements yet to be predicted (a formal representation of the context is also shown):

$$\begin{array}{cccc}
0 & 5 & 4 & 2 \\
0 & 0 & 0 & ? \\
4 & 9 & 7 & ? \\
0 & 5 & 0 & ? \\
n-3 & n-2 & n-1 & n
\end{array}$$

$$\begin{aligned}
\{\langle \text{soprano},\ -3,\ \texttt{ScaleDegree},\ \langle 0\rangle\rangle,\ \langle \text{soprano},\ -2,\ \texttt{ScaleDegree},\ \langle 5\rangle\rangle,\\
\langle \text{soprano},\ -1,\ \texttt{ScaleDegree},\ \langle 4\rangle\rangle,\ \langle \text{alto},\ -3,\ \texttt{ScaleDegree},\ \langle 0\rangle\rangle,\\
\langle \text{alto},\ -2,\ \texttt{ScaleDegree},\ \langle 0\rangle\rangle,\ \langle \text{alto},\ -1,\ \texttt{ScaleDegree},\ \langle 0\rangle\rangle,\\
\langle \text{tenor},\ -3,\ \texttt{ScaleDegree},\ \langle 4\rangle\rangle,\ \langle \text{tenor},\ -2,\ \texttt{ScaleDegree},\ \langle 9\rangle\rangle,\\
\langle \text{tenor},\ -1,\ \texttt{ScaleDegree},\ \langle 7\rangle\rangle,\ \langle \text{bass},\ -3,\ \texttt{ScaleDegree},\ \langle 0\rangle\rangle,\\
\langle \text{bass},\ -2,\ \texttt{ScaleDegree},\ \langle 5\rangle\rangle,\ \langle \text{bass},\ -1,\ \texttt{ScaleDegree},\ \langle 0\rangle\rangle\}.
\end{aligned}$$

The complete distribution of viewpoint elements which could follow the first three elements would include those which did not have a value of 2 in the soprano. We would not wish to change something which is given or has already been predicted, so only those elements with a value of 2 in the soprano are allowed in the distribution used for prediction or generation (*i.e.*, prediction elements must contain known values from the support layers). A procedure similar to this is already established within the multiple viewpoint framework; prediction of event attribute values is done in sequence, with each successive prediction probability distribution having to be compatible with the previously instantiated attribute values (see §3.3 above). Another way of introducing current information, as part of the context rather than as part of what is being predicted, forms the basis of version 4 (see Chapter 10).

In this version, it is possible to predict attributes `Duration`, `Cont` (see §3.4.4) and `Pitch` (in that order) in any combination. The structure of the overall model, options and parameters are otherwise the same as in version 0.

To summarise, this is a weighted (PPM) model employing the strictest possible application of viewpoints, which is achieved by combining four event sequences into a single sequence. The i^{th} elements of the prediction layers are linked with the i^{th} element of the support layer. Within any inter-layer linked (vertical) viewpoint, all intra-layer linked viewpoints must be the same. Prediction elements must contain the known value from the soprano support layer. Various combination techniques are possible within PPM for the determination of complete probability distributions; but the one chosen for this research is back-off smoothing using escape method C and exclusion. This is the base-level version, similar to that used by Conklin (2002), which is developed below with the objective of improving prediction performance.

3.4.2 Version 2: Breaking Up the Harmonisation Task into Subtasks

In this version, it is hypothesised that predicting all unknown symbols in an inter-layer linked viewpoint element (as in version 1) at the same time is neither necessary nor desirable. It is expected that by dividing the overall harmonisation task into a number of subtasks, each modelled by its own multiple viewpoint system, an increase in performance can be achieved. The introduction of subtasks to the multiple viewpoint framework is novel to this research, in which a subtask is the prediction or generation of at least one layer. For example, given a soprano line, the first subtask might be to predict the entire bass line. There is no supporting context in the alto and tenor layers yet; therefore these are ignored, giving rise to N-grams formed from a soprano support layer and a bass prediction layer such as this (the viewpoint name is also shown):

$$\begin{array}{cccc} 0 & 5 & 4 & 2 \\ 0 & 5 & 0 & ? \\ n-3 & n-2 & n-1 & n \end{array}$$

$$(\texttt{ScaleDegree})_S \otimes (\texttt{ScaleDegree})_{Bp}.$$

As in version 1, inter-layer linked viewpoints are restricted to using the same intra-layer linked viewpoint on each layer. The difference is that not all of the layers are now necessarily represented in an inter-layer linked viewpoint (although all available support layers are represented). At this stage, it is possible to experiment with different arrangements of subtasks. For example, having predicted the entire bass line, is it better to predict the alto and tenor lines together, or one before the other? Note that in order to be more useful as a cognitive model in future work, there would need to be an option to predict all of the notes of a chord by the application of a sequence of subtask models before moving on to the next chord.

Assuming that the bass line has been predicted, and that we are now predicting alto and tenor together, N-grams are formed from soprano and bass support layers and alto and tenor prediction layers:

$$\begin{array}{cccc}
0 & 5 & 4 & 2 \\
0 & 0 & 0 & ? \\
4 & 9 & 7 & ? \\
0 & 5 & 0 & 7 \\
n-3 & n-2 & n-1 & n
\end{array}$$

$$(\texttt{ScaleDegree})_S \otimes (\texttt{ScaleDegree})_{Ap} \otimes (\texttt{ScaleDegree})_{Tp} \otimes (\texttt{ScaleDegree})_B.$$

Notice that in this case there are two known values in the prediction, which further restricts the size of the prediction probability distribution.

The multiple viewpoint systems for each subtask are separately optimised using viewpoint selection (see §3.4.5). The structure of the subtasks may be different (*e.g.*, one may be LTM only, and the other a combination of LTM and STM). In addition, model order, viewpoint distribution combination method, biases and the parameter indicating whether or not the LTM is updated can be specified differently for each subtask. This flexibility affords the possibility of further optimisation.

To summarise, this is a weighted (PPM) model which allows linking between any number of layers. The i^{th} element(s) of the prediction layer(s) is/are linked with the i^{th} element(s) of the support layer(s). Within any inter-layer linked (vertical) viewpoint, all intra-layer linked viewpoints must be the same.

3.4.3 Version 3: Inter-layer Linking of Different Intra-layer Linked Viewpoints

There are two differences between version 2 and version 3, both of which are novel to this research. The first is that different viewpoints on different layers can now be linked; for example, `ScaleDegree` in the soprano (support) layer can be inter-layer linked with `Pitch` in the bass (prediction) layer, resulting in N-grams such as this (the viewpoint name and a formal representation of the context are also shown):

$$\begin{array}{cccc}
0 & 5 & 4 & 2 \\
50 & 43 & 50 & ? \\
n-3 & n-2 & n-1 & n
\end{array}$$

$$(\texttt{ScaleDegree})_S \otimes (\texttt{Pitch})_{Bp}$$

$$\begin{aligned}
\{&\langle \text{soprano},\ -3,\ \texttt{ScaleDegree},\ \langle 0\rangle\rangle,\ \langle \text{soprano},\ -2,\ \texttt{ScaleDegree},\ \langle 5\rangle\rangle,\\
&\qquad\langle \text{soprano},\ -1,\ \texttt{ScaleDegree},\ \langle 4\rangle\rangle,\ \langle \text{bass},\ -3,\ \texttt{Pitch},\ \langle 50\rangle\rangle,\\
&\qquad\qquad\langle \text{bass},\ -2,\ \texttt{Pitch},\ \langle 43\rangle\rangle,\ \langle \text{bass},\ -1,\ \texttt{Pitch},\ \langle 50\rangle\rangle\}.
\end{aligned}$$

The second is that linking with support layers (given parts) is not compulsory; so if we are given the soprano and bass, and are predicting the alto and tenor, we can, for instance, link viewpoints from the alto, tenor and bass, but not the soprano. It should be noted, however, that even if a support layer is not represented in a linked viewpoint, the domain is still constrained by the given note at the prediction point of this layer.

At present, to avoid too much complexity, prediction layers are assigned the same viewpoint; although support layers may have any combination of viewpoints. A relaxation of the prediction layer restriction could conceivably result in better (but more complex) models; so a sub-version without this restriction may be considered in the future. Note that, as usual, if at a point in the event sequence a derived viewpoint is undefined, a linked viewpoint containing that viewpoint is also undefined (see the first event of Figure 3.4).

Finally, since version 3 viewpoint selection running times have proven to be much longer than those of other versions, a subset of only twenty primitive and threaded viewpoints has been implemented (usefully also reducing software development time). The viewpoints, chosen primarily because of their utility with respect to versions 0 to 2 in preliminary runs, are: `Duration`, `DurRatio`, `Cont`, `Pitch`, `Interval`, `ScaleDegree`, `ScaleDegree` ⊖ `Tactus`, `ScaleDegree` ⊖ `FirstInPhrase`, `ScaleDegree` ⊖ `LastInPhrase`, `Tactus`, `PositionInBar`, `TactusPositionInBar`, `FirstInBar`, `Metre`, `Phrase`, `FirstInPhrase`, `LastInPhrase`, `Piece`, `FirstInPiece` and `LastInPiece`. It is expected that the increased flexibility with respect to linking will more than make up for this smaller pool of viewpoints.

To summarise, this is a weighted (PPM) model which allows linking between any number of layers. The i^{th} element(s) of the prediction layer(s) is/are linked with the i^{th} element(s) of the support layer(s). Within any inter-layer linked (vertical) viewpoint, intra-layer linked viewpoints need not be the same (except on prediction layers) and support layers do not have to be represented.

A further six increasingly complex ways of applying the multiple viewpoint and PPM frameworks to the harmonisation task are outlined in §10.3. Table 3.2 lists features introduced into the multiple viewpoint framework for harmony, and shows how they ideally relate to the different model versions. From a time complexity point of view, however (see Chapter 4), it may be necessary to implement a subset of the indicated features for the more complex versions.

3.4.4 Full Expansion and Viewpoint `Cont`

Basic viewpoint `Cont` is introduced in this research specifically for use in the modelling of harmony: it is required because the corpus (like music in general) is not completely homophonic. We require *simultaneities* (concurrent events) to have a single value for each part, and a single duration overall; therefore *full expansion* is used to partition music in this way. This technique has already been used in conjunction with viewpoints (Conklin, 2002). Figure 3.5 shows the first three full bars of the harmonisation of hymn tune *Caton* (or *Rockingham*) in the top system, and its fully expanded form in the

	Version								
Feature	1	2	3	4	5	6	7	8	9
Inter-layer linked viewpoints may use any number and combination of layers.		✓	✓	✓	✓	✓	✓	✓	✓
Different intra-layer linked viewpoints may be used on each layer.			✓	✓	✓	✓	✓	✓	✓
Inter-layer linking of prediction layer(s) with support layer(s) is offset by one element.				✓	*	*	*		
Future context may be used, where appropriate.					✓	✓	✓	✓	✓
Models are arranged in a back-off sequence according to their cross-entropy.						✓	✓	✓	✓
Different N-gram models with the same context size may appear in a back-off sequence.							✓	✓	✓
A more flexible inter-layer linking method is introduced.								✓	✓
A more flexible intra-layer linking method is introduced.									✓

Table 3.2: A list of features introduced into the multiple viewpoint framework for harmony, and how they relate to versions 1 to 9. Baseline version 1 is compelled to use SATB in all viewpoints (similar to melodic version 0, which uses only soprano notes). An asterisk indicates that a feature is optional.

bottom system. Looking at the first full bar, we see that the semibreve D in the alto has been split into two minims; and that where the tenor moves in crotchets from A to G, the minims in the other three parts have been split into crotchets. To model harmony correctly, we need to know which notes have been split in this way, and which have not. To distinguish between the start of a note and its continuation, Assayag et al. (1999) used different pitch symbols; for example, 'b' for the start of a note and '**b**' (in bold font) for its continuation. Our preferred solution is viewpoint type `Cont`, which obviates the need to further increase the size of the `Pitch` domain (or any other viewpoint domain, since `Cont`, like any other basic viewpoint, is an attribute of a note as a whole). This type has the value T if a note is a continuation of the previous one in the same part (*i.e.*, it has the same pitch, and is not re-sounded), and the value F otherwise; therefore the vertical $(\texttt{Cont})_{SATB}$ elements for the first full bar of Figure 3.5 are

$$\langle F,\ F,\ F,\ F\rangle\ \langle F,\ T,\ F,\ F\rangle\ \langle F,\ F,\ F,\ F\rangle\ \langle T,\ T,\ F,\ T\rangle.$$

Generation is slightly different from prediction, inasmuch as, in this implementation, the melody to be harmonised is not already expanded; therefore it must be expanded during generation. Whenever a `Duration` value shorter than the soprano note is generated, the `Duration` value of the soprano note is changed to the generated one, and a new soprano note is added directly following the note being harmonised. This is the same

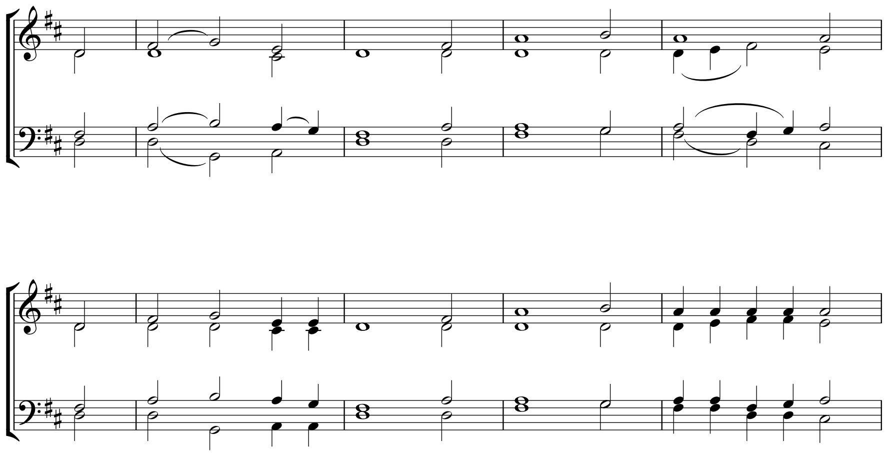


Figure 3.5: The first four full bars of the harmonisation of hymn tune *Caton* (or *Rockingham*), adapted from Vaughan Williams (1933), are presented in the top system, and the full expansion of this excerpt is shown in the bottom system.

as the preceding note except that its duration is equal to the original duration minus the new duration of the preceding note; and the added note is always a continuation (whereas the previous note may or may not be a continuation). Unfortunately, this leads to a problem at phrase and piece endings, which are indicated by a '−1' in the corpus and test data. When generating a harmonisation to a given unexpanded melody, it is possible that, for example, viewpoint `LastInPhrase` has contributed to the prediction of `Duration` for the vertical element corresponding to the start of the last soprano note of the phrase. If `Duration` is less than the length of the soprano note, however, the soprano note is expanded, and the last in phrase '−1' moves from the start of the soprano note to its continuation. We then generate a harmonisation for the soprano continuation on the basis that the previous vertical element is not at the end of the phrase. This seems unsatisfactory. One way of overcoming this problem is to supply expanded soprano parts for harmonisation, preferably created using a separate multiple viewpoint system. This may be feasible, bearing in mind that there are unexpanded and expanded melodies in the corpus that could be learned from. Another possibility is to place a '−1' at the beginning of the last note of the phrase in each part, and in all of its continuations, rather than in the very last vertical element of the phrase. These suggestions must, unfortunately, be left for future work.

3.4.5 Viewpoint Selection

3.4.5.1 The Search Space

Although viewpoints `Onset`, `Pulses`, `KeySig` and `Tonic` have been implemented, they are primarily seen as supporting the implementation of other viewpoints, and so are not tried during viewpoint selection. This leaves 39 primitive viewpoints which are available for use, in both linked and unlinked form, in multiple viewpoint systems. If we allow unlimited linking of viewpoints, the total number of linked and unlinked viewpoints is exponential in the number of primitive viewpoints. In turn, the number of possible multiple viewpoint systems is exponential in the total number of viewpoints. In fact, the total number of linked and unlinked viewpoints is about 5.5×10^{11}, and the number of possible multiple viewpoint systems is approximately $10^{1.7 \times 10^{11}}$. This extraordinarily large search space must be shrunk: one way of doing this is to allow linking between only 2 primitive viewpoints (applicable to versions 0, 1 and 2), which is an approach taken in previous research (Conklin and Witten, 1995; Pearce, 2005). This brings the total number of viewpoints down to only 780; but even with this relatively small number of viewpoints, the number of possible multiple viewpoint systems is still about 10^{235}.

In practice, things are not quite this bad because only 21 of the primitive viewpoints can predict `Duration`, `Cont` and `Pitch`; linked and unlinked viewpoints which are unable to predict these basic attributes can be identified in advance and ignored. It happens that 609 viewpoints are capable of making predictions; therefore the number of possible multiple viewpoint systems reduces to about 10^{183}.

3.4.5.2 Exploring the Search Space

Assuming that it takes on average only a second to evaluate a single multiple viewpoint system (an underestimate), evaluating 10^{183} systems to find the globally best system would take about 3.2×10^{175} years, which is far more time than the universe has so far existed. Clearly, the best we can hope for is to find a locally optimal system by using a viewpoint selection algorithm such as the one described in §2.2.4 (Pearce, 2005), which is based on *forward stepwise selection*. This algorithm reduces the search space considerably by making the general assumption that the best n viewpoint system is a subset of the best $n + 1$ viewpoint system. Bearing in mind that there is a pool of 609 possible viewpoints, however, the algorithm must be made more time efficient. Pearce (2005) was able to evaluate the addition of every viewpoint in the pool at each iteration of the algorithm by virtue of the fact that the pool then comprised only 54 viewpoints; but now a means of limiting the number of viewpoints tried at each round of addition is a practical necessity, especially for more complex versions of the framework.

Recall from §2.5.3.3 the Markov random field induction procedure described by Pickens and Iliopoulos (2005). To begin with, there is a uniform structure comprising individual onsets without dependencies. At each stage of feature addition, the set of candidate features is defined as the set of one onset extensions to the current structure, where the additional onset is not more than two simultaneities from the original. The structure of the field is incrementally modified by adding the candidate feature which results in the greatest improvement in the log-likelihood of training data. The key difference between this procedure and the one described by Pearce (2005) is that only extensions to features present in the structure are tried in the former case. This seems like a reasonable approach, bearing in mind that it is analogous to the approach we have already adopted at the multiple viewpoint system construction level; at both feature (viewpoint) level and system level, incremental augmentation of what has already been selected is the guiding principle.

The greatest improvement in time efficiency is likely to be made by adopting the following approach, termed VS1: at each iteration, try only primitive viewpoints, and viewpoints already in the multiple viewpoint system (viewpoint set) linked with one other primitive viewpoint. A less time efficient approach, termed VS2, recognises that primitive viewpoints added to the viewpoint set are worth keeping under consideration with respect to linking with other viewpoints, even if those primitive viewpoints are later removed from the set. For completeness, this distinction applies to basic viewpoints included by default in the initial viewpoint set. The least time efficient approach, termed VS3, further recognises that links with specified primitive viewpoints which have not necessarily been added to the viewpoint set are worth keeping under consideration with respect to linking with other viewpoints.

One more issue needs to be addressed with respect to the viewpoint selection algorithm: as described by Pearce (2005), it is concerned with the construction of a system which predicts only one attribute. In order to use it in situations where more than one attribute is being predicted, the algorithm must be slightly adjusted. Instead of starting with an empty viewpoint set, the initial set should contain the basic viewpoints for all of the attributes to be predicted; this is necessary so that event probabilities (the product of basic attribute probabilities) can always be calculated for evaluation purposes. The algorithm starts in addition mode in exactly the same way as before. When in deletion mode, however, one of the stipulations must be that is not permitted to delete a viewpoint which would result in one of the basic attributes not being predicted.

Version 3 requires a further modification to the viewpoint selection algorithm. In this case, the initial viewpoint set comprises basic viewpoints on the prediction layer(s) only. To begin with the existing algorithm is followed, except that only the prediction layer(s) is/are involved.[6] Once all of these possibilities have been exhausted, primitive viewpoints are separately added to the support layer(s). If, in the end, an inter-layer linked viewpoint with a primitive viewpoint in a support layer is chosen, during the next round of addition links with this primitive viewpoint are also tried.

Let us now consider how best to evaluate candidate multiple viewpoint systems during viewpoint selection. One way of doing it is to use the training corpus for evaluation purposes. The problem with this is that the chosen multiple viewpoint system is too specialised: it is much less good at predicting unseen data than it is at predicting the corpus. Similarly, evaluating using a set of data which is not in the corpus tends to specialise to that data. The best way to ensure generalisation to unseen data is to perform an N-fold cross-validation of the corpus. In this research the corpus is divided into ten

[6] The prediction probability distributions are still constrained by the given values, however; it is only in the contexts that the support layers are completely ignored.

parts, and each of the ten parts is predicted in turn using a model learned from the other nine parts. The mean of the ten calculated cross-entropies is used to compare the relative performance of different candidate multiple viewpoint systems.

Finally, when selecting for the best combined long- and short-term model, the overall best multiple viewpoint system is sought (*i.e.*, the same system is used in both the LTM and the STM).

3.4.5.3 Speeding Up the Search

A great deal of logic has been incorporated into the implementation of the viewpoint selection algorithm to ensure that:

- only multiple viewpoint systems capable of fully predicting all notes are evaluated;
- viewpoints are not linked with themselves;
- the same viewpoint is not used more than once in a system;
- the simplest of effectively equivalent viewpoints is chosen;
- and that evaluation of previously evaluated systems is avoided.

When using the algorithm as described in §3.4.5.2, the set of viewpoints tried during an addition round is largely the same as that tried in the previous such round, irrespective of how well the individual viewpoints performed. Viewpoint selection has been speeded up by removing from further consideration any viewpoint which, when added to a system, increases the cross-entropy above a certain margin. If we define δ as the improvement in cross-entropy over the last round of additions and deletions, a margin equal to the smaller of 0.04 or 1.7δ bits/symbol was settled upon by trial and error (*i.e.*, the margin is based on the generally reducing improvement at each stage). At this level, the resulting systems were the same as those optimised without removing viewpoints from consideration (although, of course, there is no guarantee that this will always be the case). Many viewpoints are, in practice, eliminated in this way during viewpoint selection, resulting in shorter runs.

During addition, different viewpoints are added to the previous best multiple viewpoint system for evaluation. A great deal of time is saved by caching lists of distributions from the previous best system at the beginning of each round of addition, so that they can be recalled from memory rather than recalculated. These distributions are then combined along with that of each candidate viewpoint in turn. Something very similar is done for each round of deletion.

It has been found in practice that at most one of all the possible deletions results in a reduction in cross-entropy; therefore to save time, as soon as a deletion is found which improves the model, the deletion round ends. Finally, the software implementation saves state at the end of each deletion round so that if, for any reason, a viewpoint selection run is interrupted, it does not have to be restarted from scratch. This is particularly useful for the very long runs of version 3.

3.4.5.4 Viewpoint Selection Prodedure

A high level algorithm for the selection of viewpoints can be found in Algorithm 3.1. This algorithm will be illustrated, in the first instance, by a fictitious version 0 example using option VS3. To keep things simple, the available pool of primitive viewpoints is restricted to {`Duration`, `Pitch`, `ScaleDegree`} and only `Pitch` is being predicted. We have the following input parameters:

$\textit{version} = 0$
$\textit{option} = \mathrm{VS3}$
$\textit{primitive} = \{\texttt{Duration}, \texttt{Pitch}, \texttt{ScaleDegree}\}$
$\textit{attribute} = \{\texttt{Pitch}\}$
$\textit{additional} = \{\texttt{Duration}\}$
$\textit{const-reject} = 0.04$
$\textit{coeff-reject} = 1.7$
$\textit{prediction-layer} = \{S\}$
$\textit{support-layer} = \{\}$
$\textit{halting-value} = 0.0015.$

Algorithm 3.2 in invoked in line 1.

In the **for** loop beginning on line 1 of Algorithm 3.2, variable *basic-viewpoint* is set to `Pitch`, the only member of array *attribute*. In line 2, variable *viewpoint* becomes $(\texttt{Pitch})_{Sp}$; but since we are only dealing with the soprano layer in this example, we shall simply call it `Pitch`. No viewpoint subscripts will be used for the remainder of the example. Array *best*[0] is the repository for the best performing multiple viewpoint system so far, which in line 3 is set to the initial system {`Pitch`}. In line 4, the array *augmented-system* also also becomes {`Pitch`} in the first instance. The predicate in line 5 is true; therefore line 6 is executed next. In the **for** loop beginning on this line, variable *additional-viewpoint* is set to `Duration`, the only member of array *additional*. In line 7, variable *viewpoint* also becomes `Duration`; and this viewpoint is added to array *augmented-system* in line 8, giving {`Pitch`, `Duration`}. In line 9, array element *best*[1] is assigned the ten-fold cross-validation cross-entropy resulting from the use of system {`Pitch`}, say 4.21 bits/note; and arrays *best*[2] and *best*[3] are initialised for future use in lines 10 and 11. Arrays *best* and *augmented-system* are assigned to *initial*[0] and *initial*[1] respectively in lines 12 and 13; and array *initial* is returned from line 14 to line 1 of Algorithm 3.1.

In line 1 of Algorithm 3.1, array *initial* now contains arrays *best* and *augmented-system*. These arrays are unpacked in lines 2 and 3. Variable *delta* is initialised to a very large number in line 4. In the **do** loop beginning on line 5, array *previous-best-system* is set to {`Pitch`}; variable *previous-min-x-entropy* is set to 4.21 bits/note; array

Algorithm 3.1 High level algorithm for the selection of viewpoints, in the style of Corman et al. (2001). Input parameter *version* takes the value 0, 1, 2 or 3; input parameter *option* takes the value VS1, VS2 or VS3; input parameter *primitive* is an array containing the pool of primitive viewpoints; input parameter *attribute* is an array containing the basic viewpoints to be predicted; input parameter *additional* contains additional user-specified primitive viewpoints for VS3; input parameters *const-reject* and *coeff-reject* are numbers used in the assessment of viewpoints' performance; input parameters *prediction-layer* and *support-layer* are arrays containing partitions of $\{S, A, T, B\}$; and input parameter *halting-value* is a number used when considering termination of the selection process (see §5.2.6). Procedure SIZE() returns the number of elements in an array; procedure GET-PRIMITIVE-VIEWPOINT() constructs a primitive viewpoint with appropriate layers; procedures CHECK-AND-TEST(), TRY-LINKS(), TRY-DELETIONS() and TRY-V3-LINKS() can be found in Algorithms 3.3, 3.5, 3.6 and 3.7, respectively; and procedure UNION() performs the set operation *union*.

```
SELECT-VIEWPOINTS(version, option, primitive, attribute, additional, const-reject,
        coeff-reject, prediction-layer, support-layer, halting-value)
initial $\leftarrow$ GET-INITIAL-SYSTEMS(version, option, attribute, additional, prediction-layer,
        support-layer)
best $\leftarrow$ initial[0]
augmented-system $\leftarrow$ initial[1]
delta $\leftarrow$ 1000
do
    previous-best-system $\leftarrow$ best[0]
    previous-min-x-entropy $\leftarrow$ best[1]
    rejected-viewpoint $\leftarrow$ best[2]
    best[3] $\leftarrow$ initialise array
    for test-primitive $\leftarrow$ primitive[0] to primitive[SIZE(primitive) $-$ 1]
        test-viewpoint $\leftarrow$ GET-PRIMITIVE-VIEWPOINT(version, test-primitive,
                prediction-layer, support-layer)
        best $\leftarrow$ CHECK-AND-TEST(test-viewpoint, previous-best-system,
                previous-min-x-entropy, best, delta, rejected-viewpoint,
                attribute, const-reject, coeff-reject)
    if option = VS1
      then system $\leftarrow$ previous-best-system
      else system $\leftarrow$ augmented-system
    if version = 3
      then best $\leftarrow$ TRY-V3-LINKS(primitive, attribute, previous-best-system,
                previous-min-x-entropy, system, rejected-viewpoint, best, delta,
                const-reject, coeff-reject, prediction-layer, support-layer)
      else best $\leftarrow$ TRY-LINKS(primitive, attribute, previous-best-system,
                previous-min-x-entropy, system, rejected-viewpoint, best,
                delta, const-reject, coeff-reject, prediction-layer)
    augmented-system $\leftarrow$ UNION(augmented-system, best[0])
    best $\leftarrow$ TRY-DELETIONS(best, delta, previous-min-x-entropy, const-reject, coeff-reject)
    delta $\leftarrow$ previous-min-x-entropy $-$ best[1]
until delta $\leq$ halting-value
return best
```

Algorithm 3.2 Algorithm for the construction of initial multiple viewpoint systems, in the style of Corman et al. (2001). Input parameter *version* takes the value 0, 1, 2 or 3; input parameter *option* takes the value VS1, VS2 or VS3; input parameter *attribute* is an array containing the basic viewpoints to be predicted; input parameter *additional* contains additional user-specified primitive viewpoints for VS3; and input parameters *prediction-layer* and *support-layer* are arrays containing partitions of $\{S, A, T, B\}$. Procedure SIZE() returns the number of elements in an array; procedure GET-PRIMITIVE-VIEWPOINT() constructs a primitive viewpoint with appropriate layers; procedure ADD() adds a new element to an array; and procedure X-VALIDATION() carries out a ten-fold cross-validation of the corpus and returns the mean cross-entropy.

```
GET-INITIAL-SYSTEMS(version, option, attribute, additional, prediction-layer, support-layer)
for basic-viewpoint ← attribute[0] to attribute[SIZE(attribute) − 1]
    viewpoint ← GET-PRIMITIVE-VIEWPOINT(version, basic-viewpoint, prediction-layer,
                                         support-layer)
    best[0] ← ADD(viewpoint, best[0])
augmented-system ← best[0]
if option = VS3
  for additional-viewpoint ← additional[0] to additional[SIZE(additional) − 1]
      viewpoint ← GET-PRIMITIVE-VIEWPOINT(version, additional-viewpoint,
                                           prediction-layer, support-layer)
      augmented-system ← ADD(viewpoint, augmented-system)
best[1] ← X-VALIDATION(best[0])
best[2] ← initialise array
best[3] ← initialise array
initial[0] ← best
initial[1] ← augmented-system
return initial
```

rejected-viewpoint is effectively initialised; and array *best*[3] is initialised (lines 6 to 9 respectively). In the **for** loop beginning on line 10, variable *test-primitive* is set to `Duration`, the first element of array *primitive*. In line 11, variable *test-viewpoint* is also set to `Duration`; and Algorithm 3.3 is invoked in line 12.

The predicate in line 1 of Algorithm 3.3 is false, since `Duration` is unable to predict `Pitch`; therefore line 8 is executed next, where unchanged array *best* is returned to line 12 of Algorithm 3.1. We run through the loop beginning on line 10 again, using primitive viewpoint `Pitch`. This time, the predicate in line 1 of Algorithm 3.3 is true; but the one in line 2 is also true, and so array *best* is returned unaltered again. We traverse the loop once more using `ScaleDegree`. The predicate in line 1 of Algorithm 3.3 is true, and the ones in lines 2 to 4 are false. This means that array *test-system* becomes {`Pitch`, `ScaleDegree`} in line 5. Array *best*[3] (the repository for viewpoints tried during this addition stage) is set to {`ScaleDegree`} in line 6; and Algorithm 3.4 is invoked in line 7.

In line 1 of Algorithm 3.4, variable *test-x-entropy* is assigned the cross-entropy due to system {`Pitch`, `ScaleDegree`}, say 3.46 bits/note. The predicate in line 2 is true; therefore lines 3 and 4 are executed next, where array *best*[0] becomes {`Pitch`, `ScaleDegree`} and variable *best*[1] is assigned the value 3.46 bits/note (the alternative in lines 5 to 8 can easily be understood on inspection). The predicate in line 9 is true, but the one in line 10 is false; therefore line 12 is executed next, where variable *best* is returned to line 7 of Algorithm 3.3. It is then immediately returned from line 8 to line 12 of Algorithm 3.1.

The predicate in line 13 of Algorithm 3.1 is false; therefore line 15 is the next to be executed, where array *system* becomes {`Pitch`, `Duration`}. The predicate in line 16 is false; therefore line 18 is executed next, where Algorithm 3.5 is invoked.

In the **for** loop beginning on line 1 of Algorithm 3.5, variable *system-viewpoint* becomes `Pitch`, which is the first element of array *system*. The predicate in line 2 is true; therefore the loop beginning on line 3 is executed, where variable *test-primitive* is set to `Duration` in the first instance. Variable *test-link* becomes `Duration` ⊗ `Pitch` in line 4; and Algorithm 3.3 is invoked in line 5.

The predicate in line 1 of Algorithm 3.3 is true, and the predicates in lines 2 to 4 are false; therefore lines 5 to 7 are executed. Array *test-system* becomes {`Pitch`, `Duration`⊗`Pitch`} in line 5; array *best*[3] is set to {`ScaleDegree`, `Duration`⊗`Pitch`} in line 6; while Algorithm 3.4 is invoked in line 7.

In line 1 of Algorithm 3.4, variable *test-x-entropy* is assigned the value 4.28 bits/note for system {`Pitch`, `Duration` ⊗ `Pitch`}. This time, the predicate in line 2 is false, and so line 5 is executed next. The predicate in this line is also false; therefore we jump to line 9. The predicates in lines 9 and 10 are both true, and so in line 11 array *best*[2] (the repository for rejected viewpoints) becomes {`Duration` ⊗ `Pitch`}. Variable *best* is returned from line 12 to line 7 of Algorithm 3.3, and then immediately returned from line 8 to line 5 of Algorithm 3.5.

Algorithm 3.3 Algorithm for checking and testing viewpoints in a multiple viewpoint system, in the style of Corman et al. (2001). Input parameter *test-viewpoint* is the viewpoint under consideration for addition to the multiple viewpoint system; input parameter *previous-best-system* is an array of selected viewpoints from the previous addition and deletion loop (see Algorithm 3.1); and input parameter *previous-min-x-entropy* is its associated ten-fold cross-validation cross-entropy. Input parameter *best* is an array containing the current best system and its associated ten-fold cross-validation cross-entropy; this array is also a convenient repository for rejected viewpoints and viewpoints already tried during an addition stage (*best*[3]). Input parameter *delta* is the reduction in cross-entropy achieved in the previous viewpoint addition and deletion loop; input parameter *rejected-viewpoint* is an array of viewpoints no longer considered for selection; input parameter *attribute* is an array containing the basic viewpoints to be predicted; and input parameters *const-reject* and *coeff-reject* are numbers used in the assessment of viewpoints' performance. Procedure VALID() determines whether or not a primitive viewpoint is able to predict a relevant attribute (*e.g.*, `LastInPhrase` is unable to predict `Duration`, `Cont` or `Pitch`); and also determines whether or not a linked viewpoint comprises a valid combination of primitives (*e.g.*, there is nothing to be gained by testing a viewpoint which has `Duration` ⊗ `Duration` on any layer). Procedure MEMBER() determines whether or not a given element is in an array; procedure ADD() adds a new element to an array; and procedure GET-BEST() can be found in Algorithm 3.4.

```
CHECK-AND-TEST(test-viewpoint, previous-best-system, previous-min-x-entropy, best, delta,
                    rejected-viewpoint, attribute, const-reject, coeff-reject)
if VALID(test-viewpoint, attribute) = TRUE
and MEMBER(test-viewpoint, previous-best-system) = FALSE
and MEMBER(test-viewpoint, rejected-viewpoint) = FALSE
and MEMBER(test-viewpoint, best[3]) = FALSE
    then test-system ← ADD(test-viewpoint, previous-best-system)
         best[3] ← ADD(test-viewpoint, best[3])
         best ← GET-BEST(test-viewpoint, test-system, best, delta,
                    previous-min-x-entropy, const-reject, coeff-reject, TRUE)
return best
```

Algorithm 3.4 Algorithm which returns the multiple viewpoint system with the lowest ten-fold cross-validation cross-entropy seen so far, in the style of Corman et al. (2001). Input parameter *test-viewpoint* is the viewpoint under consideration for addition to or deletion from the multiple viewpoint system; input parameter *test-system* is the multiple viewpoint system after addition or deletion of *test-viewpoint*. Input parameter *best* is an array containing the current best system and its associated ten-fold cross-validation cross-entropy; this array is also a convenient repository for rejected viewpoints and viewpoints already tried during an addition stage. Input parameter *delta* is the reduction in cross-entropy achieved in the previous viewpoint addition and deletion loop; input parameter *previous-min-x-entropy* is the lowest ten-fold cross-validation cross-entropy achieved during the previous viewpoint addition and deletion loop; input parameters *const-reject* and *coeff-reject* are numbers used in the assessment of viewpoints' performance; and input parameter *addition-step* takes the value TRUE only if GET-BEST() is called from CHECK-AND-TEST(). Procedure X-VALIDATION() carries out a ten-fold cross-validation of the corpus and returns the mean cross-entropy; procedure SIMPLER() returns the simpler of two multiple viewpoint systems; procedure MIN() returns the smaller of two values; and procedure ADD() adds a new element to an array.

```
GET-BEST(test-viewpoint, test-system, best, delta, previous-min-x-entropy, const-reject,
                                                          coeff-reject, addition-step)
test-x-entropy ← X-VALIDATION(test-system)
if test-x-entropy < best[1]
  then best[0] ← test-system
       best[1] ← test-x-entropy
  else if test-x-entropy = best[1]
         then if test-system = SIMPLER(test-system, best[0])
                then best[0] ← test-system
                     best[1] ← test-x-entropy
if addition-step = TRUE
and test-x-entropy − previous-min-x-entropy > MIN(const-reject, coeff-reject × delta)
    then best[2] ← ADD(test-viewpoint, best[2])
return best
```

Algorithm 3.5 Algorithm for trying version 0 to 2 linked viewpoints in a multiple viewpoint system, in the style of Corman et al. (2001). Input parameter *primitive* is an array containing the pool of primitive viewpoints; input parameter *attribute* is an array containing the basic viewpoints to be predicted; input parameter *previous-best-system* is an array of selected viewpoints from the previous addition and deletion loop (see Algorithm 3.1); input parameter *previous-min-x-entropy* is its associated ten-fold cross-validation cross-entropy; input parameter *system* is an array containing all viewpoints with which new links will be tried; and input parameter *rejected-viewpoint* is an array of viewpoints no longer considered for selection. Input parameter *best* is an array containing the current best system and its associated ten-fold cross-validation cross-entropy; this array is also a convenient repository for rejected viewpoints and viewpoints already tried during an addition stage. Input parameter *delta* is the reduction in cross-entropy achieved in the previous viewpoint addition and deletion loop; input parameters *const-reject* and *coeff-reject* are numbers used in the assessment of viewpoints' performance; and input parameter *prediction-layer* is an array containing a subset of $\{S, A, T, B\}$. Procedure SIZE() returns the number of elements in an array; procedure PRIMITIVE() determines whether or not a viewpoint is primitive; procedure LINK() links two primitive viewpoints; and procedure CHECK-AND-TEST() can be found in Algorithm 3.3.

```
TRY-LINKS(primitive, attribute, previous-best-system, previous-min-x-entropy, system,
          rejected-viewpoint, best, delta, const-reject, coeff-reject, prediction-layer)
for system-viewpoint ← system[0] to system[SIZE(system) − 1]
    if PRIMITIVE(system-viewpoint, prediction-layer) = TRUE
      then for test-primitive ← primitive[0] to primitive[SIZE(primitive) − 1]
               test-link ← LINK(test-primitive, system-viewpoint)
               best ← CHECK-AND-TEST(test-link, previous-best-system,
                          previous-min-x-entropy, best, delta, rejected-viewpoint,
                          attribute, const-reject, coeff-reject)
return best
```

The **for** loop beginning on line 3 of Algorithm 3.5 is traversed twice more for consideration of $\texttt{Pitch} \otimes \texttt{Pitch}$ (not tested) and $\texttt{Pitch} \otimes \texttt{ScaleDegree}$, following which the **for** loop beginning on line 1 is traversed again for consideration of $\texttt{Duration} \otimes \texttt{Duration}$ (not tested), $\texttt{Duration} \otimes \texttt{Pitch}$ (already tested) and $\texttt{Duration} \otimes \texttt{ScaleDegree}$. We assume that at this point array *best* contains best-performing system $\{\texttt{Pitch}, \texttt{Duration} \otimes \texttt{ScaleDegree}\}$, along with its associated cross-entropy (say 3.39 bits/note). The array is returned from line 6 to line 18 of Algorithm 3.1.

Array *augmented-system* is set to $\{\texttt{Pitch}, \texttt{Duration}, \texttt{Duration} \otimes \texttt{ScaleDegree}\}$ in line 19 of Algorithm 3.1; and Algorithm 3.6 is invoked in line 20. In line 1 of the latter, variable *addition-stage-min-x-entropy* is assigned the value 3.39 bits/note. In the **for** loop beginning on line 2, variable *system-viewpoint* is set to Pitch, which means that array *test-system* becomes $\{\texttt{Duration} \otimes \texttt{ScaleDegree}\}$. The predicate in line 4 is true; therefore line 5 is executed next in which Algorithm 3.4 is invoked, returning array *best* containing best-performing system $\{\texttt{Duration} \otimes \texttt{ScaleDegree}\}$ with a cross-entropy of 3.31 bits/note. The predicate in line 6 is true, and so array *best* is returned from line 7 to line 20 of Algorithm 3.1.

Variable *delta* is assigned the value 0.90 in line 21 of Algorithm 3.1. The predicate in line 22 is false; therefore the **do** loop is traversed again from line 5. Viewpoint selection continues in this way until the predicate in line 22 is true, at which point array *best* is returned from line 23.

We now move on to a similar version 3 example using option VS1, predicting alto and tenor given soprano and bass. The input parameters are:

version = 3
option = VS1
primitive = $\{\texttt{Duration}, \texttt{Pitch}, \texttt{ScaleDegree}\}$
attribute = $\{\texttt{Pitch}\}$
additional = $\{\}$
const-reject = 0.01
coeff-reject = 0.4
prediction-layer = $\{A, T\}$
support-layer = $\{S, B\}$
halting-value = 0.0015.

This example mirrors the version 0 example until line 13 of Algorithm 3.1 is reached, except that construction of viewpoints begins with, for example, $(\texttt{Pitch})_{Ap} \otimes (\texttt{Pitch})_{Tp}$, which for succinctness is abbreviated to $(\texttt{Pitch})_{ATp}$. In line 12, array *best* contains best-performing system $\{(\texttt{Pitch})_{ATp}, (\texttt{ScaleDegree})_{ATp}\}$, along with its associated cross-entropy (say 5.23 bits/prediction). The predicate in line 13 is true; therefore line 14 is the next to be executed, where array *system* becomes $\{(\texttt{Pitch})_{ATp}\}$. The predicate in line 16 is true, and so line 17 is executed next, where Algorithm 3.7 is invoked.

Algorithm 3.6 Algorithm for trying viewpoint deletions from a multiple viewpoint system, in the style of Corman et al. (2001). Input parameter *best* is an array containing the current best system and its associated ten-fold cross-validation cross-entropy; this array is also a convenient repository for rejected viewpoints and viewpoints already tried during an addition stage. Input parameter *delta* is the reduction in cross-entropy achieved in the previous viewpoint addition and deletion loop; input parameter *previous-min-x-entropy* is the lowest ten-fold cross-validation cross-entropy achieved during the previous viewpoint addition and deletion loop; and input parameters *const-reject* and *coeff-reject* are numbers used in the assessment of viewpoints' performance. Procedure SIZE() returns the number of elements in an array; procedure DELETE() removes an element from an array; procedure CAN-PREDICT-ALL() determines whether or not a multiple viewpoint system is able to predict all attributes at all sequence positions; and procedure GET-BEST() can be found in Algorithm 3.4.

```
TRY-DELETIONS(best, delta, previous-min-x-entropy, const-reject, coeff-reject)
addition-stage-min-x-entropy ← best[1]
for system-viewpoint ← best[0][0] to best[0][SIZE(best[0]) − 1]
    test-system ← DELETE(system-viewpoint, best[0])
    if CAN-PREDICT-ALL(test-system) = TRUE
       then best ← GET-BEST(system-viewpoint, test-system, best, delta,
                              previous-min-x-entropy, const-reject, coeff-reject, FALSE)
            if best[1] < addition-stage-min-x-entropy
               then return best
return best
```

Algorithm 3.7 Algorithm for trying version 3 linked viewpoints in a multiple viewpoint system, in the style of Corman et al. (2001). Input parameter *primitive* is an array containing the pool of primitive viewpoints; input parameter *attribute* is an array containing the basic viewpoints to be predicted; input parameter *previous-best-system* is an array of selected viewpoints from the previous addition and deletion loop (see Algorithm 3.1); input parameter *previous-min-x-entropy* is its associated ten-fold cross-validation cross-entropy; input parameter *system* is an array containing all viewpoints with which new links will be tried; and input parameter *rejected-viewpoint* is an array of viewpoints no longer considered for selection. Input parameter *best* is an array containing the current best system and its associated ten-fold cross-validation cross-entropy; this array is also a convenient repository for rejected viewpoints and viewpoints already tried during an addition stage. Input parameter *delta* is the reduction in cross-entropy achieved in the previous viewpoint addition and deletion loop; input parameters *const-reject* and *coeff-reject* are numbers used in the assessment of viewpoints' performance; and input parameters *prediction-layer* and *support-layer* are arrays containing partitions of $\{S, A, T, B\}$. Procedure SIZE() returns the number of elements in an array; procedure PRIMITIVE() determines whether or not a viewpoint is primitive; procedure INTRA-LINK() links a new primitive viewpoint to an existing primitive viewpoint on one or more layers; procedure CHECK-AND-TEST() can be found in Algorithm 3.3; procedure LAYER-EXISTS() determines whether or not there is at least one viewpoint on a given support layer; and procedure INTER-LINK() places the first primitive viewpoint on a support layer.

```
TRY-V3-LINKS(primitive, attribute, previous-best-system, previous-min-x-entropy, system,
        rejected-viewpoint, best, delta, const-reject, coeff-reject, prediction-layer, support-layer)
for system-viewpoint ← system[0] to system[SIZE(system) − 1]
    for test-primitive ← primitive[0] to primitive[SIZE(primitive) − 1]
        if PRIMITIVE(system-viewpoint, prediction-layer) = TRUE
          then test-link ← INTRA-LINK(test-primitive, system-viewpoint,
                                      prediction-layer)
               best ← CHECK-AND-TEST(test-link, previous-best-system,
                       previous-min-x-entropy, best, delta, rejected-viewpoint,
                       attribute, const-reject, coeff-reject)
        for layer ← support-layer[0] to support-layer[SIZE(support-layer) − 1]
            if LAYER-EXISTS(system-viewpoint, layer) = TRUE
              then if PRIMITIVE(system-viewpoint, layer) = TRUE
                     then test-link ← INTRA-LINK(test-primitive, system-viewpoint,
                                                 layer)
                          best ← CHECK-AND-TEST(test-link, previous-best-system,
                                  previous-min-x-entropy, best, delta, rejected-viewpoint,
                                  attribute, const-reject, coeff-reject)
              else test-link ← INTER-LINK(test-primitive, system-viewpoint, layer)
                   best ← CHECK-AND-TEST(test-link, previous-best-system,
                           previous-min-x-entropy, best, delta, rejected-viewpoint,
                           attribute, const-reject, coeff-reject)
return best
```

In the **for** loop beginning on line 1 of Algorithm 3.7, variable *system-viewpoint* becomes $(\texttt{Pitch})_{ATp}$, which is the only element of array *system*. In the **for** loop beginning on line 2, variable *test-primitive* is set to `Duration` in the first instance. The predicate in line 3 is true; therefore lines 4 and 5 are executed. Variable *test-link* becomes $(\texttt{Duration} \otimes \texttt{Pitch})_{ATp}$ in line 4, while in line 5 array *best* is returned unchanged (assuming that system $\{(\texttt{Pitch})_{ATp}, (\texttt{Duration} \otimes \texttt{Pitch})_{ATp}\}$ performs less well than $\{(\texttt{Pitch})_{ATp}, (\texttt{ScaleDegree})_{ATp}\}$). In the **for** loop beginning on line 6, variable *layer* is set to S in the first instance. The predicate in line 7 is false; therefore line 11 is executed next, in which variable *test-link* becomes $(\texttt{Duration})_S \otimes (\texttt{Pitch})_{ATp}$. Array *best* is returned unaltered in line 12. The **for** loop beginning on line 6 is traversed again, with variable *layer* set to B. This time, variable *test-link* becomes $(\texttt{Pitch})_{ATp} \otimes (\texttt{Duration})_B$ in line 11; and *best* is again returned unchanged in line 12 (note that the alternative in lines 8 to 10 can easily be understood on inspection). The **for** loop beginning on line 2 is traversed twice more for consideration of $(\texttt{Pitch} \otimes \texttt{Pitch})_{ATp}$ (not tested), $(\texttt{Pitch})_S \otimes (\texttt{Pitch})_{ATp}$, $(\texttt{Pitch})_{ATp} \otimes (\texttt{Pitch})_B$, $(\texttt{Pitch} \otimes \texttt{ScaleDegree})_{ATp}$, $(\texttt{ScaleDegree})_S \otimes (\texttt{Pitch})_{ATp}$ and $(\texttt{Pitch})_{ATp} \otimes (\texttt{ScaleDegree})_B$. We assume that at this point array *best* contains best-performing system $\{(\texttt{Pitch})_{ATp}, (\texttt{Pitch})_{ATp} \otimes \texttt{Pitch})_B\}$, along with its associated cross-entropy (say 4.91 bits/prediction). This array is returned from line 13 to line 17 of Algorithm 3.1, whence that algorithm continues in much the same way as for the version 0 example.

3.5 The Corpus

Since chord labelling has been an important part of previous research into the modelling of harmony (*e.g.*, Ponsford et al. 1999), it is necessary to give serious consideration to this issue. One option is to annotate the corpus by hand, which of course is very time consuming; another is to use an existing chord labelling program; a third is to use text files from Bach (1998) as the corpus, as they are already annotated with chord labels; and a fourth is to not label the corpus at all. There is no guarantee that the chord labels in Bach (1998) are correct, and no existing automatic chord labelling technique is completely reliable. The decision for this research was option four, which is not to label the corpus at all. Although this decision does not rule out a `Harmony` viewpoint altogether (since it can be defined in terms of simple rules which map sets of notes in simultaneities to harmonic labels), it was nevertheless also decided that no `Harmony` viewpoint would be implemented. This allows us to gauge how well combinations of viewpoints with less complex music theoretic knowledge cope with the harmonisation task in the absence of such a viewpoint, which is still very much an option for the future.

Strictly speaking there are two corpora, ‘A’ and ‘B’, each comprising the music to fifty hymns from Vaughan Williams (1933). There is also a set of test data associated with each corpus, comprising the music to five hymns. All of the music is in a major

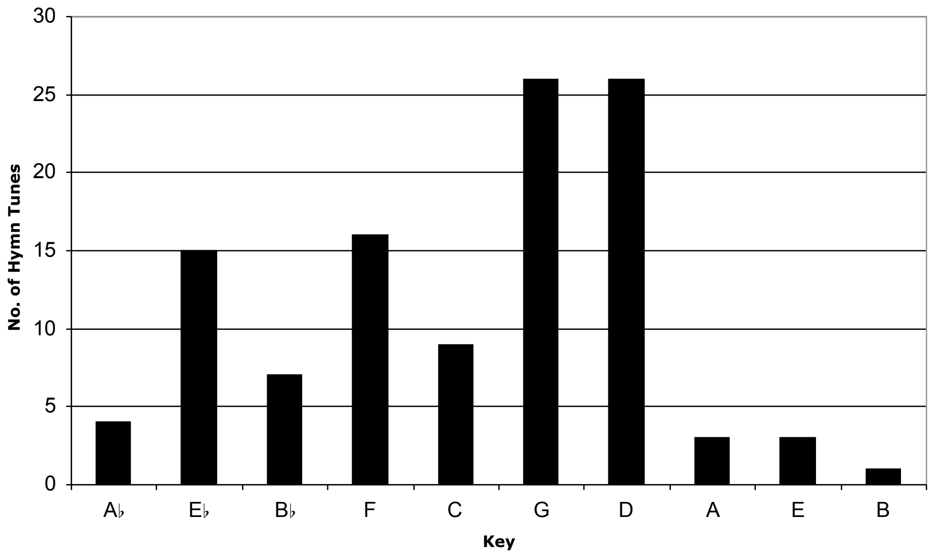


Figure 3.6: Bar chart showing the number of hymn tunes for each major key in the combined ('A+B') corpus and test data.

key (although this does not rule out excursions into minor keys) and contains no rests.[7] In order to facilitate ten-fold cross-validation, a corpus comprises ten sub-corpora of the same length (in terms of hymns). A complete listing of the corpora and test data is given in Appendix B. Corpora 'A' and 'B' can be combined in the hope and expectation that the less sparse statistics will produce better melody and harmony.

Figure 3.6 shows the distribution of keys in the combined ('A+B') corpus and test data. It transpires that G major and D major are the most common keys (26 hymns each), whereas B major is the least common of the keys which appeared at all (1 hymn). Interestingly, there were far more "2 sharp" hymns than "2 flat" hymns, and many more "3 flat" hymns than "3 sharp" hymns. Bearing in mind that the vast majority of melodies end on the tonic, this could be due to a general preference to end a melody lower rather than higher. Although E is low and comfortable to sing, however, E major is not well represented. It has 4 sharps; Vaughan Williams (1933) could well be avoiding keys with many sharps or flats for the sake of the amateur musicians who make up the vast majority of church choirs. The frequency of G major compared with F major (1 sharp and 1 flat respectively) is anomalous; but a possible explanation is that G2 and F2 are at the bottom end of the bass range, with G2 being easier to sing than F2 (bearing in mind that phrases often end on a low tonic in the bass).

In order to provide values for viewpoint Phrase, the phrase boundaries of the music must be ascertained. The automatic segmentation (or grouping) of music into motifs, phrases and so on is an active area of research; for example, the Information Dynamics of

[7] Rests are problematic, as they are not considered to be events within the current viewpoint formulation; since rests are not common in hymn tunes and their harmonisations, however, music containing rests is simply excluded from this research. This issue will be addressed in future work.

Music (IDyOM) model (Pearce et al., 2008) employs an information theoretic approach to melodic segmentation. They define *unexpectedness* as the information content of a note given its context, and *uncertainty* as the entropy of the probability distribution of all possible notes that could have occurred at that point. They hypothesise that high unexpectedness and uncertainty occur directly after a boundary. This issue is not one which is being addressed in the present research, however, which necessarily assumes that segmentation can be done. Here, then, phrase boundaries are determined by eye and ear.

Double bar lines in Vaughan Williams (1933) are not necessarily a good indication of where the phrase boundaries are. In hymn no. 14 (*Puer Nobis Nascitur*), for example, there are four lines per verse, but only two double bar lines. Half-way through each of these two "phrases" is what appears to be a cadence, suggesting that there should be four phrases. On the other hand, in hymn no. 15 (*Forest Green*) there are eight lines per verse and eight double bar lines (including the repeat). In this case, it is not clear that there are proper cadences at odd-numbered double bar lines, suggesting that there should be four longer phrases. In such cases, judgements have been made concerning the phrase boundaries.

MIDI files of hymn tunes and their harmonisations were created as a first step to inclusion in the corpus and test-data. The music was exactly as per Vaughan Williams (1933); that is, containing all unessential notes (passing notes, etc.) and including repeats. Each MIDI file was associated with a separate text file containing values for `KeySig`, `Mode`, `BarLength` and `Pulses`. Appropriate information from the files was automatically converted into the input data format required by the multiple viewpoint implementation with the exception of phrase boundary information, which was added by hand. Five input files were produced for each hymn: fully expanded soprano, alto, tenor and bass for the harmonic model; and non-expanded soprano for the melodic model. Each file takes the form of an event sequence, where each event tuple comprises values for the ten basic attributes `Duration`, `Pitch`, `KeySig`, `Mode`, `Cont`, `Onset`, `BarLength`, `Pulses`, `Phrase` and `Piece`. The Common Lisp multiple viewpoint implementation reads these files, and assembles the information into a corpus and test data.

3.6 Evaluation and Methodology

3.6.1 Evaluation

The relative performance of statistical models of melody and harmony can be determined by using them to calculate the cross-entropy of a set of test data which has been held out from the training corpus (Allan, 2002; Conklin, 1990). Cross-entropy is an upper bound on the true entropy of the data; therefore the model assigning the lowest cross-entropy to a set of test data is the most accurate model of the data. There is a problem with this approach, however, which is that there is a great deal of variation in cross-entropy depending on the data-set presented. By instead performing a ten-fold cross-validation of the corpus, we ensure that the evaluation is as general as possible. Since viewpoint selection makes use of ten-fold cross-validation in order to generalise to unseen data, the multiple viewpoint system constructed by the viewpoint selection algorithm has already effectively been evaluated. Although not used in this research, further evaluation of models which (separately) create and harmonise melodies may be carried out by evaluating the melodies and harmonisations they produce, using a development of the consensual assessment technique (Pearce, 2005, see §3.3). A more informal analysis of the output will be carried out instead.

3.6.2 Methodology

We wish to discover which combination of ideas and techniques produces the best possible performance in an overall model of melody or harmony. A series of viewpoint selection runs will be carried out using different values of bias and different maximum N-gram orders, to pinpoint the best models that we are able to find, from amongst long-term only, short-term only and combined long- and short-term. The following comparisons will be made: weighted geometric with weighted arithmetic combination; updated long-term model with static long-term model; and three different ways of combining long- and short-term models. Models using a single multiple viewpoint system to predict all required musical attributes will be compared with models making use of a separate system to predict each attribute. The effect of different corpora on viewpoint selection will be investigated, as will the effect of using the augmented `Pitch` domain (see Chapter 4) rather than the seen domain. Finally, it should be possible to ascertain whether versions 2 and 3 have improved upon version 1.

3.7 Conclusion

In this chapter, the requirements for the pre-processing of data were discussed, and then a detailed analysis of model structure was given. During the course of the latter, three viewpoint selection options were proposed; fifteen new viewpoints were described; and a representational scheme for the harmonic modelling framework was introduced. In addition, three increasingly complex versions of the multiple viewpoint framework were expounded. After a brief discussion of evaluation, the research methodology was outlined. It is anticipated that version 3 will outperform both version 1 and version 2. It is likely that compromises will need to be made between performance and computational complexity.

Chapter 4

Viewpoint Domains and Time Complexity

4.1 Introduction

A *primitive viewpoint* describes a single feature of a sequence of musical objects; in the case of harmony, the objects are concurrently sounding notes. An example of a viewpoint employed in the modelling of music is chromatic pitch (`Pitch`). In this and earlier related research (Conklin and Witten, 1995; Pearce and Wiggins, 2004), the set of valid elements (or symbols, or values) for this viewpoint, its *domain*, comprises pitch values represented as MIDI numbers. This chapter addresses three fundamental issues relating to domains. One issue is the size of the `Pitch` domain; to predict all possible SATB note combinations, the domain is so large that prediction is excessively slow. We discuss principled ways of reducing the domain size, such that reasonable running times are made possible. Another issue is whether or not a domain can be fixed at the beginning of the prediction process, such that it can be used unchanged at all positions in a musical sequence. We explain why this is not, in general, possible. Finally, we see that the construction of domains for *linked viewpoints* (viewpoints formed by combining two or more *primitive viewpoints*) is far from straightforward. We show how to reliably construct linked viewpoint domains of various complexities, and explain why this is important.

§4.2 introduces domain-related issues by considering melodic viewpoint domains; and §4.3, §4.4 and §4.5 deal with the viewpoint domains of versions 1, 2 and 3 of the framework for modelling harmony respectively. A procedure for the construction of complex viewpoint domains is presented in the latter section, along with a detailed example. There is an analysis of the time complexity of the computer model in §4.6; and finally, there is a statement of conclusions in §4.7.

4.2 Melodic Viewpoint Domains

We can ease our way into consideration of issues relating to domains by considering melody alone. Melodic domains are very small compared with harmonic domains; for example, there are only 19 different chromatic pitches (B♭3 to E5) in the soprano parts of our fifty-five hymn 'A' corpus plus test data, with MIDI values of 58 to 76 inclusive. Run time, therefore, is not a problem; but other issues do need to be tackled. First, though, recall from §2.2.4 that $[\tau]$ denotes the domain of τ, and that $\langle\tau\rangle$ is the type set of τ, where τ is the viewpoint type.

4.2.1 Can a Domain be Fixed?

The first of these issues, whether or not a domain can be fixed, has been addressed by Pearce (2005). The domain of a basic viewpoint, such as `Pitch`, "is predefined to be the set of viewpoint elements occurring in the corpus" (Pearce, 2005, p. 115). Basic viewpoint domains are therefore fixed; but let us consider what happens if we assume that a derived viewpoint, such as `Interval`, also has a domain fixed in the same way. A domain comprising all `Interval` values which occur in the corpus will have both positive and negative values; that is, it will contain both ascending and descending intervals. If the previous note had a pitch which was, for example, the lowest in the `Pitch` domain, then any negative values ending up in the `Interval` prediction probability distribution will predict `Pitch` values which are not in the `Pitch` domain. On the other hand, some of the higher values in the `Pitch` domain will not be predicted by anything in the `Interval` distribution. Since "[a] model m_τ must return a complete distribution over the basic attributes in $\langle\tau\rangle$" (Pearce, 2005, p. 115), something must be done. The solution Pearce (2005, p. 115) comes up with is as follows:

> To address this problem, the domain of each derived type τ is set prior to prediction of each event such that there is a one-to-one correspondence between $[\tau]$ and the domain of the basic type $\tau_b \in \langle\tau\rangle$ currently being predicted.

Although a truly one-to-one correspondence between `Interval` and `Pitch` domains is achievable, this is not necessarily the case for other viewpoints. It is important to be clear what is really meant here by a one-to-one correspondence; the term is used in an informal rather than a strictly mathematical sense. Let us consider the case of `ScaleDegree` (chromatic pitch interval from tonic). Unless the `Pitch` domain is unrealistically small, the `ScaleDegree` domain is

$$\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}.$$

For a `Pitch` domain covering more than an octave, therefore, the function from `Pitch` to `ScaleDegree` is not one-to-one, but *surjective.* This is also true of other derived viewpoints; so the solution to the problem should be clarified as follows. The domain of a derived type τ is set prior to prediction of each event such that, between them, its members are able to predict all of, and only, the members of the domain of the basic type $\tau_b \in \langle\tau\rangle$ currently being predicted.

Irrespective of `ScaleDegree`, which (in theory and practice) has a fixed domain, there exist derived viewpoint domains which are not fixed or static, but dynamic with respect to sequence position (*i.e.*, we do not know *a priori* what the domains are). In the case of `Interval`, the domain comprises the intervals between the pitch of the previous note in the sequence and each of the members of the `Pitch` domain in turn; that is, the domain is a partition of [`Pitch`] × [`Pitch`]. In general, each element of the basic domain in turn is converted to an element of the derived domain, and the latter is added to the derived viewpoint domain if it is not already a member of the domain (alternatively, they can all be added to the domain initially, and duplicates removed afterwards).

4.2.2 Linked Viewpoint Domains

We now turn our attention to the construction of domains for linked viewpoints. Conklin and Witten (1995) state that the linked domain is the Cartesian product of the individual viewpoint domains, that is,

$$[\tau] = [\tau_1] \times \ldots \times [\tau_n].$$

For links between functionally unrelated viewpoints, such as `Duration` and `Pitch`, this is undoubtedly true. Constructing linked domains for viewpoints capable of predicting the same basic viewpoint is rather different, however. Let us consider the linked viewpoint `Pitch`$\otimes$`Interval`. The majority of elements in the Cartesian product of the `Pitch` and `Interval` domains are nonsensical: this is because, as we have already established above, there is a true one-to-one correspondence between elements of the individual viewpoint domains in this case. Only links between corresponding elements make logical sense; therefore only these links should be included in the domain. Researchers have so far only used linked viewpoints comprising two individual viewpoints; but the methods outlined in this and following sections can easily be extended to the linking of more than two viewpoints (explicitly so in §4.5).

4.3 Version 1 Viewpoint Domains

4.3.1 Domain Size and Run Time

In this section, we provide some numbers to give an idea of the run time problem. A detailed empirical analysis of time complexity is unfolded in §4.6. As noted at the beginning of §4.2, there are 19 elements in the soprano `Pitch` domain. In addition, there are 18, 20 and 23 elements in the alto, tenor and bass `Pitch` domains respectively. To enable a probability distribution to predict any combination of these pitches, the *full* (`Pitch`)$_{SATB}$ domain (and distributions based on it) must contain 157,320 vertical viewpoint elements; and since, for example, the $(\texttt{Duration} \otimes \texttt{Pitch})_{SATB}$ domain is the Cartesian product of the constituent viewpoint domains, there would be well over a million elements in the linked domain. There is also a total of 277 chords in our fully expanded test set of five hymn tune harmonisations, each chord having three attributes requiring prediction. If we assume a long-term model only, having a multiple viewpoint system with three viewpoints able to predict `Duration`, three able to predict `Cont` and four able to predict `Pitch`, then the total number of prediction probability distributions required is 2,770 (ignoring the fact that the viewpoints might not always be defined). If, in addition, we use a short-term model, this figure rises to 5,540. This may not seem too bad; but now consider what happens during viewpoint selection. Each multiple viewpoint system tried is evaluated using ten-fold cross-validation of the corpus, which means that fifty hymn tune harmonisations are predicted (in the case of corpus 'A'), rather than just five; and many systems need to be evaluated during viewpoint selection. In spite of a certain amount of caching of distributions, a typical viewpoint selection run (using a pool of 39 primitive viewpoints) for the optimisation of both a long-term and a short-term model requires about 3×10^6 distributions to be constructed. The need to repeatedly construct very large probability distributions definitely results in run time problems.

One way of substantially improving run time is to cut the number of different combinations by placing into a basic viewpoint domain only those vertical elements which occur in the corpus, plus any others likely to occur. Since there is no easy way of identifying these additional elements, one solution is to simply include such elements which occur in (nominally unseen) test data.[1] This has the practical advantage that probabilities can be assigned to all chords in the test data. There are currently 882 vertical elements in our *seen* $(\texttt{Pitch})_{SATB}$ domain. The basic viewpoints $(\texttt{Duration})_{SATB}$ and $(\texttt{Cont})_{SATB}$ have much smaller domains than $(\texttt{Pitch})_{SATB}$: only 11 vertical $(\texttt{Duration})_{SATB}$ elements appear in the data, while the $(\texttt{Cont})_{SATB}$ domain is limited to 15 (since $\langle T, T, T, T\rangle$ does not occur in the data).

If we were predicting or generating all four parts of the harmonic texture, this would be as far as we could go with respect to simplifying the basic domains. What we are particularly interested in, however, is the harmonisation of given melodies.[2] In this case, since the soprano is given, and we do not wish to change it, only those vertical elements having a soprano pitch value the same as that given are allowed in the $(\texttt{Pitch})_{SATB}$ domain (and the resulting prediction probability distribution). In other words, domain elements contain a "must have" value from the support layer. This again greatly improves run time, since if the 882 vertical $(\texttt{Pitch})_{SATB}$ elements were divided equally amongst the 19 soprano $(\texttt{Pitch})_{SATB}$ elements, the size of the domain would be

[1] No statistics are collected from the test data; we are merely acknowledging the existence of vertical viewpoint elements which were not seen in the corpus.

[2] This is a hard enough problem to be tackling for the time being. Predicting all four parts is even more difficult, because it is a less constrained problem.

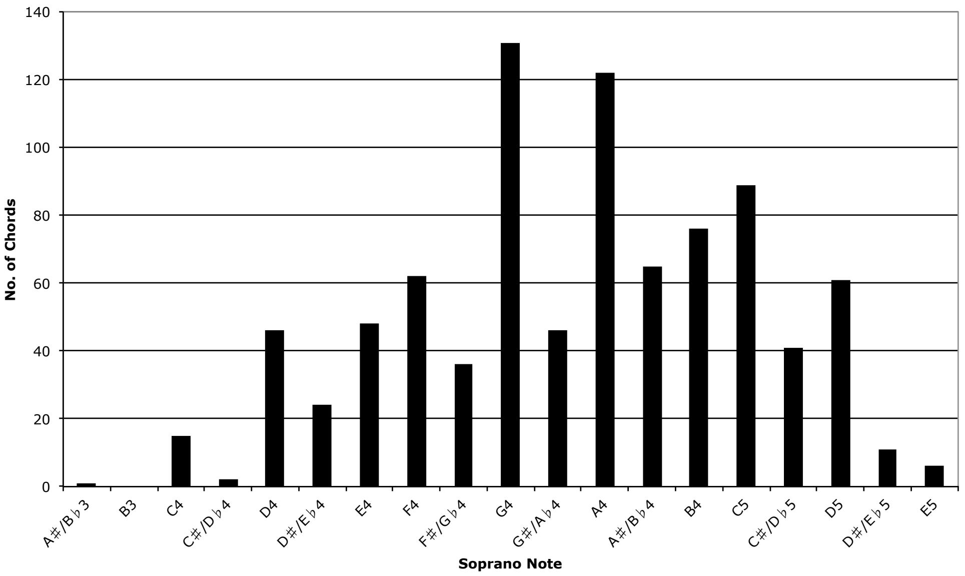


Figure 4.1: Bar chart showing the number of different vertical elements in the seen $(\texttt{Pitch})_{SATB}$ domain for each soprano note.

reduced to about 46. Of course, the distribution of elements is not really uniform; the true distribution is shown in Figure 4.1.

The use of the seen domain for the generation of test data melody harmonisations is not ideal, however. If the test data contains a single vertical element containing a soprano note not seen in the corpus, then that element has to be used to harmonise that soprano note. In addition, we know from Figure 4.5, which includes data from 110 hymns, that roughly 400 elements will be added to our 55-hymn seen $(\texttt{Pitch})_{SATB}$ domain by doubling the number of hymns. In other words, there are a great many chords “out there” which have not been seen by the machine learning program, but which could perfectly well be used to harmonise melodies. Our solution is simply to transpose chords which have been seen up and down, semitone by semitone, until one or other of the parts goes out of the range seen in the data. Such elements are added to the *augmented* $(\texttt{Pitch})_{SATB}$ domain, with the proviso that no duplicates are allowed. Obviously, these elements do not appear in the $(\texttt{Pitch})_{SATB}$ statistics gathered from the corpus, and so have very low prediction probabilities in $(\texttt{Pitch})_{SATB}$ distributions; but they can potentially have relatively high prediction probabilities in derived viewpoint distributions, such as those using $(\texttt{ScaleDegree})_{SATB}$. There are currently 5,040 vertical elements in our augmented $(\texttt{Pitch})_{SATB}$ domain. If these elements were divided equally amongst the 19 soprano `Pitch` elements, the size of the domain for each prediction would be reduced to about 265. The true distribution of elements is shown in Figure 4.2. Note that the augmented domain is much closer in size to the seen domain than it is to the full domain. The seen domain is always a subset of the augmented domain, and for a large enough corpus could potentially closely approach the augmented domain in terms of size. Similarly, the augmented domain is always a subset of the full

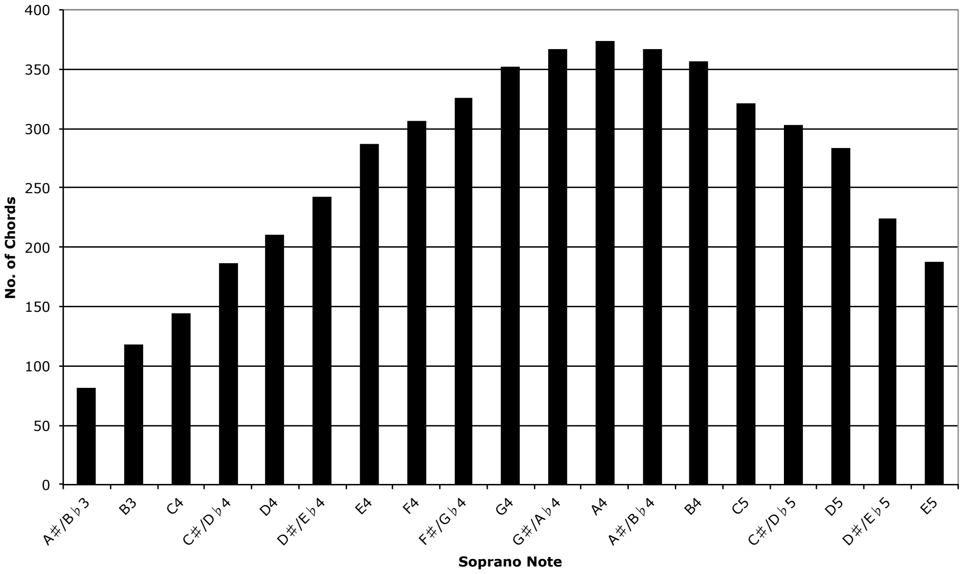


Figure 4.2: Bar chart showing the number of different vertical elements in the augmented $(\texttt{Pitch})_{SATB}$ domain for each soprano note.

domain; but unless the corpus were to contain a large number of very strange chords, the size of the augmented domain could never approach that of the full domain.

4.3.2 Less Obvious Constraints

Following Conklin (1990) and Pearce (2005), we predict each basic attribute in turn at each chord prediction. In this research, prediction is carried out in the following order: `Duration`, `Cont`, `Pitch`. The reasoning is that `Duration` distributions were thought to have the lowest entropy,[3] meaning that `Duration` is the most predictable attribute. Similarly, in general, `Pitch` distributions have the highest entropy, meaning that `Pitch` is usually the least predictable attribute. Following their prediction, known values of `Duration` and `Cont` (or viewpoint types derived from them) are used in linked viewpoints better to predict the (normally) more unpredictable `Pitch`.

Let us consider the basic viewpoint $(\texttt{Duration})_{SATB}$ from the point of view of generation of a harmonisation. First of all, since a simultaneity has a single overall duration, a vertical viewpoint element always contains the same duration value for each part, such as $\langle 48,\ 48,\ 48,\ 48\rangle$. The domain comprises vertical elements containing durations which are less than or equal to the duration of the soprano note to be harmonised.[4] This allows the possibility of passing and other unessential notes in the lower three parts. If, for example, the soprano note had a duration of 24 (a crotchet), and a duration of 12 (a quaver) was generated, the soprano note would be expanded, with the second quaver being assigned a `Cont` value of T. The $(\texttt{Cont})_{SATB}$ and $(\texttt{Pitch})_{SATB}$ values of the first

[3] This was later found to be incorrect. We can infer from the results in Chapter 6 that `Cont` distributions generally have the lowest entropy.

[4] During prediction of expanded data, this includes any continuations of the soprano note.

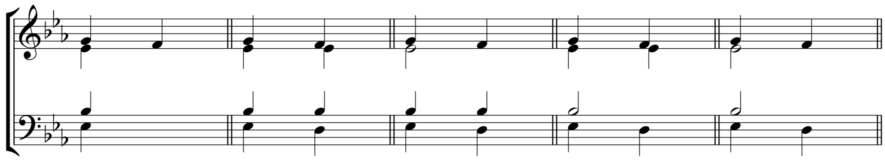


Figure 4.3: The effect of viewpoint $(\texttt{Cont})_{SATB}$ on a chord progression. Bar 1 shows the soprano note to be harmonised and the preceding chord. Bars 2 to 5 illustrate the effect of vertical $(\texttt{Cont})_{SATB}$ elements $\langle F,\ F,\ F,\ F\rangle$, $\langle F,\ T,\ F,\ F\rangle$, $\langle F,\ F,\ T,\ F\rangle$ and $\langle F,\ T,\ T,\ F\rangle$ respectively on the chosen second chord.

quaver are predicted before the focus is shifted to the second quaver, when prediction begins again with $(\texttt{Duration})_{SATB}$. After the prediction of $(\texttt{Duration})_{SATB}$, its domain contains only the predicted vertical viewpoint.

There are interactions between `Cont` and `Pitch`, with different constraints operating depending on the attribute being predicted. During the prediction of $(\texttt{Cont})_{SATB}$, the $(\texttt{Cont})_{SATB}$ domain must not contain elements which would predict elements outside of the $(\texttt{Pitch})_{SATB}$ domain. For example, let us assume that we have an E♭ major chord, $\langle 67,\ 63,\ 58,\ 51\rangle$, followed by an F4 (MIDI value 65) in the soprano, as shown in the first bar of Figure 4.3. For simplicity, let us also assume that the only vertical element in the $(\texttt{Pitch})_{SATB}$ domain constrained by the soprano note is $\langle 65,\ 63,\ 58,\ 50\rangle$. The set of vertical $(\texttt{Cont})_{SATB}$ elements compatible with this artificially small $(\texttt{Pitch})_{SATB}$ domain is

$$\{\langle F,\ F,\ F,\ F\rangle,\ \langle F,\ T,\ F,\ F\rangle,\ \langle F,\ F,\ T,\ F\rangle,\ \langle F,\ T,\ T,\ F\rangle\}.$$

Any other $(\texttt{Cont})_{SATB}$ element, such as $\langle F,\ T,\ T,\ T\rangle$ or $\langle F,\ F,\ F,\ T\rangle$, would map onto a $(\texttt{Pitch})_{SATB}$ element which is not in the domain. Bars 2 to 5 of Figure 4.3 show the chord progression resulting from each of these $(\texttt{Cont})_{SATB}$ elements respectively, in conventional musical notation. Even realistically sized $(\texttt{Pitch})_{SATB}$ domains can restrict the $(\texttt{Cont})_{SATB}$ domain beyond what might be expected by constraining according to the `Cont` attribute of the soprano note. Once $(\texttt{Cont})_{SATB}$ has been predicted, its domain comprises a single vertical element, and the $(\texttt{Pitch})_{SATB}$ domain is further constrained to vertical elements which are compatible with the predicted vertical $(\texttt{Cont})_{SATB}$ element, given the preceding $(\texttt{Pitch})_{SATB}$ element.

4.3.3 Construction of Derived and Linked Viewpoint Domains

Derived viewpoint domains are constructed by converting each vertical element of the relevant basic viewpoint domain (however it is currently constrained) into a vertical element of the derived domain. By constructing the domain in this way, we can be sure that between them, the members will be able to predict all of, and only, the members of the basic domain. As with melodic domains, after each conversion the vertical element is added to the derived domain only if it is not already a member (recalling that the conversion function is surjective). For example, assuming a key of A, vertical $(\texttt{Pitch})_{SATB}$ elements $\langle 69,\ 64,\ 61,\ 57\rangle$ and $\langle 69,\ 64,\ 61,\ 45\rangle$ both map to vertical $(\texttt{ScaleDegree})_{SATB}$ element $\langle 0,\ 7,\ 4,\ 0\rangle$.

As with melodic domains, for links between functionally unrelated viewpoints, the linked domain is the Cartesian product of the individual viewpoint domains; and for links between viewpoints capable of predicting the same basic viewpoint, the informal unidirectional one-to-one correspondence between elements of the individual viewpoint domains means that only corresponding elements are included in the linked domain.[5] Continuing the example in the paragraph above, for a key of A, vertical elements $\langle\langle 69,\ 0\rangle,\ \langle 64,\ 7\rangle,\ \langle 61,\ 4\rangle,\ \langle 57,\ 0\rangle\rangle$ and $\langle\langle 69,\ 0\rangle,\ \langle 64,\ 7\rangle,\ \langle 61,\ 4\rangle,\ \langle 45,\ 0\rangle\rangle$ would both appear in the $(\texttt{Pitch} \otimes \texttt{ScaleDegree})_{SATB}$ domain; but $\langle\langle 69,\ 0\rangle,\ \langle 64,\ 7\rangle,\ \langle 61,\ 4\rangle,\ \langle 55,\ 0\rangle\rangle$ would not, since in the bass part the `Pitch` value of 55 corresponds to a G, and the `ScaleDegree` value of 0 corresponds to an A.

The reason that we must be able to reliably construct such domains is because prediction probability distributions are completed by backing off to uniform distributions based on these domains. The various viewpoint distributions are converted into basic viewpoint distributions prior to combination. To facilitate combination, the distributions are sorted into the same order. An unreliable domain construction procedure might, for example, result in one or two predictions being missing from some distributions, which means that at some point a succession of probabilities will be erroneously combined.

4.4 Version 2 Viewpoint Domains

Provided we always predict only one part at a time, and always constrain the `Pitch` domain as much as possible, we could reasonably construct full domains (*i.e.*, without having to resort to using only combinations which occur in the corpus and test data). Let us, for example, assume that we are predicting bass given soprano. The `Pitch` domain would comprise vertical elements containing the given soprano pitch and each of the pitches occurring in the bass in turn, giving a total of only 23 elements. We wish, however, to retain the option of predicting more than one part at a time; and a domain constructed in this way which is capable of predicting two parts at once would require up to 460 vertical viewpoint elements. The following treatment, then, assumes the use of vertical basic viewpoint elements seen in the corpus and test data, with the option of augmentation with transposed elements as described in §4.3.1.

To obtain basic viewpoint domains for the required combination of parts, say soprano and bass only, the corpus and test data can be traversed, with each new soprano/bass combination (and optionally its transpositions) being added to the relevant domain.

[5] There are also correspondences between `Cont` and `Pitch` (and viewpoints derived from it) as discussed above.

Although not present in the domain, transposed alto and tenor notes must still be within their part ranges. Alternatively, if a domain of vertical elements containing all four parts has already been found, this can be similarly traversed to obtain the domain of soprano/bass elements. Construction of derived and linked viewpoint domains is then carried out in exactly the same way as in version 1.

4.5 Version 3 Viewpoint Domains

4.5.1 Domain Construction Issues

Version 3 viewpoints can be much more complex than any we have seen before: especially when three parts are given and we are predicting the remaining part. In this case, each of the four parts could have a different viewpoint; but let us begin with something far more simple. During prediction of bass given soprano, we may use the viewpoint $(\texttt{ScaleDegree})_S \otimes (\texttt{Pitch})_{Bp}$. Although the constituent viewpoints are able to predict the same basic viewpoint, the fact that they are assigned to different layers means that there are no correspondences which need to be taken into account; therefore taking the Cartesian product of the individual layer domains (as constrained by the given soprano note) is a perfectly acceptable way of constructing the inter-layer linked domain. We would not wish to do so in practice, however, for reasons outlined above. As before, we would place elements into the domain which correspond to `Pitch` combinations found in the corpus and test data (and optionally, transpositions of these combinations).

Unfortunately, most version 3 viewpoints are not as straightforward as this. We are predicting basic viewpoints `Duration`, `Cont` and `Pitch`, each of which has its own domain of a particular size; there are other basic viewpoints which we assume to be given; there are many derived viewpoints, most of them derived from `Pitch`, having domains of various sizes; and there are up to four layers represented in any viewpoint. A general method for reliably constructing domains for these complex viewpoints is required. As usual, the domains must be able to predict all of, and only, the members of the basic viewpoint domain(s). It is possible for each layer to have a different viewpoint; therefore it is expedient to deal with each layer in turn. This being the case, to achieve precisely the correct combinations of viewpoint elements, we need to know in advance how many elements there are in the provisional inter-layer linked viewpoint domain (which, as we shall see, may end up containing undefined or duplicate elements which must be removed). To calculate this number, we also need to know the set of basic viewpoints that the constituent viewpoints are derived from (*i.e.*, the type set $\langle\tau\rangle$);[6] since we can determine what the domains of these basic viewpoints are for the combination of layers in question, we are easily able to ascertain their sizes. For the purposes of constructing a provisional inter-layer linked domain, we assume that the size of each

[6] The set of basic viewpoints that the inter-layer linked viewpoint is able to predict is a subset of $\langle\tau\rangle$, since only constituent viewpoints of the prediction layers should be considered.

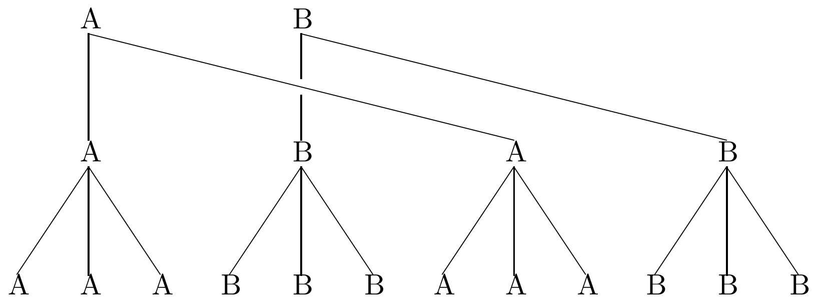


Figure 4.4: Trees illustrating the effect of multipliers on primitive domain $\{A, B\}$. Branching from the roots is due to the outer-multiplier (2); and further branching to produce the leaves is due to the inner-multiplier (3).

derived domain (covering all of the parts represented in the viewpoint) is the same as that of the relevant basic domain. This means that we can relatively easily assign *inner-* and *outer-multipliers* to each basic or derived constituent viewpoint prior to construction of the inter-layer linked viewpoint. These multipliers respectively determine the number of times a primitive domain element is repeated prior to moving on to the next, and the number of times the entire primitive domain is repeated (along with any internal repeats).

As a simple example, let us assume that there are only two elements (A and B) in the primitive domain, and that the inner- and outer-multipliers have values of 3 and 2 respectively. Figure 4.4 illustrates the effect of the multipliers on the primitive domain. The elements of this domain are each shown as the root of a tree. The outer-multiplier is 2; therefore each root divides into two branches, resulting in the primitive domain being shown twice. The inner-multiplier is 3; so each of the four nodes splits into three, thereby duplicating elements within each of the copies of the primitive domain. The number of elements in the provisional inter-layer linked domain is the same as the total number of leaves on the trees, which is the product of the primitive domain size and the two multipliers.

4.5.2 Determination of Multipliers

In calculating the multipliers, we assume that the basic viewpoints are in a particular order: `Duration`, `Cont`, `Pitch`, followed by other basic viewpoints. The order of the other basic viewpoints is not important, as in our research they are given and therefore effectively have a domain containing only one element. Once we know which of `Duration`, `Cont` and `Pitch` the constituent viewpoints are derived from, we can assign multipliers according to their relative positions in the ordered list. For a particular basic viewpoint, the inner- and outer-multipliers are the product of the domain sizes of any basic viewpoints occurring after it, and before it, respectively. The default value of both is 1.

Generalised algorithms for the determination of multipliers can be found in Algo-

Algorithm 4.1 Generalised algorithm for the determination of multipliers, in the style of Corman et al. (2001). It is used in both Algorithm 4.2 and Algorithm 4.3. A domain is an array of all viewpoint elements which could possibly be predicted. Input parameter *basic-domain* is an array of domains corresponding to the basic types in type set $\langle\tau\rangle$. Procedure SIZE() returns the number of elements in an array.

```
GET-MULTIPLIER(start, finish, basic-domain)
1  multiplier ← 1
2  for i ← start to finish
3      multiplier ← multiplier × SIZE(basic-domain[i])
4  return multiplier
```

rithms 4.1, 4.2 and 4.3. As an example, consider a linked viewpoint with constituents derived from `Duration`, `Cont` and `Pitch`, and which also contains viewpoint `LastInPhrase` (derived from basic viewpoint `Phrase`). Algorithm input parameter *basic-type* is an array equivalent to the type set $\langle\tau\rangle$, which is {`Duration`, `Cont`, `Pitch`, `Phrase`}; and input parameter *basic-domain* is an array of corresponding domains. Let us assume that `Duration`, `Cont` and `Pitch` have domain sizes of 5, 10 and 40 respectively, and that we wish to determine the multipliers relating to `Duration` in the first instance. In this case, the input parameter k (the *basic-type* index for the current constituent viewpoint) has a value of 0. The inner-multiplier is obtained from Algorithm 4.2. In line 1, variable n is set to 3, which is the highest *basic-type* index. The predicate in line 2 is false; therefore line 4 is executed next. Algorithm 4.1 is called in this line, with input parameters *start* and *finish* set to 1 and 3 respectively. On line 1, variable *multiplier* is set to 1; and in the **for** loop beginning on line 2, counter i is initially set to 1. In line 3, there are 10 elements in the `Cont` domain; therefore *multiplier* is reassigned to 10. The **for** loop is executed again with i set to 2. In line 3, there are 40 elements in the `Pitch` domain; therefore *multiplier* becomes 400. The **for** loop is executed once more with i set to 3. In line 3, there is only one element in the `Phrase` domain (since `Phrase` attribute values are given); therefore the value of *multiplier* remains at 400. This value is returned to line 4 of Algorithm 4.2, where is is assigned to variable *inner-multiplier.* The value 400 is returned in line 5. The outer-multiplier is obtained from Algorithm 4.3. The predicate in line 1 is true; therefore line 2 is executed next. In this line, variable *outer-multiplier* is set to 1. This value is returned in line 4.

By the continued application of these algorithms, for this example we can see that for any constituent viewpoint derived from `Duration`, the inner- and outer-multipliers are 400 and 1; for any constituent viewpoint derived from `Cont`, the inner- and outer-multipliers are 40 and 5; for any constituent viewpoint derived from `Pitch`, the inner- and outer-multipliers are 1 and 50; and for any other constituent viewpoint (`LastInPhrase` in this case), the inner- and outer-multipliers are 1 and 2000 respectively.

Algorithm 4.2 Algorithm for the determination of inner-multipliers, in the style of Corman et al. (2001). A domain is an array of all viewpoint elements which could possibly be predicted. Input parameter *basic-type* is an array equivalent to the type set $\langle\tau\rangle$; input parameter *basic-domain* is an array of corresponding domains; and input parameter k is the *basic-type* index for the current constituent viewpoint. Procedure SIZE() returns the number of elements in an array; and procedure GET-MULTIPLIER() can be found in Algorithm 4.1.

```
GET-INNER-MULTIPLIER($k$, basic-type, basic-domain)
$n \leftarrow$ SIZE(basic-type) $- 1$
if basic-type[$k$] $=$ basic-type[$n$]
  then inner-multiplier $\leftarrow 1$
  else inner-multiplier $\leftarrow$ GET-MULTIPLIER($k+1$, $n$, basic-domain)
return inner-multiplier
```

Algorithm 4.3 Algorithm for the determination of outer-multipliers, in the style of Corman et al. (2001). A domain is an array of all viewpoint elements which could possibly be predicted. Input parameter *basic-type* is an array equivalent to the type set $\langle\tau\rangle$; input parameter *basic-domain* is an array of corresponding domains; and input parameter k is the *basic-type* index for the current constituent viewpoint. Procedure GET-MULTIPLIER() can be found in Algorithm 4.1.

```
GET-OUTER-MULTIPLIER($k$, basic-type, basic-domain)
if basic-type[$k$] $=$ basic-type[0]
  then outer-multiplier $\leftarrow 1$
  else outer-multiplier $\leftarrow$ GET-MULTIPLIER(0, $k-1$, basic-domain)
return outer-multiplier
```

4.5.3 Domain Construction Procedure

A generalised algorithm for the construction of version 3 inter-layer linked viewpoint domains can be found in Algorithm 4.4. This algorithm will be illustrated by a modified real example: only a small subset of the actual `Pitch` domain is used, comprising note names such as A♭4 rather than MIDI numbers, in order to make the illustration more readily intelligible. We are predicting bass given soprano using, amongst other viewpoints,

$$(\texttt{Duration} \otimes \texttt{Pitch})_S \otimes (\texttt{Cont} \otimes \texttt{ScaleDegree})_{Bp}.$$

Input parameter *viewpoint-type* is an array of layers, which in turn are arrays of primitive viewpoints; *viewpoint-type* is therefore the 2-dimensional array

$$\begin{array}{cccc} & & 0 & 1 \\ S & 0 & \texttt{Duration} & \texttt{Pitch} \\ B & 1 & \texttt{Cont} & \texttt{ScaleDegree}. \end{array}$$

The set of basic viewpoints that the above's constituent viewpoints are derived from (type set $\langle\tau\rangle$) is {`Duration`, `Cont`, `Pitch`}. Input parameter *basic-type* is an array containing these three viewpoints in the order given, with indices 0, 1 and 2 respectively. For the purposes of this example, we assume that the `Duration` attribute of a bass note has already been predicted; therefore the `Duration` domain contains only one element:

$$\{\langle\langle S,\ 0,\ \texttt{Duration},\ \langle 24\rangle\rangle\ \langle B,\ 0,\ \texttt{Duration},\ \langle 24\rangle\rangle\rangle\}.$$

The next attribute to be predicted is `Cont`. The given soprano note has a `Cont` attribute of F; therefore the domain is constrained to two elements:

$$\begin{array}{l} \{\langle\langle S,\ 0,\ \texttt{Cont},\ \langle F\rangle\rangle\ \langle B,\ 0,\ \texttt{Cont},\ \langle T\rangle\rangle\rangle, \\ \ \langle\langle S,\ 0,\ \texttt{Cont},\ \langle F\rangle\rangle\ \langle B,\ 0,\ \texttt{Cont},\ \langle F\rangle\rangle\rangle\}. \end{array}$$

The 5-element `Pitch` domain (constrained to have a soprano A♭4) is

$$\begin{array}{l} \{\langle\langle S,\ 0,\ \texttt{Pitch},\ \langle \mathrm{A}\flat 4\rangle\rangle\ \langle B,\ 0,\ \texttt{Pitch},\ \langle \mathrm{F2}\rangle\rangle\rangle, \\ \ \langle\langle S,\ 0,\ \texttt{Pitch},\ \langle \mathrm{A}\flat 4\rangle\rangle\ \langle B,\ 0,\ \texttt{Pitch},\ \langle \mathrm{E}\flat 3\rangle\rangle\rangle, \\ \ \langle\langle S,\ 0,\ \texttt{Pitch},\ \langle \mathrm{A}\flat 4\rangle\rangle\ \langle B,\ 0,\ \texttt{Pitch},\ \langle \mathrm{E3}\rangle\rangle\rangle, \\ \ \langle\langle S,\ 0,\ \texttt{Pitch},\ \langle \mathrm{A}\flat 4\rangle\rangle\ \langle B,\ 0,\ \texttt{Pitch},\ \langle \mathrm{F3}\rangle\rangle\rangle, \\ \ \langle\langle S,\ 0,\ \texttt{Pitch},\ \langle \mathrm{A}\flat 4\rangle\rangle\ \langle B,\ 0,\ \texttt{Pitch},\ \langle \mathrm{F}\sharp 3\rangle\rangle\rangle\}. \end{array}$$

Input parameter *basic-domain* is an array containing the above three domains in the order given, with indices 0, 1 and 2 respectively.

The highest part (in terms of pitch) is dealt with first. If the viewpoint on this layer is, or contains, a basic viewpoint, each of the symbols or values belonging to that layer in the basic viewpoint domain is added, in turn, to what will become the inter-layer linked viewpoint domain, with repeats determined by the inner- and outer-multipliers.

Algorithm 4.4 Generalised algorithm for the construction of version 3 inter-layer linked viewpoint domains, in the style of Corman et al. (2001). A domain is an array of all viewpoint elements which could possibly be predicted, which in turn are arrays of layers. Input parameter *viewpoint-type* is an array of layers, which in turn are arrays of primitive viewpoint types; input parameter *basic-type* is an array equivalent to the type set $\langle\tau\rangle$; and input parameter *basic-domain* is an array of corresponding domains. Procedure SIZE() returns the number of elements in an array; procedure GET-BASIC-TYPE-INDEX() returns the *basic-type* index for the current constituent viewpoint; procedures GET-INNER-MULTIPLIER() and GET-OUTER-MULTIPLIER can be found in Algorithms 4.2 and 4.3 respectively; procedure GET-DERIVED-ELEMENT() converts a basic viewpoint element into a derived viewpoint element; procedure INTRA-LAYER-ELEMENTS-COMPATIBLE() returns TRUE if a viewpoint symbol to be added to a layer portion of a domain element is logically compatible with a symbol (or symbols) already in it; and procedure MERGE-DOMAIN-ELEMENT-LAYER() links a viewpoint type and symbol with a type and symbol already in a layer portion of a domain element. Procedures REMOVE-UNDEFINED-ELEMENTS() and REMOVE-DUPLICATE-ELEMENTS() are self-explanatory.

```
CONSTRUCT-DOMAIN(viewpoint-type, basic-type, basic-domain)
domain ← initialise array
p ← −1
for layer ← viewpoint-type[0] to viewpoint-type[SIZE(viewpoint-type) − 1]
    layer-constituent-count ← −1
    p ← p + 1
    for constituent-viewpoint ← layer[0] to layer[SIZE(layer) − 1]
        layer-constituent-count ← layer-constituent-count + 1
        i ← GET-BASIC-TYPE-INDEX(constituent-viewpoint)
        inner-multiplier ← GET-INNER-MULTIPLIER(i, basic-type, basic-domain)
        outer-multiplier ← GET-OUTER-MULTIPLIER(i, basic-type, basic-domain)
        e ← −1
        for outer-counter ← 1 to outer-multiplier
            j ← SIZE(basic-domain[i]) − 1
            for basic-element ← basic-domain[i][0] to basic-domain[i][j]
                derived-element ← GET-DERIVED-ELEMENT(constituent-viewpoint,
                                                        basic-element)
                for inner-counter ← 1 to inner-multiplier
                    e ← e + 1
                    if layer-constituent-count = 0
                      then domain[e][p] ← derived-element[p]
                      else if INTRA-LAYER-ELEMENTS-COMPATIBLE(domain[e][p],
                                                  basic-element[p]) = TRUE
                             then domain[e][p] ← MERGE-DOMAIN-ELEMENT-
                                   LAYER(domain[e][p], derived-element[p])
                             else  domain[e][p] ← "undef"
domain ← REMOVE-UNDEFINED-ELEMENTS(domain)
domain ← REMOVE-DUPLICATE-ELEMENTS(domain)
return domain
```

If the viewpoint is derived, the procedure is the same except that each basic viewpoint symbol is converted to the relevant derived viewpoint symbol.

In the **for** loop beginning on line 3, variable *layer* is initially assigned the array of viewpoints in the soprano layer. Having been initialised as -1 in line 2, counter p (an index indicating the layer) is incremented to 0 in line 5. In the **for** loop beginning on line 6, variable *constituent-viewpoint* is initially assigned *layer* element `Duration`. Having been initialised as -1 in line 4, *layer-constituent-count* is incremented to 0 in line 7. In line 8, variable i is assigned the *basic-type* index for `Duration`, which is 0. The return value of Algorithm 4.2, in this case, is the product of the `Cont` and `Pitch` domain sizes, which is 10. This value is assigned to variable *inner-multiplier* on line 9. Algorithm 4.3 returns the default value 1, which is assigned to variable *outer-multiplier* on line 10. In line 12, variable *outer-counter* is set to 1; and since there is only one element in *basic-domain*[0] (the `Duration` domain), variable j is set to 0 in line 13. Variable *basic-element* is assigned the single `Duration` element

$$\langle\langle S,\ 0,\ \texttt{Duration},\ \langle 24\rangle\rangle,\ \langle B,\ 0,\ \texttt{Duration},\ \langle 24\rangle\rangle\rangle$$

in line 14. Since the constituent viewpoint is `Duration`, in line 15 variable *derived-element* is assigned precisely the same element as *basic-element*. In line 16, in the first instance, *inner-counter* is set to 1. Having been initialised as -1 in line 11, variable e (an index indicating the inter-layer linked viewpoint domain element) is incremented to 0 in line 17. The predicate in line 18 is true; therefore line 19 is executed, which assigns $\langle S,\ 0,\ \texttt{Duration},\ \langle 24\rangle\rangle$ to array element *domain*[0][0]. Lines 16 to 19 (inclusive) are executed a further nine times, giving a provisional linked domain of

$$\begin{aligned}\{&\langle\langle S,\ 0,\ \texttt{Duration},\ \langle 24\rangle\rangle\rangle,\ \langle\langle S,\ 0,\ \texttt{Duration},\ \langle 24\rangle\rangle\rangle,\\ &\langle\langle S,\ 0,\ \texttt{Duration},\ \langle 24\rangle\rangle\rangle,\ \langle\langle S,\ 0,\ \texttt{Duration},\ \langle 24\rangle\rangle\rangle,\\ &\langle\langle S,\ 0,\ \texttt{Duration},\ \langle 24\rangle\rangle\rangle,\ \langle\langle S,\ 0,\ \texttt{Duration},\ \langle 24\rangle\rangle\rangle,\\ &\langle\langle S,\ 0,\ \texttt{Duration},\ \langle 24\rangle\rangle\rangle,\ \langle\langle S,\ 0,\ \texttt{Duration},\ \langle 24\rangle\rangle\rangle,\\ &\langle\langle S,\ 0,\ \texttt{Duration},\ \langle 24\rangle\rangle\rangle,\ \langle\langle S,\ 0,\ \texttt{Duration},\ \langle 24\rangle\rangle\rangle\}.\end{aligned}$$

At this point, the provisional inter-layer linked viewpoint domain contains all of the elements we need (and probably more), albeit that they are incomplete. If there is a second basic or derived viewpoint associated with this layer (*i.e.*, a constituent of an intra-layer linked viewpoint), then the same procedure is followed except that the symbols are linked with the symbols already in the provisional domain, in the same order as the original additions to the domain.

The **for** loop beginning on line 6 is executed again, with *constituent-viewpoint* set to `Pitch`. In line 7, *layer-constituent-count* is incremented to 1; and in line 8, i is assigned the value 2 (the *basic-type* index for `Pitch`). Algorithm 4.2 returns the default value 1, which is assigned to *inner-multiplier* on line 9. The return value of Algorithm 4.3 is the product of the Duration and Cont domain sizes, which is 2. This value is assigned to *outer-multiplier* on line 10. In line 12, *outer-counter* is initially set to 1; and since there are five elements in *basic-domain*[2] (the Pitch domain), j is set to 4 in line 13. Variable *basic-element* is initially assigned the Pitch element

$$\langle\langle S,\ 0,\ \texttt{Pitch},\ \langle \mathrm{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Pitch},\ \langle \mathrm{F2}\rangle\rangle\rangle$$

in line 14. Since the constituent viewpoint is Pitch, in line 15 *derived-element* is assigned precisely the same element as *basic-element*. In line 16, *inner-counter* is set to 1; and having been initialised as -1 in line 11, e is incremented to 0 in line 17. The predicate in line 18 is false; therefore line 20 is executed next. The predicate in this line can only possibly be false if the current layer contains both Cont and a viewpoint derived from Pitch. In this case, the predicate is true; therefore line 21 is executed, which replaces the contents of *domain*[0][0] with $\langle S,\ 0,\ \texttt{Duration} \otimes \texttt{Pitch},\ \langle 24,\ A\flat 4\rangle\rangle$. The **for** loop beginning on line 14 is executed a further four times (thereby running through the rest of the Pitch domain) before executing the **for** loop beginning on line 12 one more time. At this stage the soprano layer has been completed, giving a provisional linked domain of

$$\begin{aligned}\{&\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle\rangle,\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle\rangle,\\ &\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle\rangle,\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle\rangle,\\ &\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle\rangle,\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle\rangle,\\ &\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle\rangle,\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle\rangle,\\ &\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle\rangle,\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle\rangle\}.\end{aligned}$$

We then move on to the next layer, again adding to the elements already in the provisional domain, and so on, until all the layers have been dealt with. There is only one other layer in the example, the bass part, which contains the intra-layer linked viewpoint Cont ⊗ ScaleDegree.

The **for** loop beginning on line 3 is executed again, with *layer* assigned the array of viewpoints in the bass layer. Counter p is incremented to 1 in line 5; and in the **for** loop beginning on line 6, *constituent-viewpoint* is initially assigned Cont. Having been initialised as -1 in line 4, *layer-constituent-count* is incremented to 0 in line 7; and in line 8, i is assigned the value 1 (the *basic-type* index for Cont). The return value of Algorithm 4.2 is the Pitch domain size, which is 5. This value is assigned to variable *inner-multiplier* on line 9. Algorithm 4.3 returns 1 (the size of the Duration domain), which is assigned to variable *outer-multiplier* on line 10. In line 12, *outer-counter* is set to 1; and since there are two elements in *basic-domain*[1] (the Cont domain), j is set to 1 in line 13. Variable *basic-element* is initially assigned the Cont element

$$\langle\langle S,\ 0,\ \texttt{Cont},\ \langle F\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont},\ \langle T\rangle\rangle\rangle$$

in line 14. Since the constituent viewpoint is `Cont`, in line 15 *derived-element* is assigned precisely the same element as *basic-element*. In line 16, in the first instance, *inner-counter* is set to 1; and having been initialised as -1 in line 11, e is incremented to 0 in line 17. The predicate in line 18 is true; therefore line 19 is executed, which assigns $\langle B,\ 0,\ \texttt{Cont},\ \langle T\rangle\rangle$ to array element *domain*[0][1]. Lines 16 to 19 (inclusive) are executed a further four times, and then the **for** loop beginning on line 14 is executed again (thereby processing the remaining `Cont` domain element). The provisional linked domain then becomes

$$\begin{aligned}
\{&\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont},\ \langle T\rangle\rangle\rangle,\\
&\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont},\ \langle T\rangle\rangle\rangle,\\
&\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont},\ \langle T\rangle\rangle\rangle,\\
&\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont},\ \langle T\rangle\rangle\rangle,\\
&\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont},\ \langle T\rangle\rangle\rangle,\\
&\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont},\ \langle F\rangle\rangle\rangle,\\
&\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont},\ \langle F\rangle\rangle\rangle,\\
&\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont},\ \langle F\rangle\rangle\rangle,\\
&\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont},\ \langle F\rangle\rangle\rangle,\\
&\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \mathrm{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont},\ \langle F\rangle\rangle\rangle\}.
\end{aligned}$$

The final viewpoint to be added is `ScaleDegree` in the bass part; because it predicts `Pitch`, its inner-multiplier is 1, and its outer-multiplier is 2. There is a particular problem to be overcome with respect to the intra-layer linking of `Cont` with `Pitch`, or viewpoints derived from it, however. As we have seen in §4.3.2, there is an interaction between the `Cont` and `Pitch` domains; therefore the Cartesian product would contain pairings making no logical sense. In this case, as the `Pitch` domain is traversed, the `Pitch` symbol is checked against the relevant `Cont` symbol (which is already in the provisional domain): if the pairing makes logical sense, the `Pitch` symbol, or a symbol derived from it, is added to the element in the provisional domain as usual; if not, the element is tagged as undefined. The previous note was F3 (MIDI value 53); therefore 53 is the only `Pitch` value which can be sensibly paired with a `Cont` value of T. The tonic is E♭; therefore F has a `ScaleDegree` value of 2.

The **for** loop beginning on line 6 is executed again, with *constituent-viewpoint* set to `ScaleDegree`. In line 7, *layer-constituent-count* is incremented to 1; and in line 8, i is assigned the value 2 (the *basic-type* index for `Pitch`). Algorithm 4.2 returns the default value 1, which is assigned to *inner-multiplier* on line 9. The return value of Algorithm 4.3 is the product of the `Duration` and `Cont` domain sizes, which is 2. This value is assigned to *outer-multiplier* on line 10. In line 12, *outer-counter* is initially set to 1; and since there are five elements in *basic-domain*[2] (the `Pitch` domain), j is set to 4 in line 13. Variable *basic-element* is initially assigned the `Pitch` element

$$\langle\langle S,\ 0,\ \texttt{Pitch},\ \langle \mathrm{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Pitch},\ \langle \mathrm{F2}\rangle\rangle\rangle$$

in line 14. This time, since the constituent viewpoint is `ScaleDegree`, in line 15 *derived-element* is assigned the element

$$\langle\langle S,\ 0,\ \texttt{ScaleDegree},\ \langle 5\rangle\rangle,\ \langle B,\ 0,\ \texttt{ScaleDegree},\ \langle 2\rangle\rangle\rangle.$$

In line 16, *inner-counter* is set to 1; and having been initialised as -1 in line 11, e is incremented to 0 in line 17. The predicate in line 18 is false; therefore line 20 is executed next. The predicate in this line is false, because F3 followed by F2[7] is not compatible with a `Cont` value of T; therefore line 22 is executed, in which the contents of *domain*[0][1] are replaced with *undef*. The **for** loop beginning on line 14 is executed a further four times, thereby running through the rest of the `Pitch` domain. The predicate in line 20 is true on only one of these passes (when F3 is followed by F3), in which case the contents of *domain*[3][1] are replaced with $\langle B,\ 0,\ \texttt{Cont}\otimes\texttt{ScaleDegree},\ \langle T,\ 2\rangle\rangle$. The **for** loop beginning on line 12 is executed one more time, giving a provisional linked domain of

$$\begin{aligned}
&\{\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \text{A}\flat 4\rangle\rangle,\ \textit{undef}\rangle,\\
&\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \text{A}\flat 4\rangle\rangle,\ \textit{undef}\rangle,\\
&\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \text{A}\flat 4\rangle\rangle,\ \textit{undef}\rangle,\\
&\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \text{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont}\otimes\texttt{ScaleDegree},\ \langle T,\ 2\rangle\rangle\rangle,\\
&\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \text{A}\flat 4\rangle\rangle,\ \textit{undef}\rangle,\\
&\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \text{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont}\otimes\texttt{ScaleDegree},\ \langle F,\ 2\rangle\rangle\rangle,\\
&\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \text{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont}\otimes\texttt{ScaleDegree},\ \langle F,\ 0\rangle\rangle\rangle,\\
&\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \text{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont}\otimes\texttt{ScaleDegree},\ \langle F,\ 1\rangle\rangle\rangle,\\
&\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \text{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont}\otimes\texttt{ScaleDegree},\ \langle F,\ 2\rangle\rangle\rangle,\\
&\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \text{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont}\otimes\texttt{ScaleDegree},\ \langle F,\ 3\rangle\rangle\rangle\}.
\end{aligned}$$

The provisional domain, constructed in this way in order to ensure that symbols which do not correspond with each other are not linked, may contain elements tagged as undefined, and may also contain duplicate elements; therefore the final inter-layer linked viewpoint domain is achieved once undefined and duplicate elements have been removed (lines 23 and 24):

$$\begin{aligned}
&\{\langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \text{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont}\otimes\texttt{ScaleDegree},\ \langle T,\ 2\rangle\rangle\rangle,\\
&\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \text{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont}\otimes\texttt{ScaleDegree},\ \langle F,\ 0\rangle\rangle\rangle,\\
&\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \text{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont}\otimes\texttt{ScaleDegree},\ \langle F,\ 1\rangle\rangle\rangle,\\
&\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \text{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont}\otimes\texttt{ScaleDegree},\ \langle F,\ 2\rangle\rangle\rangle,\\
&\ \langle\langle S,\ 0,\ \texttt{Duration}\otimes\texttt{Pitch},\ \langle 24,\ \text{A}\flat 4\rangle\rangle,\ \langle B,\ 0,\ \texttt{Cont}\otimes\texttt{ScaleDegree},\ \langle F,\ 3\rangle\rangle\rangle\}.
\end{aligned}$$

[7] It is assumed that previous predictions have global scope; that is, they can be accessed from anywhere.

4.6 Time Complexity Analysis

To demonstrate the utility of reducing domain size, we have carried out an empirical time complexity analysis on version 1 prediction runs. This involved running the computer model with different numbers of viewpoints, different sizes of corpus, and different types of $(\texttt{Pitch})_{SATB}$ domain (seen, augmented and full).[8] Seen $(\texttt{Duration})_{SATB}$ and $(\texttt{Cont})_{SATB}$ domains are used throughout, which are small enough to be neglected for the purposes of this analysis. During the course of this exercise, we have also determined how $(\texttt{Pitch})_{SATB}$ domain size varies with corpus/test data size for the seen, augmented and full domain cases. It is with this that we shall begin.

4.6.1 Variation of Domain Size with Corpus/Test Data Size

Seen, augmented and full $(\texttt{Pitch})_{SATB}$ domains were computed using 19 different combined domain and test data sizes between 1 and 110 hymns. For each of these sizes (except 110, which comprised all available data), there were three randomly selected sets of hymns; since the hymns vary in length, this greatly increased the number of different corpus/test data sizes in terms of the number of events. A plot of number of events in the corpus/test data against seen $(\texttt{Pitch})_{SATB}$ domain size (see Figure 4.5) shows a rapid discovery of novel chords at very low corpus/test data sizes. The discovery rate tails off with increasing corpus/test data size until the size reaches about 2,000 events, above which there appears to be a linear relation. Given sufficiently large corpus/test data sizes, we would expect the discovery rate to decline, leading to the relationship becoming asymptotic.

A similar plot for the augmented $(\texttt{Pitch})_{SATB}$ domain has much the same shape (see Figure 4.6); but the data points are more dispersed, since the domain size depends on both the number of novel chords and the domain sizes of the individual parts (SATB). The augmented domain is about five times larger than the seen domain. We would expect the augmented domain relationship to become asymptotic at lower corpus/test data sizes than that for the seen domain.

Finally, Figure 4.7 is a similar plot for the full $(\texttt{Pitch})_{SATB}$ domain. In this case, there is initially an extremely rapid increase in domain size with corpus/test data size, followed by a rapid decline in the increase, followed by a long tail with relatively little increase (*i.e.*, the relationship is already close to asymptotic). The data points are very dispersed, due to the fact that domain size depends solely on the domain sizes of the individual parts. A log curve seems to fit this data best, which is not the case for the seen and augmented data.

[8] Maximum N-gram order also influences run time, but this has not yet been investigated; all runs used a maximum N-gram order of 3.

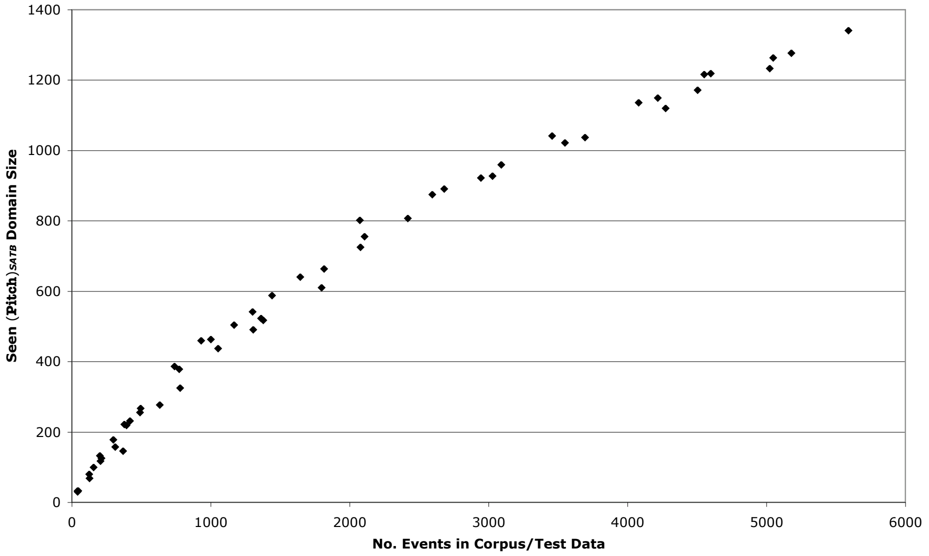


Figure 4.5: Plot of number of events in the corpus/test data against seen (Pitch)$_{SATB}$ domain size.

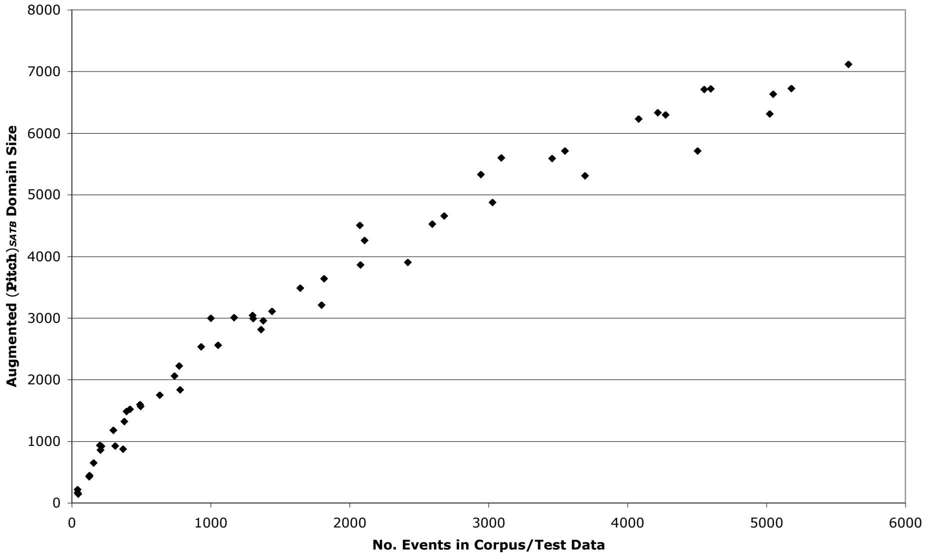


Figure 4.6: Plot of number of events in the corpus/test data against augmented (Pitch)$_{SATB}$ domain size.

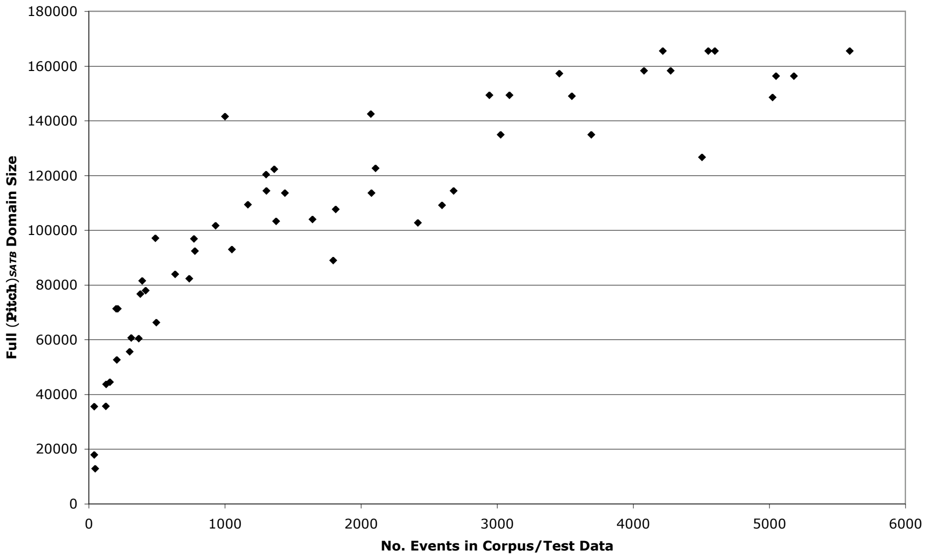


Figure 4.7: Plot of number of events in the corpus/test data against full $(\texttt{Pitch})_{SATB}$ domain size.

4.6.2 Effect of Domain Size on Program Running Time

Figure 4.8 is a log–log plot of $(\texttt{Pitch})_{SATB}$ domain size against time for the learning phase of the program, which was run using a corpus of 30 hymns and multiple viewpoint systems comprising 2 and 10 viewpoints. There are three domain sizes, corresponding to the seen, augmented and full domains. The seen domain causes the program to run about two orders of magnitude faster than the full domain. Encouragingly, the use of the augmented domain results in a running time which is not too much slower than that of the seen domain.

For the prediction phase of the program (`Duration`, `Cont` and `Pitch` prediction), the relative differences in running time are even greater: the seen domain causes the program to run about three orders of magnitude faster than the full domain (see Figure 4.9). The running time of the program using the augmented domain is still fairly close to that using the seen domain.

We have demonstrated the utility of reducing the $(\texttt{Pitch})_{SATB}$ domain size. The full $(\texttt{Pitch})_{SATB}$ domain causes the program to run very slowly even for the prediction of only one short harmonisation. For viewpoint selection runs, which involve ten-fold cross-validation of the corpus, the situation is far worse. During the course of such a run, many multiple viewpoint systems are tried. For each of these systems, the learning phase is run ten times, and every harmonisation in the corpus is predicted. Since it is clearly not practical to use the full domain, further analysis will concentrate on the use of the seen and augmented domains. Time complexities will be derived in terms of number of viewpoints, size of corpus and size of test data, to give prospective developers of similar programs an idea of their effect on running time.

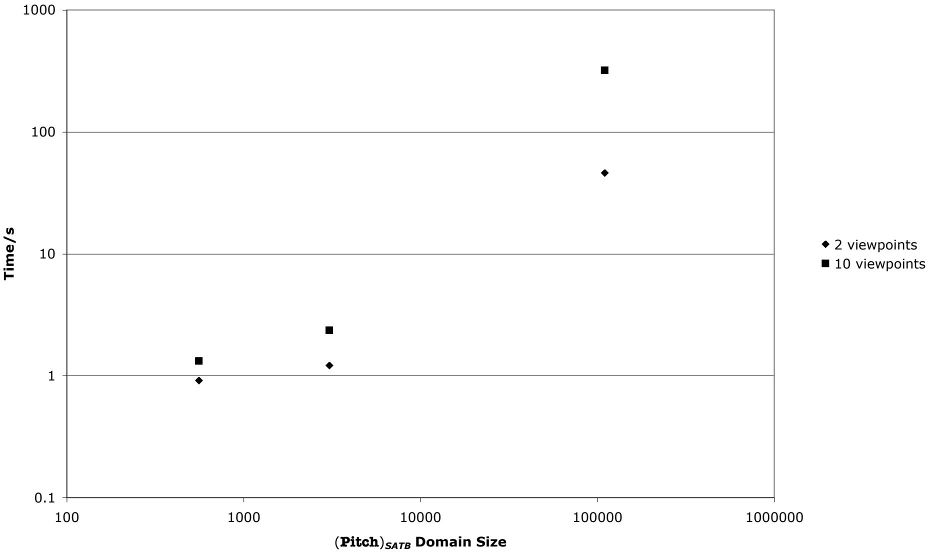


Figure 4.8: Log–log plot of (Pitch)$_{SATB}$ domain size against time for the learning phase of the program, which was run using a corpus of 30 hymns and multiple viewpoint systems comprising 2 and 10 viewpoints.

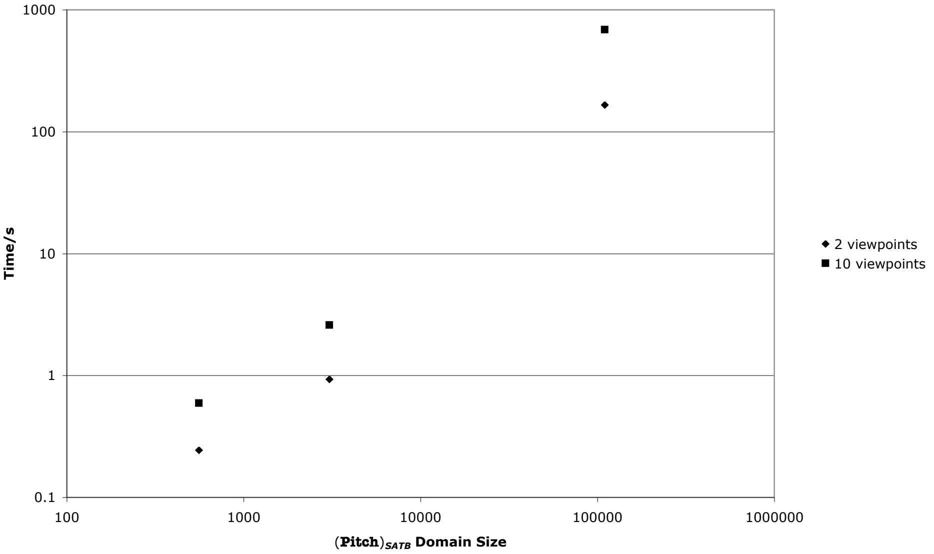


Figure 4.9: Log–log plot of (Pitch)$_{SATB}$ domain size against time for the prediction phase of the program, which was run using a corpus of 30 hymns and multiple viewpoint systems comprising 2 and 10 viewpoints. A single hymn tune harmonisation comprising 33 events was predicted.

4.6.3 Empirical Time Complexity Analysis Using Seen and Augmented Domains

In this more detailed analysis, the learning phase is split into domain construction and model construction phases. First of all, it is appropriate to explain why we have decided to derive time complexities empirically rather than analytically. The domain construction phase should be the easiest to analyse, since the number of viewpoints is irrelevant: we are only constructing the three basic domains during this phase. We can simplify things further by neglecting the `Duration` and `Cont` domains, which are very small in comparison with that of `Pitch`. The procedure is to traverse the corpus and test data, adding previously unseen elements to the `Pitch` domain. The traversal time, ignoring domain processing time, is proportional to the number of events in the corpus and test data; but we cannot ignore domain processing. For each event in the corpus and test data, its `Pitch` element must, in the worst case, be compared with every element in the `Pitch` domain, which increases in size throughout this process. The rate of increase and ultimate size is much greater for the augmented domain than for the seen domain. In neither case, however, can this information be accurately determined *a priori* for any given corpus and test data (although inspection of Figures 4.5 and 4.6 can give a rough idea for this style of music); we consider it better, therefore, to derive the time complexity empirically.

Having said that, we know that in this particular case there are only 882 elements in the seen domain, compared with 2,601 events in the corpus and test data (a ratio of 0.34). If we define n_c to be the number of events in the corpus and n_{td} to be the number of events in the test data, at worst the time complexity for the seen domain is $O((n_c+n_{td})^{1.34})$. On the other hand, there are 5,040 elements in the augmented domain; therefore in this case we would expect a complexity of $O((n_c + n_{td})^{2.94})$ at worst.

Since time complexities for the model construction and prediction phases are even more difficult to derive analytically (*e.g.*, we cannot determine the size and structure of the viewpoint models *a priori*), we have made no attempt to do so. Let us now look at the empirical analyses.

4.6.3.1 Domain Construction Phase

Figure 4.10 shows a plot of number of viewpoints against time for the seen domain construction phase of the program, which was run using corpora of 5, 10, 15, 20, 25 and 30 hymns. Each data point has a mean time of ten runs, each run using a different randomly selected multiple viewpoint system capable of predicting `Duration`, `Cont` and `Pitch` (the ten runs vary widely in duration). Straight lines fit the data reasonably well. All six lines are close to horizontal; therefore it has been concluded that, in reality, the run times are constant with respect to the number of viewpoints. This makes sense, considering that only the three basic viewpoint domains are constructed during this phase. A similar plot for the augmented domain (see Figure 4.11) also leads us to

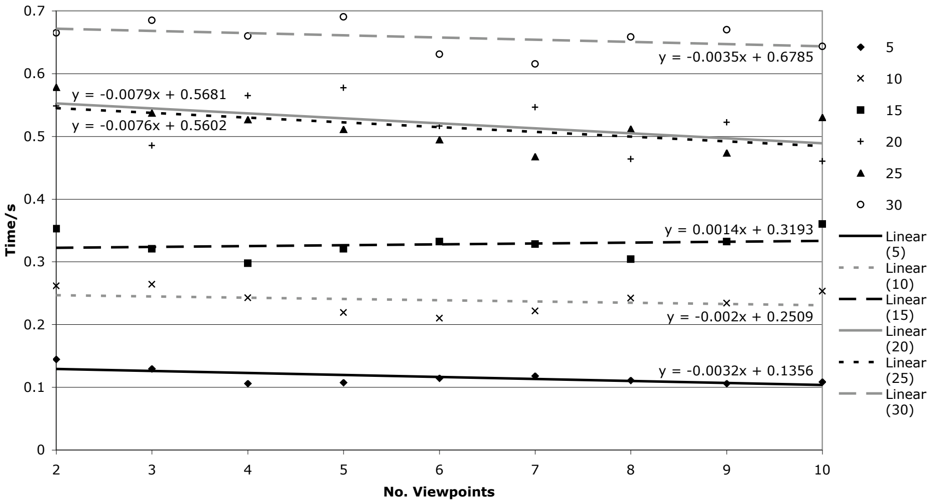


Figure 4.10: Plot of number of viewpoints against time for the seen domain construction phase of the program, which was run using corpora of 5, 10, 15, 20, 25 and 30 hymns.

the conclusion that the times are constant with respect to the number of viewpoints. The times are taken to be those in the centre of the fitted lines; that is, at the 6 viewpoint mark. Two other data sets were also generated and analysed, using different sets of hymns for each corpus size (resulting in different numbers of corpus events). The graphs, which are not shown here, are similar to those for the first data set.

Bearing in mind that the test data (in addition to the corpus) is taken into account when constructing domains, we are now in a position to plot the number of events in the corpus and test data against time; see Figure 4.12, which makes use of all three of the generated data sets. The data is rather sparse and scattered; but a straight line seems to produce a reasonable fit for the seen domain, and a quadratic curve a reasonable fit for the augmented domain (both fits are constrained to extend to the origin). The fact that time increases more rapidly with increasing number of events for the augmented domain is not unreasonable, since there is additional processing involved in its construction.

From the foregoing we gather that the time complexity of the domain construction phase of the program is at worst $O((n_c + n_{td})^2)$. In other words, run time is proportional to the square of the number of events in the corpus and test data. This is better than the semi-analytically derived worst case complexity of $O((n_c + n_{td})^{2.94})$, which necessarily made use of simplifying assumptions.

4.6.3.2 Model Construction Phase

Figure 4.13 shows a plot of number of viewpoints against time for the model construction phase of the program, which was run using the seen $(\texttt{Pitch})_{SATB}$ domain and corpora of 5, 10, 15, 20, 25 and 30 hymns. Again, straight lines fit the data reasonably well; but in this case there is a definite increase in run time with increasing number of viewpoints, which is due to the fact that a separate model needs to be built for each viewpoint. A

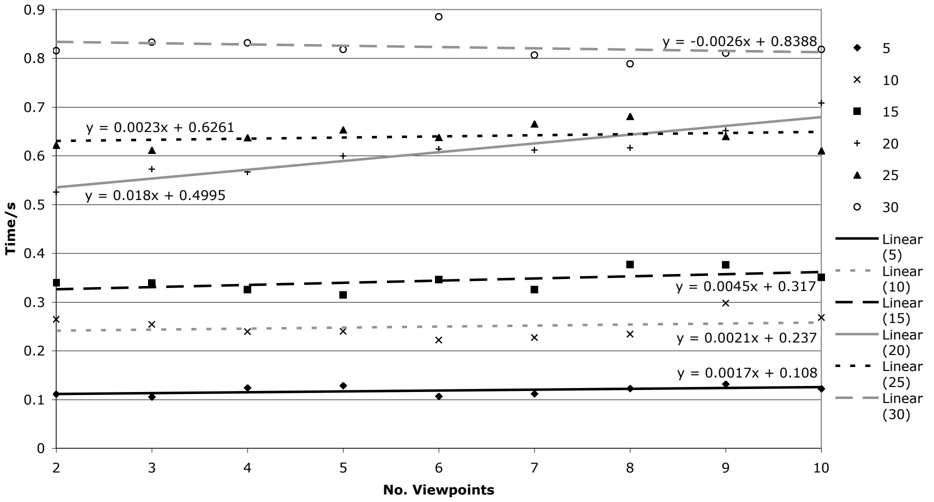


Figure 4.11: Plot of number of viewpoints against time for the augmented domain construction phase of the program, which was run using corpora of 5, 10, 15, 20, 25 and 30 hymns.

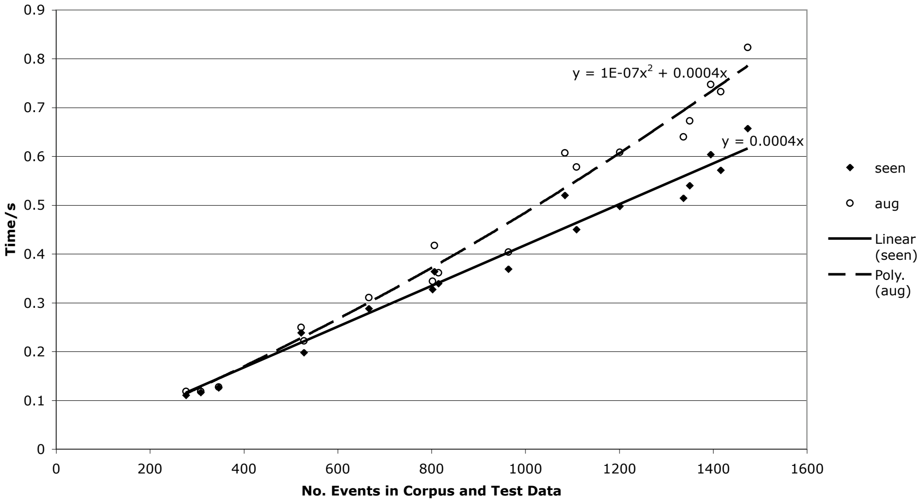


Figure 4.12: Plot of number of events in the corpus and test data against time for the seen and augmented domain construction phases of the program.

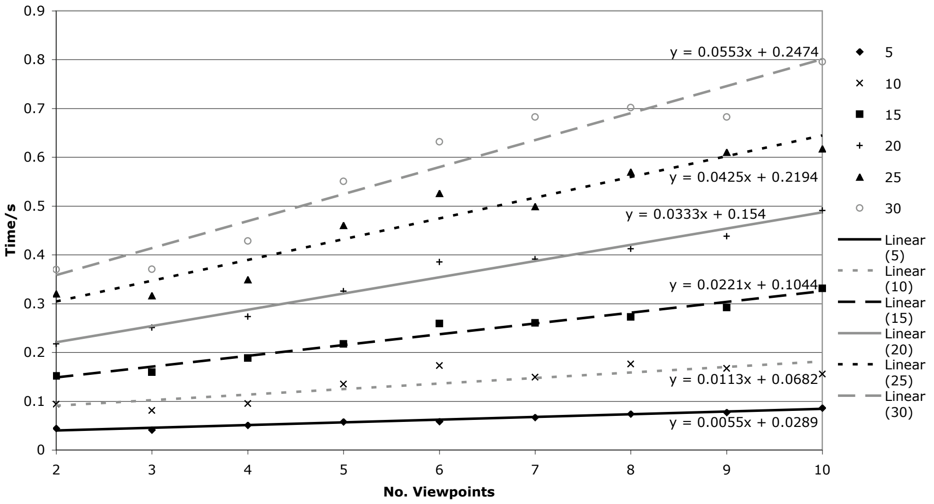


Figure 4.13: Plot of number of viewpoints against time for the model construction phase of the program, which was run using the seen $(\texttt{Pitch})_{SATB}$ domain and corpora of 5, 10, 15, 20, 25 and 30 hymns.

similar plot for the augmented $(\texttt{Pitch})_{SATB}$ domain (see Figure 4.14) shows a larger increase in run time with increasing number of viewpoints. This is almost certainly not due to an increase in the time required to construct the models from the corpus *per se*, but rather to the time needed for other processing involving domains in this part of the program. Again, two other data sets were also generated and analysed, with similar results.

We can now plot the number of of events in the corpus against time; see Figure 4.15. The times are taken from the fitted lines on the plots described above rather than from the original data points. In this case, the best fit to the data is a quadratic curve (constrained to extend to the origin); but the trend is close to linear for the range of data analysed. As expected, run time increases with the number of events in the corpus: there is a more rapid increase in time with increasing number of viewpoints, and the augmented domain gives rise to longer run times than the seen domain.

If we define v to be the number of viewpoints, then from the above we conclude that the time complexity of the model construction phase of the program is $O(vn_c^2)$. In other words, run time is proportional to the number of viewpoints, and also proportional to the square of the number of events in the corpus. For the range of data analysed here, however, we can say that run time is approximately proportional to n_c.

4.6.3.3 Prediction Phase

Figure 4.16 shows a plot of number of viewpoints against time for the prediction phase of the program, which was run using the seen $(\texttt{Pitch})_{SATB}$ domain and corpora of 5, 10, 15, 20, 25 and 30 hymns. Once more, there is a reasonably good linear fit to the data, with increasing run time resulting from increasing number of viewpoints. The reason for this

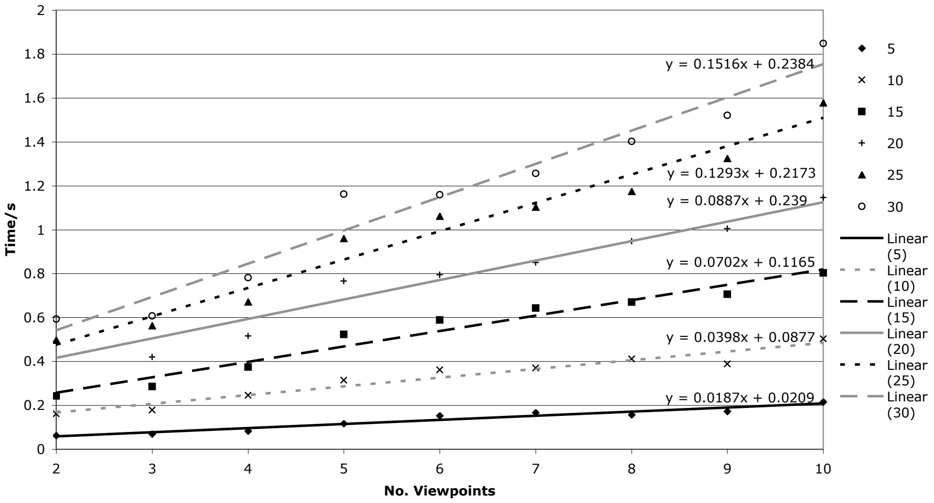


Figure 4.14: Plot of number of viewpoints against time for the model construction phase of the program, which was run using the augmented (`Pitch`)$_{SATB}$ domain and corpora of 5, 10, 15, 20, 25 and 30 hymns.

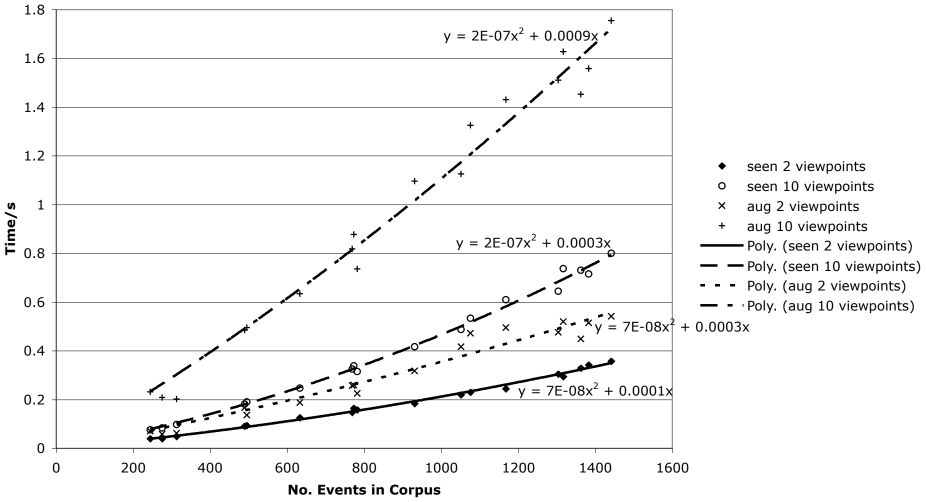


Figure 4.15: Plot of number of events in the corpus against time for the model construction phase of the program, using seen and augmented (`Pitch`)$_{SATB}$ domains and multiple viewpoint system sizes of 2 and 10 viewpoints.

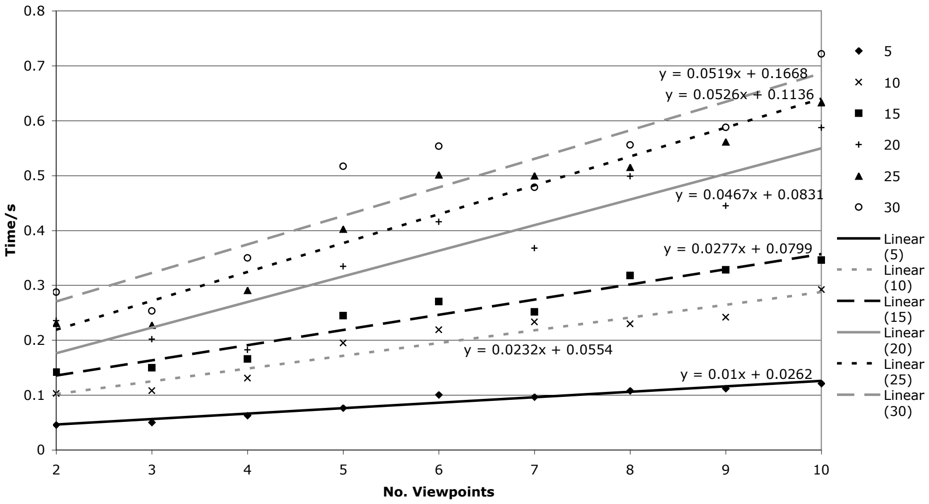


Figure 4.16: Plot of number of viewpoints against time for the prediction phase of the program, which was run using the seen $(\texttt{Pitch})_{SATB}$ domain and corpora of 5, 10, 15, 20, 25 and 30 hymns.

is that more prediction probability distributions need to be constructed and combined as the number of viewpoints increases. A similar plot for the augmented $(\texttt{Pitch})_{SATB}$ domain (see Figure 4.17) shows a larger increase in run time with increasing number of viewpoints. Once more, two other data sets were also generated and analysed, with similar results.

Figure 4.18 shows a plot of the number of events in the the corpus against time (taken from the fitted lines on the plots described above). The data is best fitted using straight lines, which have been constrained to extend to the origin. Again, run time increases with the number of events in the corpus, and the effect of domain type and number of viewpoints is similar to that in the model construction phase.

We can be fairly certain, without doing experiments, that the run time of this phase is proportional to the number of events in the test data being predicted. This, combined with the above analysis, results in a time complexity of $O(vn_cn_{td})$ for the prediction phase of the program. In other words, run time is proportional to the number of viewpoints, the size of the corpus, and the size of the test data.

4.6.3.4 Complete Prediction Run

The time complexity of a complete prediction run is the same as the worst complexity of the three constituent phases, which is $O(vn_c^2)$ as for model construction.

4.6.4 Time complexity Analysis of Viewpoint Selection

For viewpoint selection, an important factor in determining run time is the number of primitive viewpoints v_p in a pool to be used in the stepwise optimisation of the multiple viewpoint system. At any viewpoint addition stage, single additions are made to the

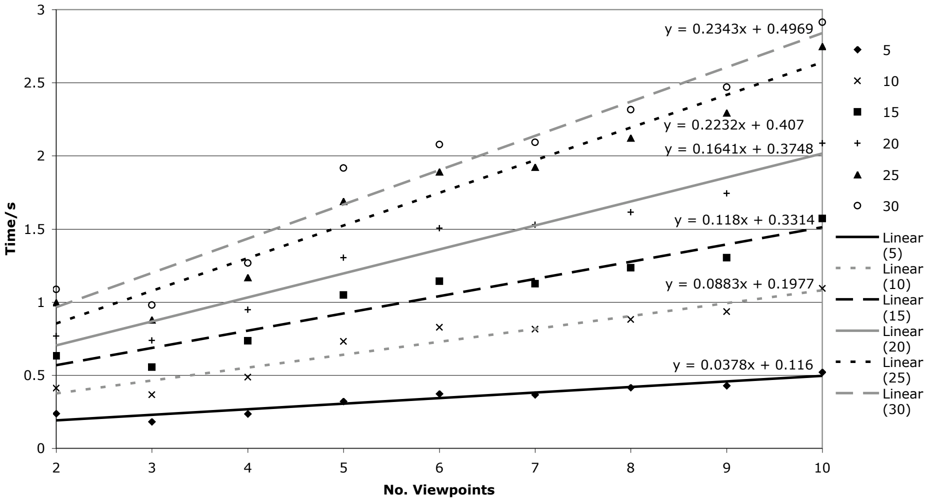


Figure 4.17: Plot of number of viewpoints against time for the prediction phase of the program, which was run using the augmented $(\texttt{Pitch})_{SATB}$ domain and corpora of 5, 10, 15, 20, 25 and 30 hymns.

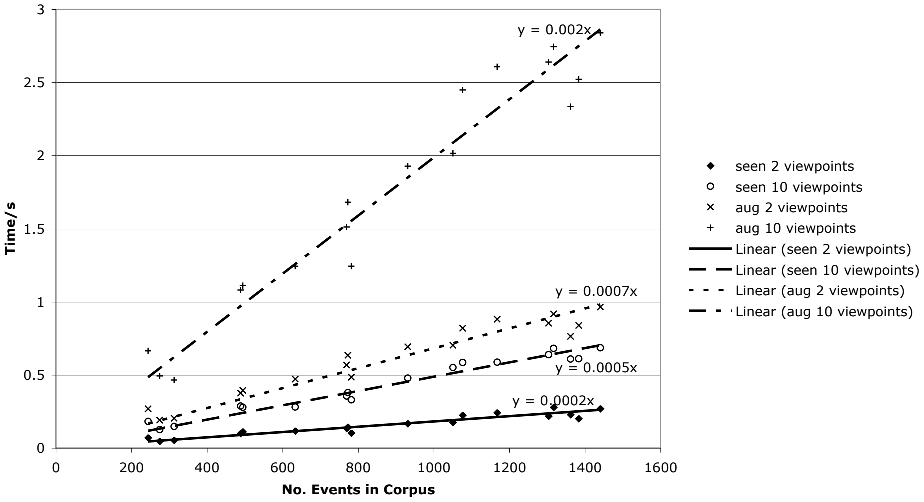


Figure 4.18: Plot of number of events in the corpus against time for the prediction phase of the program, using seen and augmented $(\texttt{Pitch})_{SATB}$ domains and multiple viewpoint system sizes of 2 and 10 viewpoints.

current multiple viewpoint system, and each of these new systems is evaluated using ten-fold cross-validation. In the limiting case, each primitive viewpoint in turn is added to the current system on its own and as part of a link with primitive viewpoints already in the system. This means that an evaluation is run the same number of times (at most) for each primitive viewpoint in the pool, which in turn means that run time is proportional to v_p (with a very large coefficient). This is only approximately true, however, since the optimisation path can change dramatically with the addition of a primitive viewpoint to the pool; it could, for example, result in a different number of viewpoint addition stages, which cannot be predicted in advance. Furthermore, the limiting case for v is the number of viewpoints in the multiple viewpoint system at the conclusion of selection, which also cannot be predicted in advance. With these provisos in mind, the time complexity of a viewpoint selection run is $O(v_p v n_c^2)$, where v is the number of viewpoints in the ultimately selected multiple viewpoint system.

4.6.5 Time Complexity of Versions 2 and 3

Although no analysis has yet been done of the time complexity of versions 2 and 3, we anticipate that it will be broadly the same as for version 1 except for the additional factor of prediction in stages. If, for example, the alto, tenor and bass parts are predicted one after the other, the version 2 run time is likely to be approximately three times that for prediction of ATB together. For version 3, each each successive prediction stage will take longer than the preceding one. Other than that, coefficients and constants are likely to be higher in versions 2 and 3 (especially 3), thereby contributing to longer run times.

4.7 Conclusions

In the context of the multiple viewpoint framework, a viewpoint domain is the set of valid elements (or symbols, or values) for a viewpoint, which is a means of representing a musical attribute such as pitch. We have discussed three issues affecting domains, and described how to construct them for increasingly complex viewpoints, culminating in a formal procedure for the construction of the most complex viewpoint domains.

Firstly, very large domains, like that of $(\texttt{Pitch})_{SATB}$ in version 1, can be vastly reduced in size, thereby greatly reducing run time, by only including elements seen in the corpus and test data. To take better account of elements as yet unseen, this domain can be augmented by chord transpositions occurring within the bounds of the known part ranges. If a part is given, such as the soprano in version 1, the domain is further constrained by the attributes of the given note.

Secondly, it has been known for some time that it is not possible in general to fix derived viewpoint domains: they need to be specially constructed before each prediction such that between them, the members of the domain are able to predict all of, and only, the members of the basic viewpoint domain.

Thirdly, we have shown that linked viewpoint domains are not, in general, constructed simply by taking the Cartesian product of the constituent viewpoint domains. This is due to the correspondences between basic and derived viewpoints, and between the basic viewpoints `Cont` and `Pitch`: many of the tuples in the Cartesian product make no logical sense.

We have described how to reliably construct domains for melody alone, and for three versions of the framework for representing and modelling harmony that we have developed. The domain construction method outlined for the most complex version has been implemented, and so far found to be suitably robust. The method can easily be extended beyond the prediction of three basic viewpoints.

We have carried out an empirical analysis of the time complexity of version 1 of our computer model, demonstrating the effect of number of viewpoints, corpus size and type of domain (seen, augmented or full). The time complexity of a complete prediction run is $O(vn_c^2)$, and that of a viewpoint selection run is $O(v_pvn_c^2)$ (with certain provisos; see §4.6.4), where n_c is the number of events in the corpus, v is the number of viewpoints in the multiple viewpoint system and v_p is the number of primitive viewpoints in the pool used during viewpoint selection.

Chapter 5

Prediction Performance Analysis of Version 0

5.1 Introduction

In this chapter we systematically search for the best performing version 0 (melody only) long-term model (LTM), short-term model (STM), and combined long- and short-term model (BOTH) for the prediction of `Duration` and `Pitch` together and separately. The effects of using different corpora are also investigated. To avoid repeated lengthy descriptions, an LTM which is updated after each note has been predicted is termed LTM+. Similarly, when such a model is used in combination with an STM, the combination is termed BOTH+. Maximum N-gram order is denoted by $\hbar$, which should not be confused with mean cross-entropy, $\bar{H}$. We aim to provide independent corroboration of some of the findings of Pearce (2005) as well as addressing an unanswered question concerning the best way to combine long- and short-term models.

Since the terms "bias" and "L-S bias" are used often in this and following chapters, a brief review is given here. The idea of the bias is to give more weight to viewpoints which are capable of predicting with more certainty. A prediction set in which the probability distribution is completely uniform will predict with a great deal of uncertainty; the less uniform the distribution, the greater the certainty. What Conklin (1990) calls *relative entropy*, *Re*, is a measure of the uncertainty of a distribution with respect to maximum uncertainty. Lower *Re* indicates greater certainty, which warrants greater weighting; therefore an exponential bias b is introduced into the weighting function which favours distributions with low *Re*. The weighting of the i^{th} viewpoint is given by:

$$w_i = Re([\tau])^{-b}.$$

The L-S bias is similarly used to favour whichever of the LTM or STM has the greatest prediction certainty at any point in the prediction process. For more information on weighting, see §2.2.4.

Viewpoint selection method VS3 is used (see §3.4.5.2), which recognises that links with specified primitive or threaded viewpoints which have not necessarily been added to the viewpoint set are worth keeping under consideration with respect to linking with other viewpoints. The particular viewpoints kept under consideration here (arising from preliminary trials) are `DurRatio`, `Interval`, `ScaleDegree`, `ScaleDegree` $\ominus$ `FirstInPhrase` and `InScale`.

Different types of model capable of predicting both `Duration` and `Pitch` together are compared in §5.2. Firstly, there is a comparison of weighted geometric against weighted arithmetic viewpoint combination in the LTM, and then LTM+ is compared with LTM. Next, there is a comparison of weighted geometric against weighted arithmetic combination in the STM. This is followed by a comparison of three different ways of combining long- and short-term models using BOTH, and then BOTH+ is compared with BOTH. After this, there is a discussion about what the criterion should be for halting viewpoint selection. The section ends with an investigation into the effects of using different corpora. In §5.3, consideration is given to the prediction of `Duration` and `Pitch` separately, as well as further investigating the use of different corpora. Finally, the chapter is summarised and a conclusion given in §5.4.

5.2 Prediction of `Duration` and `Pitch` Together

In order to provide accurate comparisons between different charts and graphs at a glance, all such figures in this section have a cross-entropy range from 2.5 to 5.0 bits/note unless otherwise stated. Each cross-entropy value plotted is the mean of ten values, since tenfold cross-validation is used during viewpoint selection. All plots except Figure 5.5 (a three-dimensional surface plot) show standard errors.

5.2.1 LTM: Weighted Geometric vs. Weighted Arithmetic Combination

Viewpoint selection runs were carried out using the 50-hymn corpus 'A' with the objective of comparing weighted geometric and weighted arithmetic viewpoint combination. In the first set of runs, the bias was fixed at 0 and maximum N-gram order $\hbar$ varied from 0 to 5. In each case, once the multiple viewpoint system had been selected, the bias was optimised and the resulting cross-entropy recorded. It can be seen from Figure 5.1 that for both geometric and arithmetic combination an $\hbar$ of 2 produces the lowest cross-entropy, although we should be careful about this: there is a fair amount of overlap between the error bars, which means that we cannot be confident that an $\hbar$ of 2 really is best. On the other hand, the error bars indicate that we can be confident that weighted geometric combination results in lower cross-entropies across the board. In this chapter, the average cross-entropies will be used as a guide to performance, with the error bars serving as a reality check. Standard errors will be referred to only occa-

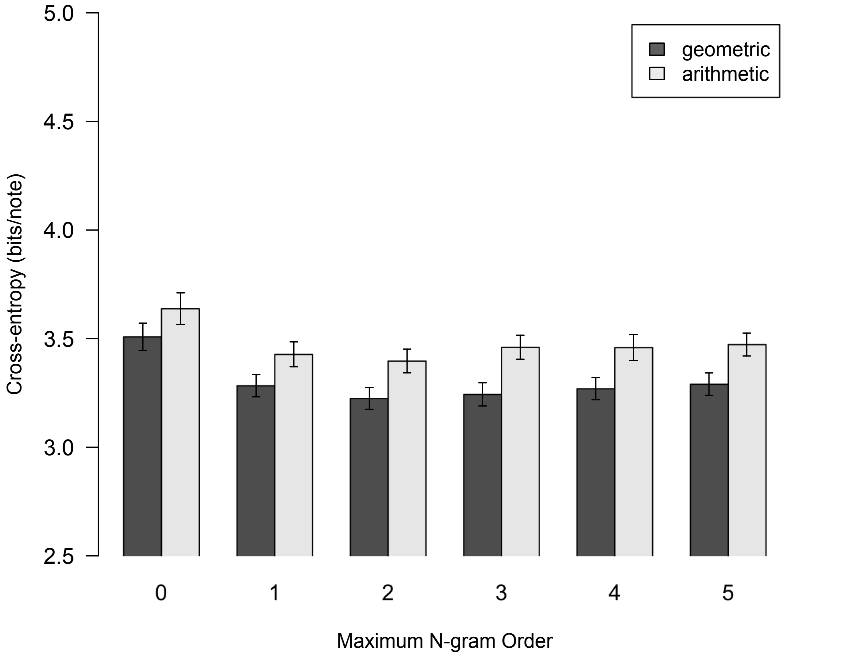


Figure 5.1: Bar chart showing how cross-entropy varies with maximum N-gram order, $\hbar$, for an LTM using corpus ‘A’ and a bias of 0 for viewpoint selection. Weighted geometric and weighted arithmetic viewpoint combination are compared.

sionally from now on. Pearce (2005) found that, for a BOTH+ model predicting `Pitch` only, changing arithmetic to geometric combination reduced cross-entropy from 2.131 to 2.045 bits/note. The more marked reduction in Figure 5.1 (3.40 to 3.23 bits/note for an $\hbar$ of 2) can be explained by making the reasonable assumption that `Duration` prediction performance is similarly improved.

In the next set of viewpoint selection runs, $\hbar$ was fixed at 2 and the bias varied from 0 to 9. Once again, the bias was optimised after selection (as is always the case). The results indicated that for the better performing of the combination methods, weighted geometric, the optimal bias for viewpoint selection purposes was likely to be between 1 and 2; therefore viewpoint selection was carried out in that range at intervals of 0.1 (see Figure 5.2). For weighted geometric combination the optimal bias for viewpoint selection is 1.4 (subsequent bias optimisation also results in a bias of 1.4), giving a cross-entropy of 3.21 bits/note. It can be seen from the graph that weighted arithmetic combination is far inferior, and so will no longer be considered for the LTM.

To finish off this investigation, the first set of runs was repeated, using weighted geometric combination, but with bias fixed at 1.4 during viewpoint selection. The results in Figure 5.3 confirm that an $\hbar$ of 2 is optimal, which corroborates the findings of Pearce (2005) with respect to an LTM (albeit one comprising the single viewpoint `Pitch`).

At this point, let us examine how cross-entropy varies with bias, given a multiple

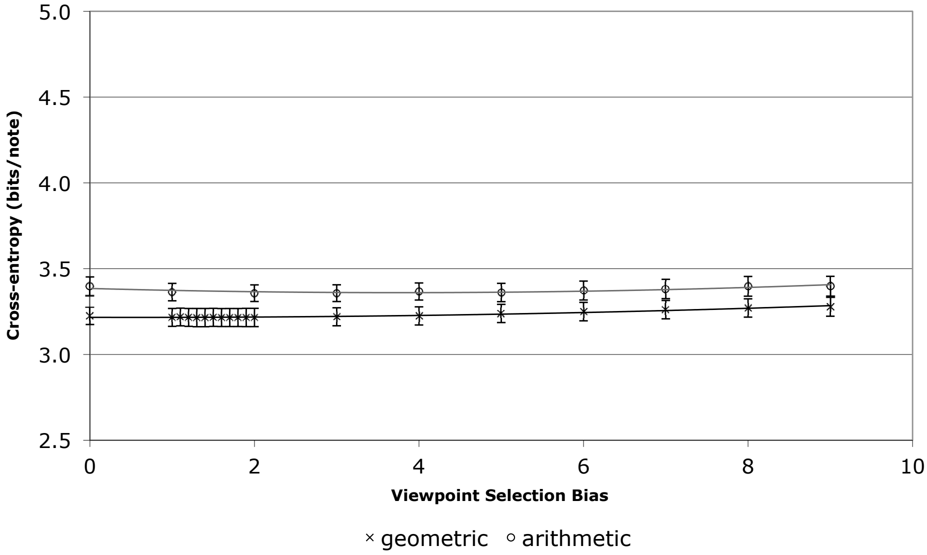


Figure 5.2: Plot of bias used during viewpoint selection against cross-entropy for an LTM using corpus 'A' and an $\hbar$ of 2. Weighted geometric and weighted arithmetic viewpoint combination are compared.

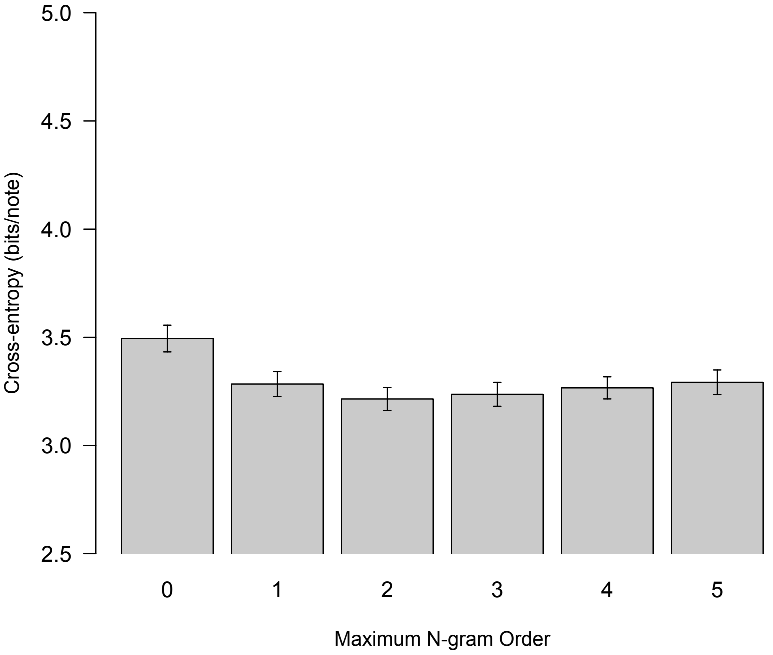


Figure 5.3: Bar chart showing how cross-entropy varies with $\hbar$ for an LTM using corpus 'A', weighted geometric viewpoint combination and a bias of 1.4 for viewpoint selection.

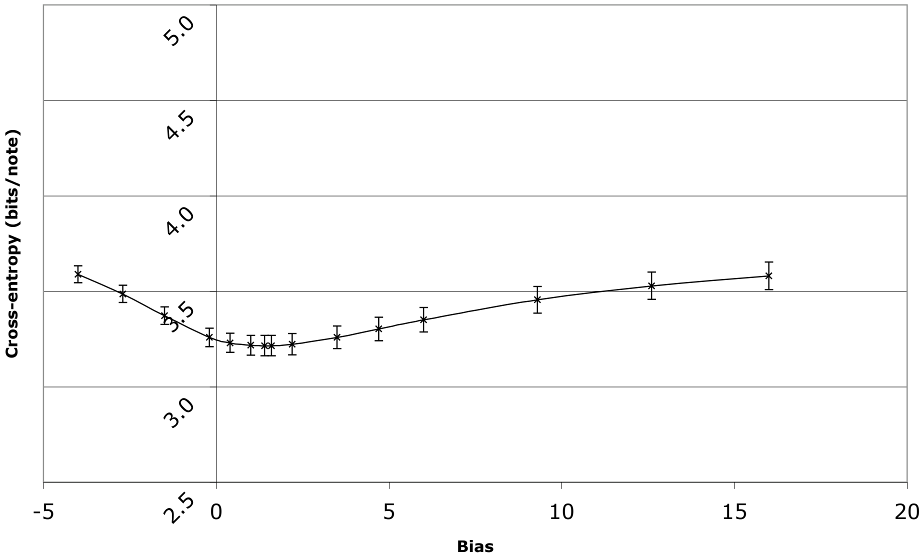


Figure 5.4: Plot of bias against cross-entropy for the best performing LTM using corpus 'A'.

viewpoint system. The data in Figure 5.4 is taken from the bias optimisation run for the best LTM system (some data points have been added to reveal a truer curve shape, while others close to the optimal point have been removed for the sake of clarity).

Before proceeding any further, we should consider whether the methodology used so far really finds the best combination of viewpoint selection bias and $\hbar$. It is conceivable that there is more than one local minimum cross-entropy within the search space, and that a different starting assumption (*e.g.*, a bias of 9 instead of 0) could result in the discovery of a different (possibly better) local minimum. In order to test this, the portion of the search space under consideration was thoroughly explored (using weighted geometric viewpoint combination), resulting in the three dimensional surface plot of Figure 5.5.[1] Please note the non-standard cross-entropy range. The shape of the surface is such that there is a single minimum cross-entropy, which means that the methodology used so far is suitable for this case, at least. Although a thorough search is preferable, time constraints mean that this is not feasible. It appears from this case that analysing one variable at a time is a reasonable approach, which has been employed in previous research. For example, in the assessment of variant techniques, Pearce and Wiggins (2004) typically used the best performing model from one experiment as the starting point for the next, knowing that the resulting model would not necessarily be globally optimal. This time-saving methodology will therefore continue to be used.

[1] Although maximum N-gram order has discrete values, it is assumed to be continuous here for ease of representation.

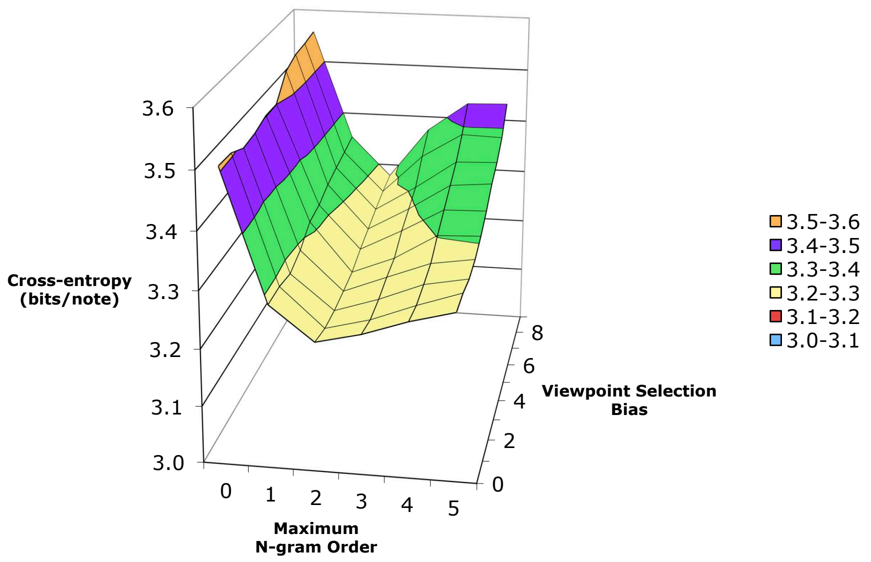


Figure 5.5: Three dimensional surface plot of $\hbar$ and bias used during viewpoint selection against cross-entropy (note the non-standard range) for an LTM using corpus 'A' and weighted geometric viewpoint combination.

5.2.2 LTM+ vs. LTM

Having established that geometric viewpoint combination is the better option for the LTM, we can be reasonably confident that this is also true for the LTM+, since the only difference between the models is that the corpus size gradually increases for the latter. Viewpoint selection runs were therefore carried out for LTM+ using corpus 'A', weighted geometric combination and a bias of 0 for viewpoint selection. Maximum N-gram order $\hbar$ was varied. Figure 5.6 compares the results of these runs with the LTM results obtained earlier. The updated model is generally much better than the model without update. At this stage, an $\hbar$ of 4 is optimal for the updated model.

In the next set of viewpoint selection runs for LTM+, $\hbar$ was fixed at 4 and the bias varied from 0 to 9. Figure 5.7 compares the results of these runs with the LTM results ($\hbar = 2$) obtained earlier. It can be seen that the updated model produces much lower cross-entropies across the board, with a bias of 2 being optimal for viewpoint selection.

To conclude this investigation, the first set of runs was repeated, but with bias fixed at 2 during viewpoint selection. The results in Figure 5.8 confirm that the updated model has far superior performance, and that an $\hbar$ of 4 is optimal for viewpoint selection; but only just, as an $\hbar$ of 3 produces an almost identical cross-entropy,[2] and an $\hbar$ of 5 results in a model which performs only very slightly worse. The best model uses a bias of 1.9 (after optimisation), giving a cross-entropy of 3.05 bits/note (*cf.* 3.21 bits/note when not updated). A reduction in cross-entropy was also found by Pearce (2005) for this comparison, using the single viewpoint system {`Pitch`}. Notice that the error bars in

[2] To be sure, an $\hbar$ of 3 was investigated further; but its performance was found to be very slightly inferior.

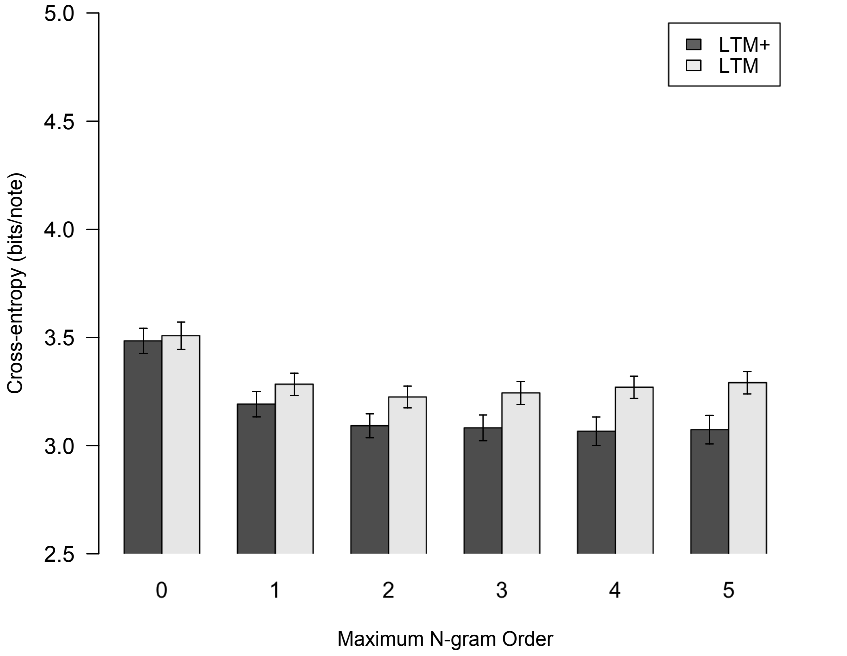


Figure 5.6: Bar chart showing how cross-entropy varies with $\hbar$ for LTM+ and LTM using corpus ‘A’, weighted geometric combination and a bias of 0 for viewpoint selection.

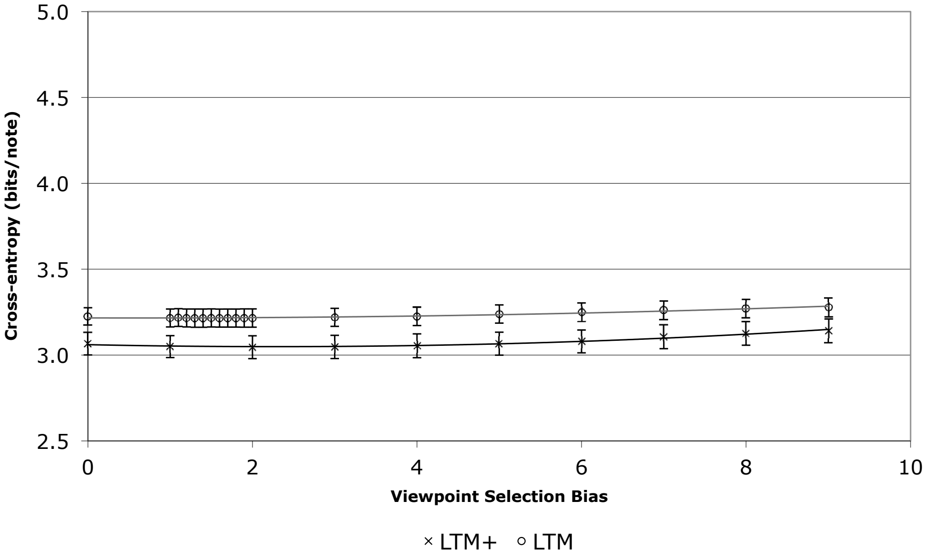


Figure 5.7: Plot of bias used during viewpoint selection against cross-entropy for LTM+ and LTM ($\hbar$ of 4 and 2 respectively) using corpus ‘A’ and weighted geometric combination.

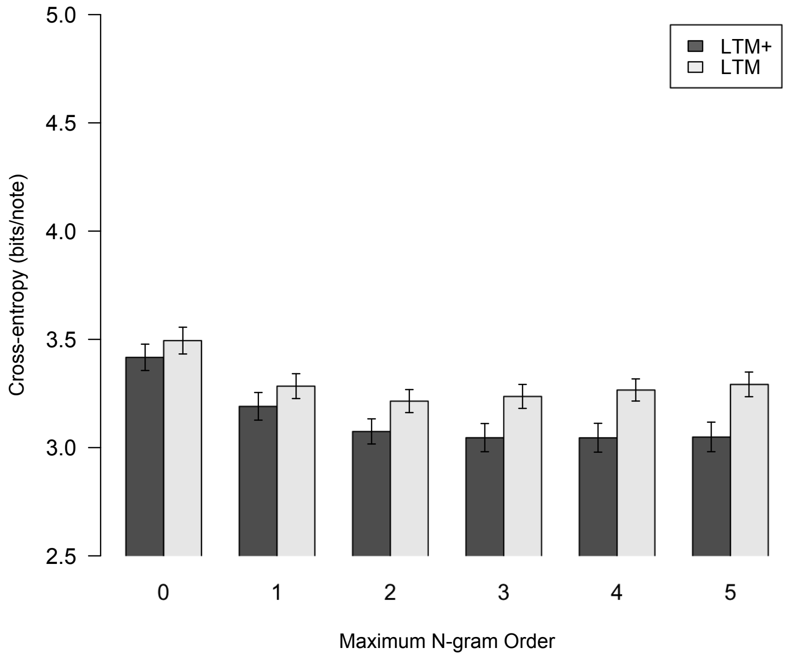


Figure 5.8: Bar chart showing how cross-entropy varies with $\hbar$ for LTM+ and LTM (viewpoint selection bias of 2 and 1.4 respectively) using corpus 'A' and weighted geometric combination.

Figure 5.8 have a fairly consistent size, which may seem suspicious. It is emphasised that the standard errors really are calculated.

It is no great surprise that the updated model produces lower cross-entropies, since its corpus size gradually increases as it predicts, thereby producing more accurate statistics. What might, on the face of it, seem surprising, however, is that the optimal value of $\hbar$ changes from 2 to 4, bearing in mind that during ten-fold cross-validation the corpus size increases from 45 hymns (without update) to a mean of only 47.5 hymns (with update). A likely reason for the increase in $\hbar$ is that any repeats in predicted tunes would be able to make use of long contexts from the earlier appearance of the musical material. While it is encouraging that the best multiple viewpoint systems for LTM+ and LTM have 22 viewpoints in common (total number of viewpoints: 34 and 30 respectively), it is interesting to note that the first five viewpoints selected for the LTM+ system (`ScaleDegree` $\otimes$ `Phrase`, `DurRatio` $\otimes$ `TactusPositionInBar`, `Interval` $\otimes$ (`ScaleDegree` $\ominus$ `Tactus`), `Duration` $\otimes$ `LastInPhrase` and `Interval` $\otimes$ `TactusPositionInBar`) do not appear in the LTM system at all. It is also interesting to note that all of the selected viewpoints are new to this research (although `ScaleDegree` $\otimes$ `Phrase` is essentially a combination of the previously available `ScaleDegree` $\otimes$ `FirstInPhrase` and `ScaleDegree` $\otimes$ `LastInPhrase`). See Table 5.1 for a complete listing of these multiple viewpoint systems. If the better performing of these viewpoints were added to the available pool for use in the construction of cognitive models of melodic expectancy (Pearce, 2005), it is

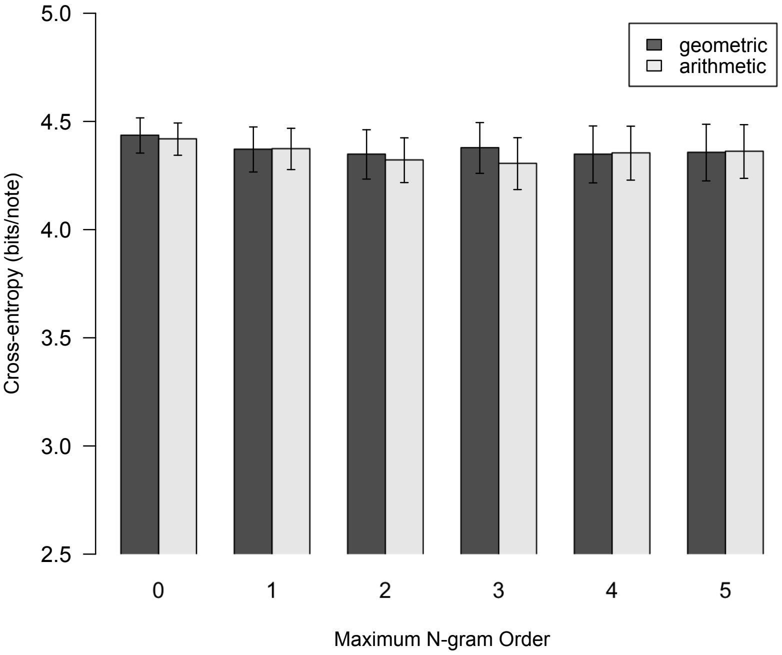


Figure 5.9: Bar chart showing how cross-entropy varies with $\hbar$ for an STM using corpus 'A' and a bias of 0 for viewpoint selection. Weighted geometric and weighted arithmetic viewpoint combination are compared.

conceivable that such models could be improved.

5.2.3 STM: Weighted Geometric vs. Weighted Arithmetic Combination

The procedure used in §5.2.1 is followed in the investigation of short-term models, beginning with 50-hymn corpus 'A' and varying $\hbar$. This time, with bias fixed at 0, an $\hbar$ of 4 is just found to be best for geometric combination, while an $\hbar$ of 3 is optimal for arithmetic combination; see Figure 5.9. At this point, weighted arithmetic combination appears to be the better option, although it is not so clear-cut as it was for the LTM.

Having fixed $\hbar$ at 4 for weighted geometric and 3 for weighted arithmetic combination, and varied the bias (see Figure 5.10), we find that arithmetic combination produces slightly lower cross-entropies across the board. The optimal bias for viewpoint selection is 3 in each case.

Next, viewpoint selection was carried out using a bias of 3, with $\hbar$ varied. Figure 5.11 shows that an $\hbar$ of 2 is optimal at this higher bias for both geometric and arithmetic combination. The model with the lowest cross-entropy overall is arithmetically combined; but there is very little difference in performance between the two combination methods.

To finish off this investigation, $\hbar$ was fixed at 2 for both weighted geometric and

Viewpoint	L+	L	B+	B
ScaleDegree $\otimes$ Tessitura	×	×	×	×
Duration $\otimes$ Metre	×	×	×	×
Pitch $\otimes$ ScaleDegree	×	×	×	×
(ScaleDegree $\ominus$ FirstInPhrase) $\otimes$ FirstInPiece	×	×	×	×
DurRatio $\otimes$ Phrase	×	×	×	×
IntFirstInPhrase $\otimes$ ScaleDegree	×	×	×	×
ScaleDegree $\otimes$ Metre	×	×	×	×
Duration $\otimes$ (ScaleDegree $\ominus$ LastInPhrase)	×	×	×	×
(ScaleDegree $\ominus$ FirstInPhrase) $\otimes$ Piece	×	×	×	×
IntFirstInBar $\otimes$ ScaleDegree	×	×	×	×
Interval $\otimes$ (ScaleDegree $\ominus$ LastInPhrase)	×	×	×	×
Duration $\otimes$ (ScaleDegree $\ominus$ FirstInPhrase)	×	×	×	×
(ScaleDegree $\ominus$ FirstInPhrase) $\otimes$ Tessitura	×	×	×	×
IntFirstInBar $\otimes$ (ScaleDegree $\ominus$ FirstInPhrase)	×	×	×	×
IntFirstInPiece $\otimes$ (ScaleDegree $\ominus$ FirstInPhrase)	×	×	×	
Interval $\otimes$ (ScaleDegree $\ominus$ FirstInBar)	×	×	×	×
Interval $\otimes$ FirstInBar	×	×	×	×
(ScaleDegree $\ominus$ FirstInPhrase) $\otimes$ SeqPositionInBar	×	×	×	×
(Pitch $\ominus$ Tactus) $\otimes$ (ScaleDegree $\ominus$ FirstInPhrase)	×	×	×	×
(ScaleDegree $\ominus$ FirstInPhrase) $\otimes$ FirstInBar	×	×	×	×
(ScaleDegree $\ominus$ FirstInPhrase) $\otimes$ (IOI $\ominus$ Tactus)	×	×		
(ScaleDegree $\ominus$ FirstInPhrase) $\otimes$ IOI	×	×		
ScaleDegree $\otimes$ Phrase	×		×	×
DurRatio $\otimes$ TactusPositionInBar	×		×	
Interval $\otimes$ (ScaleDegree $\ominus$ Tactus)	×			×
Duration $\otimes$ LastInPhrase	×		×	
Interval $\otimes$ TactusPositionInBar	×		×	×
Interval $\otimes$ Phrase	×			
Pitch $\otimes$ FirstInPhrase	×			×
Duration $\otimes$ PositionInBar	×		×	
Pitch $\otimes$ PositionInBar	×			
IntFirstInPiece $\otimes$ InScale	×			
Duration $\otimes$ (ScaleDegree $\ominus$ FirstInBar)	×			
(ScaleDegree $\ominus$ FirstInPhrase) $\otimes$ Metre	×			
Interval $\otimes$ LastInPhrase		×	×	×
ScaleDegree $\otimes$ Piece		×		
Pitch $\otimes$ LastInPhrase		×		
(Interval $\ominus$ FirstInPhrase) $\otimes$ InScale		×		
Interval $\otimes$ FirstInPhrase		×	×	×
ScaleDegree $\otimes$ Tactus		×		
Contour $\otimes$ ScaleDegree		×	×	×
Interval $\otimes$ (ScaleDegree $\ominus$ FirstInPhrase)		×	×	×
Pitch $\otimes$ TactusPositionInBar			×	
(Interval $\ominus$ FirstInPhrase) $\otimes$ (ScaleDegree $\ominus$ FirstInPhrase)			×	
(Interval $\ominus$ Tactus) $\otimes$ (ScaleDegree $\ominus$ FirstInPhrase)			×	

Table 5.1: Best version 0 multiple viewpoint systems (predicting Duration and Pitch) for LTM+ *(L+)*, LTM *(L)*, BOTH+ *(B+)* and BOTH *(B)*.

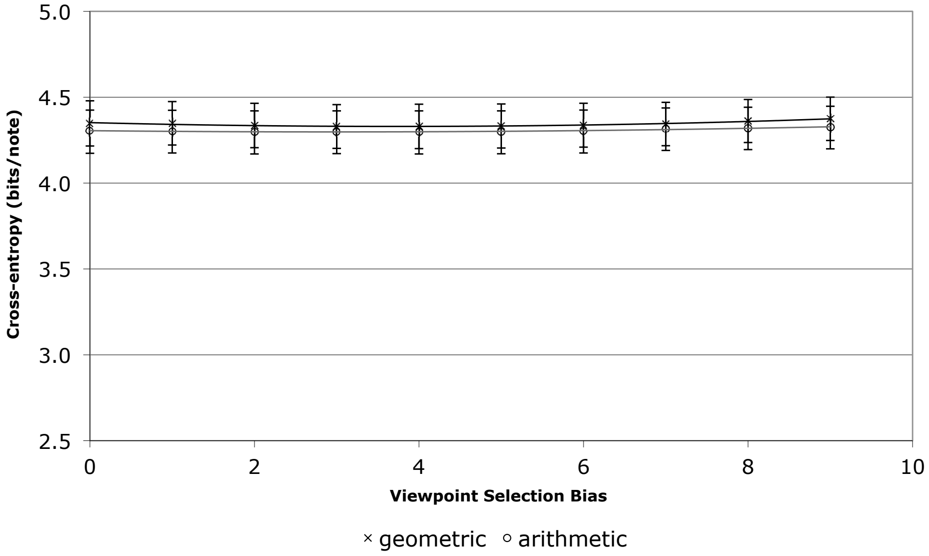


Figure 5.10: Plot of bias used during viewpoint selection against cross-entropy for an STM using corpus ‘A’. Weighted geometric and weighted arithmetic viewpoint combination are compared, using an $\hbar$ of 4 and 3 respectively.

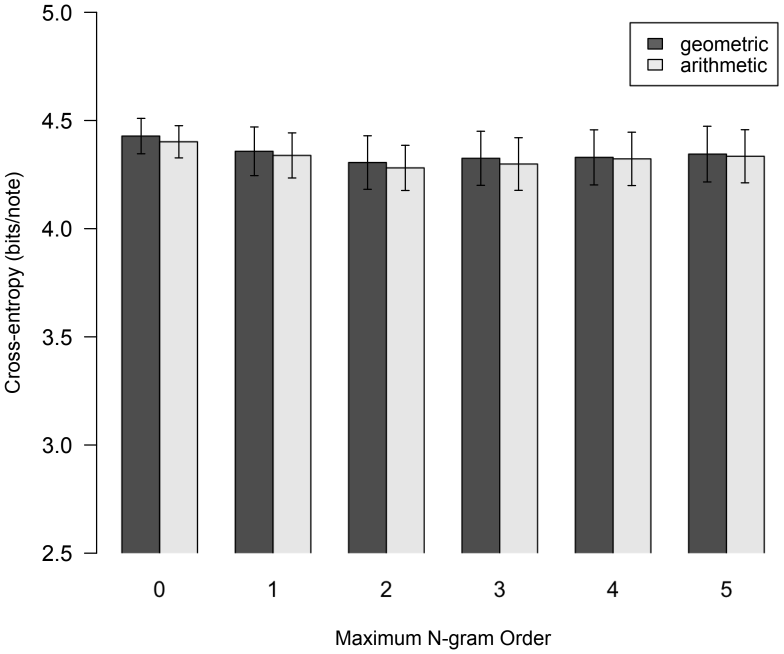


Figure 5.11: Bar chart showing how cross-entropy varies with $\hbar$ for an STM using corpus ‘A’ and a viewpoint selection bias of 3. Weighted geometric and weighted arithmetic viewpoint combination are compared.

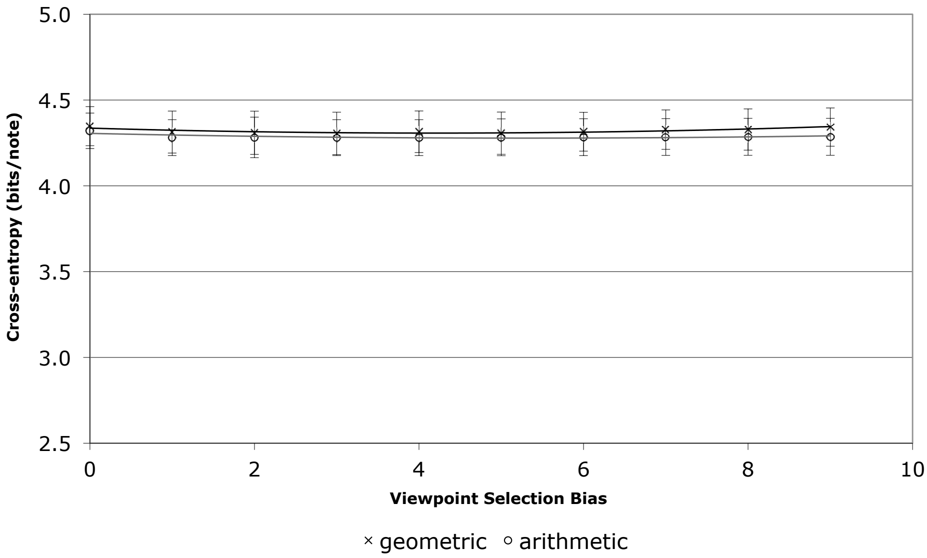


Figure 5.12: Plot of bias used during viewpoint selection against cross-entropy for an STM using corpus 'A' and an $\hbar$ of 2. Weighted geometric and weighted arithmetic viewpoint combination are compared.

weighted arithmetic combination. It can be seen from Figure 5.12 that arithmetic combination is again generally slightly better than geometric. The best model overall is arithmetically combined, with a bias for viewpoint selection of 3 and a bias after optimisation of 3.5, giving a cross-entropy of 4.28 bits/note.

The fact that an $\hbar$ of 2 is optimal is rather odd, bearing in mind that the best LTM+ has an $\hbar$ of 4 (see §5.2.2 above) and that Pearce (2005) found that an $\hbar$ of 5 was optimal for the single viewpoint system {`Pitch`} using escape method C. It was thought that the prediction of repeats in tunes would be able to make use of long contexts from the earlier appearance of the musical material; but the STM appears to be unable to do this effectively.

Since models using geometric and arithmetic combination have a similar performance, the best multiple viewpoint systems for each are recorded in Table 5.2. The arithmetically combined system is more parsimonious than the other (8 viewpoints *cf.* 13), with which it has all 8 viewpoints in common.

It is interesting to note that whereas here the best STM has a cross-entropy which is 1.07 bits/note higher than that of the LTM, this is apparently far from the case in Pearce (2005), where the difference between average cross-entropies is a mere 0.17 bits/note for models using escape method C with exclusion. Part of the reason for this is that Pearce (2005) is using the single viewpoint system {`Pitch`}. If we use the system {`Duration`, `Pitch`} instead of the systems arrived at by viewpoint selection, the difference reduces from 1.07 to 0.45 bits/note (although, of course, both cross-entropies increase). Bearing in mind that we are predicting `Duration` as well as `Pitch`, it is not unreasonable that the difference should be greater than 0.17 bits/note, especially considering the huge

Viewpoint	G	A
Interval ⊗ InScale	×	×
Duration ⊗ LastInPhrase	×	×
Duration ⊗ Metre	×	×
InScale ⊖ Tessitura	×	×
Duration ⊗ (ScaleDegree ⊖ FirstInPhrase)	×	×
Pitch ⊗ FirstInPhrase	×	×
(ScaleDegree ⊖ FirstInPhrase) ⊗ FirstInBar	×	×
(ScaleDegree ⊖ FirstInPhrase) ⊗ (IOI ⊖ FirstInBar)	×	×
InScale	×	
Interval ⊗ Phrase	×	
Duration ⊗ (Pitch ⊖ Tactus)	×	
ScaleDegree ⊖ FirstInPhrase	×	
(ScaleDegree ⊖ FirstInPhrase) ⊗ Metre	×	

Table 5.2: Best version 0 multiple viewpoint systems (predicting Duration and Pitch) for STM using geometric *(G)* and arithmetic combination *(A)*.

variation between data sets in Pearce (2005); for example, for data set 2 the STM is 0.61 bits/note higher, whereas for data set 6 it is (strangely) 0.20 bits/note lower.

5.2.4 BOTH: Comparison of Three Combination Methods

Pearce (2005) proposes three ways of combining long- and short-term models. The first, LS1, consists in combining viewpoint distributions within each of the LTM and STM first, using one bias, and then combining the two resulting distributions using another. This has been the method of choice in research to date. The second, LS2, effects a pairwise combination of the distributions of identical viewpoints in the LTM and STM first, using one bias, and then combines the resulting distributions using another. The third, LS3, combines all viewpoint distributions at once, irrespective of whether they are in the LTM or STM, using a single bias. To keep things relatively simple, only geometric combination is used in this comparison. This is a reasonable thing to do, bearing in mind that this form of combination has been shown to be better for the LTM, and only slightly inferior for the STM.

The investigation began by using 50-hymn corpus 'A', BOTH, weighted geometric combination, an $\hbar$ of 2, a bias of 1.4 (the latter two values being optimal for the LTM alone) and, separately, LTM-STM combination methods LS1 and LS2 for comparison. The graph in Figure 5.13 shows an initial fall in cross-entropy with increasing viewpoint selection L-S bias values, after which there is no further significant change in cross-entropy. Combination method LS1 consistently produces very slightly lower cross-entropies. The lowest cross-entropy multiple viewpoint system is chosen using an L-S bias of 9 during viewpoint selection. On optimising the biases afterwards, a bias of 0.8 and an L-S bias of 27.4 give a cross-entropy of 3.12 bits/note. This is 0.09 bits/note

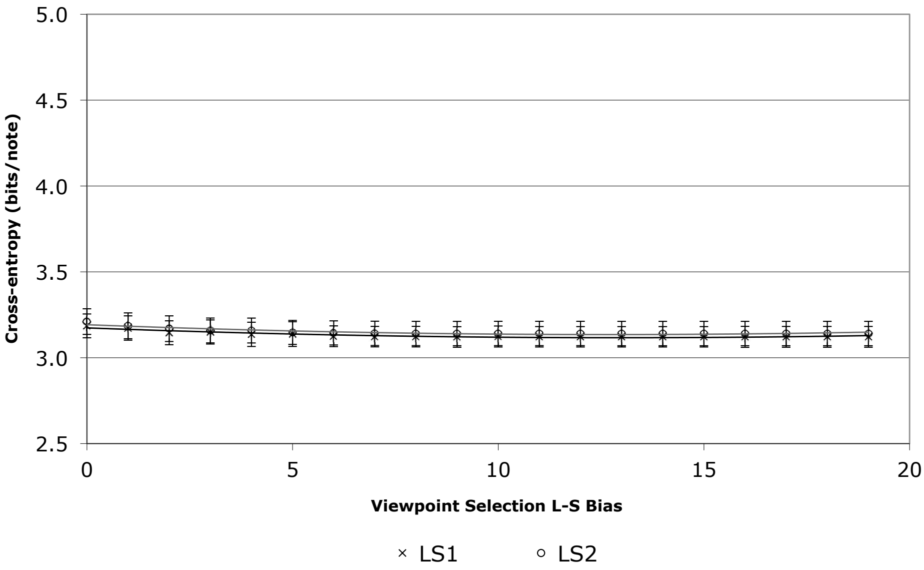


Figure 5.13: Plot of L-S bias used during viewpoint selection against cross-entropy for BOTH using corpus 'A', weighted geometric viewpoint combination, an $\hbar$ of 2 and a bias of 1.4 for viewpoint selection. LTM-STM combination methods LS1 and LS2 are compared.

lower than the LTM, but 0.7 bits/note higher than the LTM+. At this stage, the lowest cross-entropy LS2 model is chosen using an L-S bias of 16 during viewpoint selection.

In the next set of viewpoint selection runs, the bias was fixed at 1.4, L-S bias fixed at 9 and 16 respectively for LS1 and LS2, and $\hbar$ varied from 0 to 5. It can be seen from Figure 5.14 that an $\hbar$ of 3 has a very similar performance to an $\hbar$ of 2 for LS1; and for LS2, an $\hbar$ of 3 is actually slightly better, making it worthy of further investigation. Viewpoint selection runs identical to those depicted in Figure 5.13, except for the use of an $\hbar$ of 3, were therefore carried out. The results (not plotted) show that an $\hbar$ of 2 remains optimal for LS1, whereas an $\hbar$ of 3 gives a very slightly better performance for LS2. After optimisation, a bias of 1 and an L-S bias of 28.7 give a cross-entropy of 3.14 bits/note for LS2, which means that LS1 is still slightly better.

Having noticed that the cross-entropy becomes more or less constant at quite low viewpoint selection L-S bias values, some runs were carried out using L-S biases of 50 and 100 to indicate whether or not this trend continued. The results for LS1 and LS2, using an $\hbar$ of both 2 and 3, suggest that the trend does indeed continue. In all cases, the cross-entropies are very slightly higher than the lowest and, in general, the optimised biases are similar to those seen previously. There is one notable exception, however, which is the LS1 run using an $\hbar$ of 2 and L-S bias for viewpoint selection of 100. Whereas for other viewpoint selection L-S biases (all else being equal) the optimised L-S bias is in the range 11.4 to 27.6, the exceptional run has a value of 111.7.

The LTM-STM combination method was then switched to LS3. Viewpoint selection runs were carried out using 50-hymn corpus 'A', BOTH, weighted geometric combination

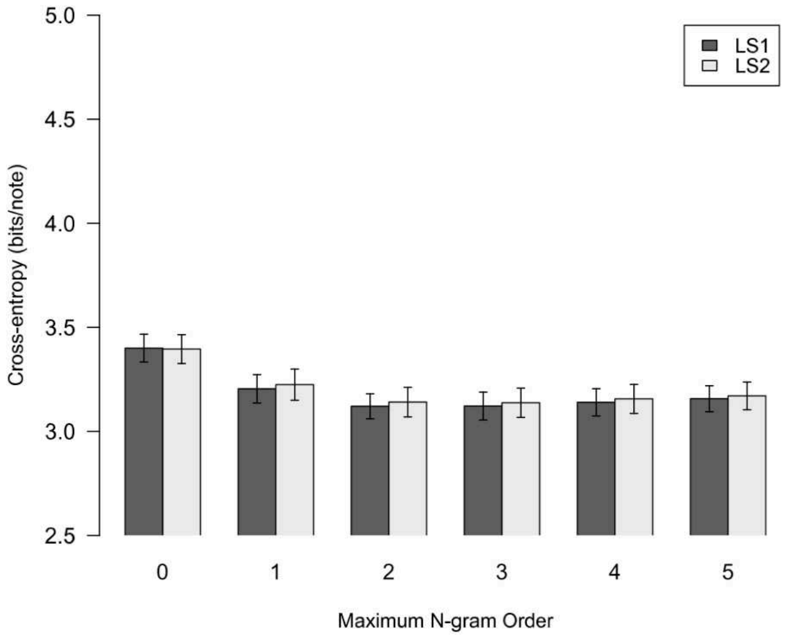


Figure 5.14: Bar chart showing how cross-entropy varies with $\hbar$ for BOTH using corpus 'A', a viewpoint selection bias of 1.4 and weighted geometric combination. LTM-STM combination methods LS1 and LS2 (viewpoint selection L-S bias of 9 and 16 respectively) are compared.

and an $\hbar$ of 2, with bias varied. The results in Figure 5.15 show that the lowest cross-entropy multiple viewpoint system is chosen using a bias of 4 during viewpoint selection (Figure 5.16 confirms that an $\hbar$ of 2 is optimal, if only just). After optimisation, a bias of 3.8 gives rise to a cross-entropy of 3.21 bits/note. This is the worst performing of the LTM-STM combination methods; indeed, its performance is only about the same as the LTM alone. The best performing LTM-STM combination method is therefore LS1.

At this point, let us examine how cross-entropy varies with L-S bias, given a multiple viewpoint system. The data in Figure 5.17 is taken from the bias optimisation run for the best BOTH system, with additional data points at low L-S bias values to give a better indication of the shape of the graph (other data points were removed for the sake of clarity). Although an L-S bias of 27.4 is optimal, there is clearly very little change in cross-entropy over a very wide range of values.

5.2.5 BOTH+ vs. BOTH

This part of the investigation began by using 50-hymn corpus 'A', BOTH+, weighted geometric combination, LTM-STM combination method LS1, an $\hbar$ of 4 and a bias of 2 (the latter two values are optimal for LTM+ alone). The graph in Figure 5.18 compares the results of these viewpoint selection runs with the set of runs producing the best performing BOTH. The graphs have a similar shape; but the updated model consistently

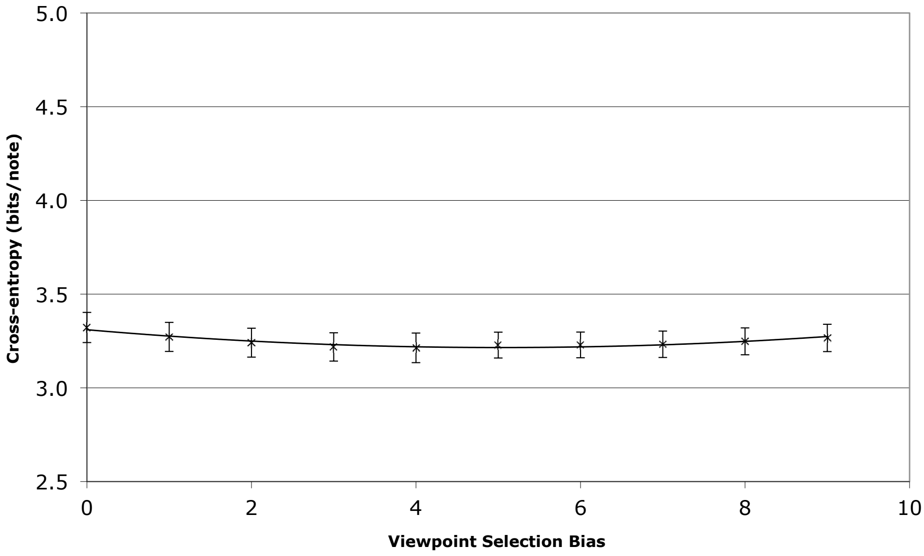


Figure 5.15: Plot of bias used during viewpoint selection against cross-entropy for BOTH using corpus ‘A’, weighted geometric viewpoint combination, an $\hbar$ of 2 and LTM-STM combination method LS3.

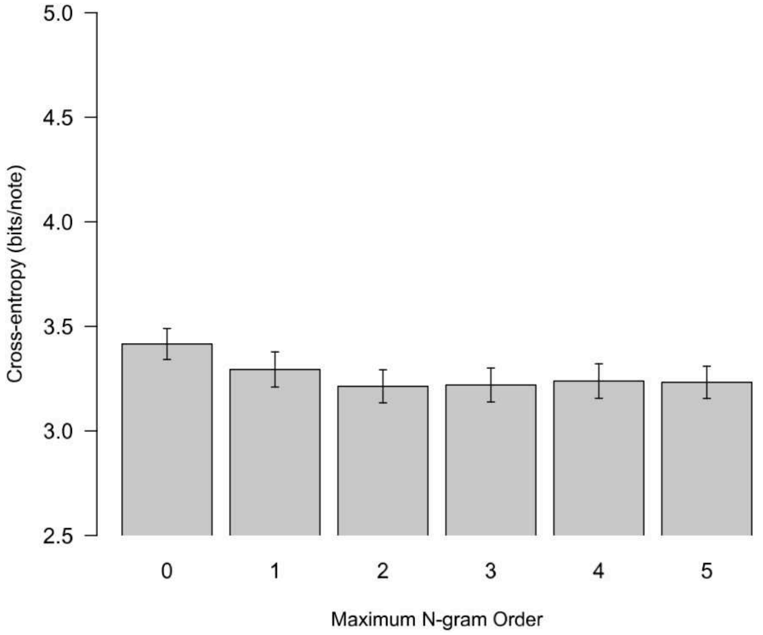


Figure 5.16: Bar chart showing how cross-entropy varies with $\hbar$ for BOTH using corpus ‘A’, weighted geometric combination, a bias of 4 and LTM-STM combination method LS3.

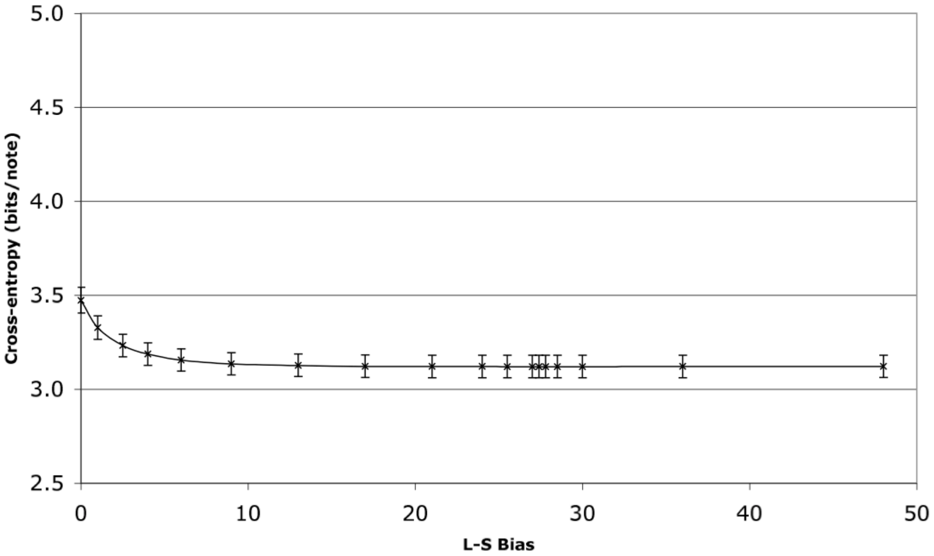


Figure 5.17: Plot of L-S bias against cross-entropy for the best performing BOTH using corpus ‘A’.

produces lower cross-entropies. The lowest cross-entropy multiple viewpoint system is chosen using an L-S bias of 14 during viewpoint selection.

In the next set of viewpoint selection runs, the bias and L-S bias were fixed at 2 and 14 respectively, and $\hbar$ varied from 0 to 5. Figure 5.19 compares this set of runs with a similar set of runs without update, using a bias and L-S bias of 1.4 and 9 respectively. The chart shows that BOTH+ has the better performance across the board, and that an $\hbar$ of 3 has the lowest cross-entropy, which is 3.02 bits/note (the optimised bias and L-S bias are 1.5 and 19.4 respectively). This is 0.10 bits/note lower than BOTH, and 0.03 bits/note lower than LTM+ alone. Pearce (2005) also finds that BOTH+ outperforms LTM+ (again, using the single viewpoint system {Pitch}).

Once again, some viewpoint selection runs were carried out using L-S biases of 50 and 100 ($\hbar = 4$, bias $= 2$). In this case, both of the resulting cross-entropies are very slightly lower than the lowest previously found, with a viewpoint selection L-S bias of 100 being optimal. This is different from the model without update findings; however, the change in cross-entropy is so minuscule that further investigation is not warranted.

The best multiple viewpoint systems for BOTH+ and BOTH have 25 viewpoints in common (total number of viewpoints: 32 and 27 respectively); that is, very similar systems were selected in each case (see Table 5.1). The similarity extends to LTM+ and LTM: there are 19 viewpoints which are common to all four systems, 7 viewpoints common to three systems and 7 viewpoints common to two systems (out of a total of 45 viewpoints selected).

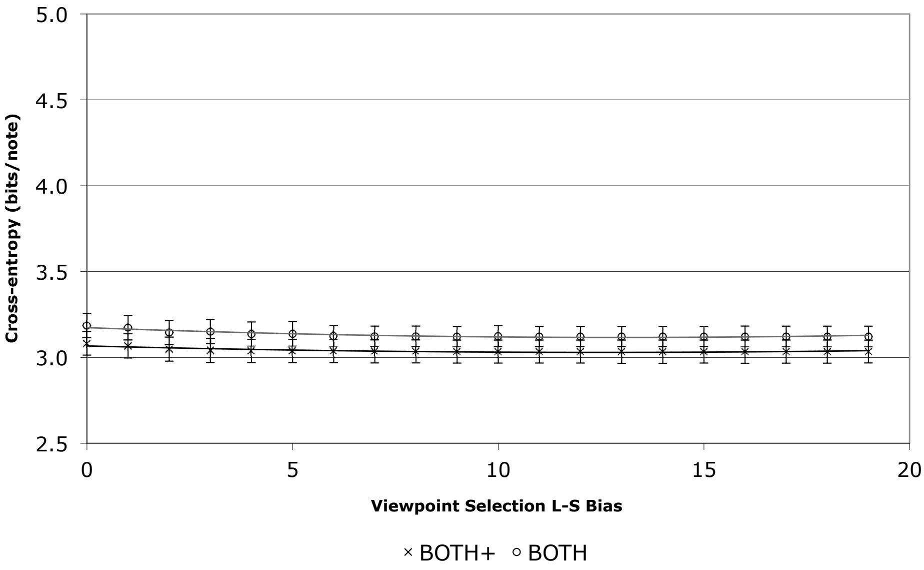


Figure 5.18: Plot of L-S bias used during viewpoint selection against cross-entropy for BOTH+ and BOTH using corpus ‘A’, weighted geometric viewpoint combination and LTM-STM combination method LS1. BOTH+ uses an $\hbar$ of 4 and a bias of 2, while BOTH uses an $\hbar$ of 2 and a bias of 1.4.

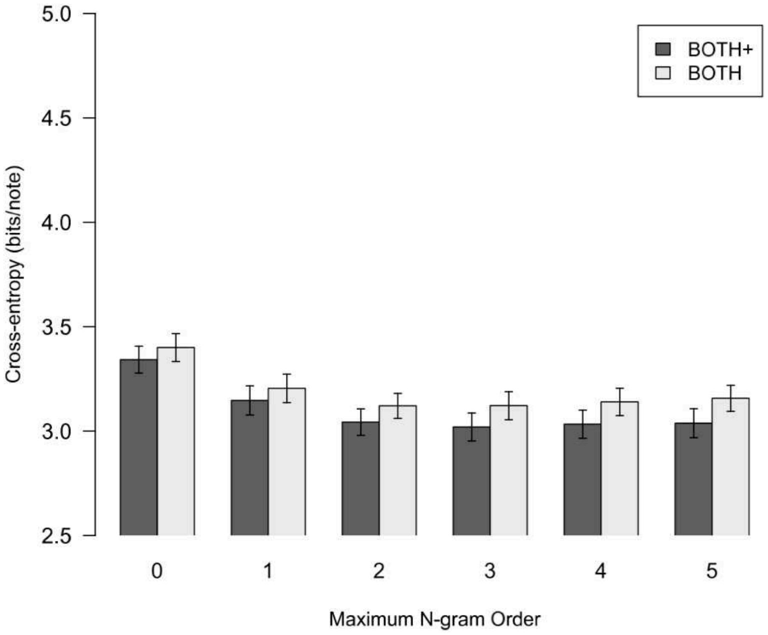


Figure 5.19: Bar chart showing how cross-entropy varies with $\hbar$ for BOTH+ and BOTH using corpus ‘A’, weighted geometric viewpoint combination and LTM-STM combination method LS1. BOTH+ uses a bias of 2 and an L-S bias of 14, while BOTH uses a bias of 1.4 and an L-S bias of 9.

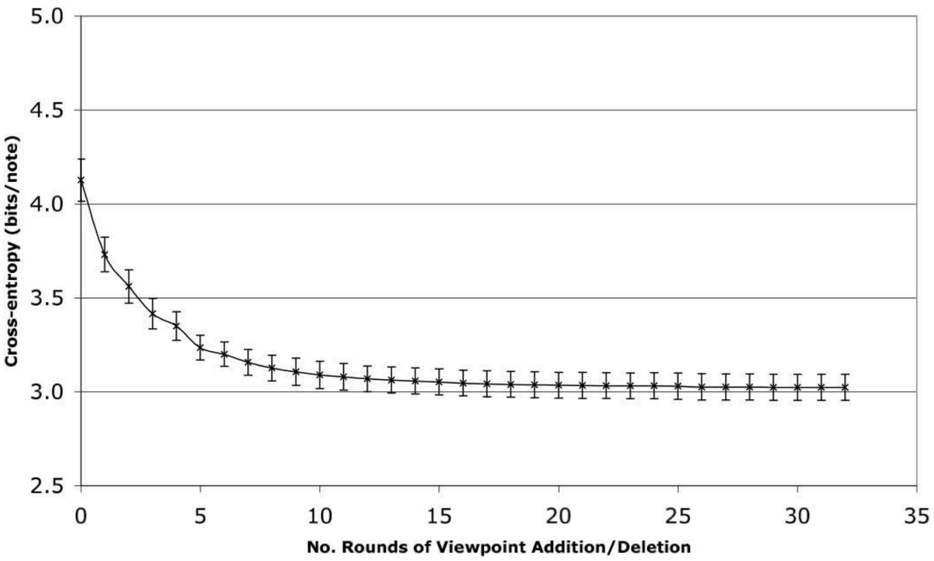


Figure 5.20: Plot of number of rounds of viewpoint addition/deletion against cross-entropy for the best performing BOTH+ using corpus 'A'.

5.2.6 Curtailing Viewpoint Selection

Viewpoint selection is a very time consuming process, as discussed in §3.4.5; therefore finding ways to reduce the time required to carry out viewpoint selection is always useful. What happens in a typical viewpoint selection run is that there is a large reduction in cross-entropy associated with the first viewpoint added to the initial multiple viewpoint system and, in general, smaller reductions with each successive added viewpoint. Eventually, there is a long tail of viewpoints which contribute comparatively little to the overall cross-entropy reduction, as illustrated in Figure 5.20. This plot of number of rounds of viewpoint addition/deletion against cross-entropy for the best performing BOTH+ clearly shows such a tail.

Before coming up with a way to shorten the tail, let us examine this viewpoint selection run in more detail. Table 5.3 shows cross-entropies for each of the ten corpus/data set combinations used during ten-fold cross-validation, followed by the mean cross-entropy. Looking at the first column of figures, we find that the cross-entropy falls until viewpoint `IntFirstInPhrase` ⊗ `ScaleDegree` is added (the increase is highlighted by the colour red). This is a mere hiccough, however, as the trend remains downward (albeit slowly) until the lowest cross-entropy is reached on the addition of (`ScaleDegree` ⊖ `FirstInPhrase`) ⊗ `SeqPositionInBar` (highlighted by the colour blue). By the end of viewpoint selection, the cross-entropy has risen very slightly. In columns 2 and 8, the lowest cross-entropy occurs at the end of viewpoint selection; while in columns 3 and 9 it appears well before the end of selection, with a noticeable increase in cross-entropy occurring thereafter.

There seems to be no obvious principled way of determining when to halt the view-

Multiple Viewpoint System	1	2	3	4	5	6	7	8	9	10	$\bar{H}$
{Duration, Pitch}	3.511	4.545	3.919	4.330	4.379	4.431	3.829	4.366	4.354	3.594	4.126
+ ScaleDegree ⊗ Phrase	3.183	4.090	3.487	3.862	3.926	3.858	3.561	3.908	4.044	3.384	3.730
+ Interval ⊗ FirstInPhrase	3.009	3.870	3.370	3.734	3.719	3.653	3.393	3.756	3.883	3.217	3.560
+ Duration ⊗ Metre	2.908	3.701	3.214	3.561	3.562	3.507	3.289	3.634	3.676	3.101	3.415
+ ScaleDegree ⊗ Metre	2.867	3.659	3.197	3.462	3.440	3.471	3.206	3.553	3.570	3.067	3.349
+ DurRatio ⊗ Phrase	2.860	3.535	3.167	3.397	3.385	3.434	3.209	3.435	3.479	2.960	3.286
− Duration	2.820	3.449	3.126	3.304	3.324	3.396	3.197	3.360	3.452	2.915	3.234
+ IntFirstInBar ⊗ ScaleDegree	2.813	3.425	3.094	3.268	3.274	3.390	3.116	3.363	3.405	2.857	3.200
− Pitch	2.807	3.414	3.129	3.271	3.262	3.388	3.102	3.367	3.380	2.877	3.200
+ ScaleDegree ⊗ Tessitura	2.745	3.395	3.065	3.170	3.260	3.336	3.073	3.364	3.334	2.826	3.157
+ Interval ⊗ TactusPositionInBar	2.668	3.357	3.063	3.166	3.208	3.293	3.039	3.341	3.295	2.832	3.126
+ Duration ⊗ LastInPhrase	2.656	3.428	2.999	3.138	3.177	3.296	3.030	3.338	3.204	2.798	3.106
+ (ScaleDegree ⊖ FirstInPhrase) ⊗ FirstInPiece	2.635	3.415	2.981	3.113	3.165	3.270	3.023	3.317	3.188	2.789	3.090
+ DurRatio ⊗ TactusPositionInBar	2.624	3.343	2.992	3.134	3.135	3.270	3.010	3.312	3.200	2.771	3.079
+ IntFirstInPhrase ⊗ ScaleDegree	2.636	3.342	2.952	3.124	3.127	3.271	3.009	3.279	3.178	2.770	3.069
+ Interval ⊗ LastInPhrase	2.624	3.321	2.942	3.143	3.130	3.258	2.971	3.275	3.196	2.773	3.063
+ Duration ⊗ (ScaleDegree ⊖ LastInPhrase)	2.616	3.331	2.934	3.128	3.125	3.258	2.968	3.267	3.173	2.768	3.057
+ (ScaleDegree ⊖ FirstInPhrase) ⊗ Piece	2.609	3.329	2.929	3.117	3.121	3.248	2.969	3.260	3.167	2.767	3.052
+ Pitch ⊗ ScaleDegree	2.617	3.311	2.929	3.091	3.143	3.238	2.955	3.267	3.161	2.751	3.046
+ Interval ⊗ (ScaleDegree ⊖ LastInPhrase)	2.609	3.307	2.917	3.087	3.152	3.233	2.946	3.264	3.160	2.752	3.043
+ (ScaleDegree ⊖ FirstInPhrase) ⊗ Tessitura	2.606	3.302	2.919	3.077	3.152	3.227	2.944	3.259	3.154	2.755	3.040
+ Interval ⊗ (ScaleDegree ⊖ FirstInBar)	2.599	3.306	2.913	3.076	3.146	3.239	2.947	3.254	3.136	2.753	3.037
+ Duration ⊗ PositionInBar	2.593	3.296	2.935	3.071	3.148	3.220	2.936	3.253	3.146	2.755	3.035
+ IntFirstInPiece ⊗ (ScaleDegree ⊖ FirstInPhrase)	2.585	3.293	2.938	3.071	3.147	3.216	2.935	3.250	3.147	2.756	3.034
+ IntFirstInBar ⊗ (ScaleDegree ⊖ FirstInPhrase)	2.582	3.293	2.940	3.069	3.147	3.214	2.933	3.246	3.144	2.756	3.033
+ (Pitch ⊖ Tactus) ⊗ (ScaleDegree ⊖ FirstInPhrase)	2.583	3.292	2.940	3.067	3.146	3.210	2.933	3.243	3.149	2.756	3.032
+ Interval ⊗ FirstInBar	2.571	3.283	2.940	3.086	3.137	3.212	2.939	3.244	3.154	2.747	3.031
+ Contour ⊗ ScaleDegree	2.573	3.293	2.930	3.074	3.131	3.219	2.946	3.233	3.160	2.738	3.030
+ Pitch ⊗ TactusPositionInBar	2.567	3.288	2.940	3.078	3.131	3.232	2.932	3.209	3.158	2.731	3.026
+ (ScaleDegree ⊖ FirstInPhrase) ⊗ SeqPositionInBar	2.567	3.287	2.939	3.074	3.122	3.232	2.934	3.209	3.158	2.731	3.025
+ Duration ⊗ (ScaleDegree ⊖ FirstInPhrase)	2.569	3.287	2.943	3.072	3.108	3.230	2.939	3.210	3.157	2.733	3.025
+ Interval ⊗ (ScaleDegree ⊖ FirstInPhrase)	2.568	3.284	2.939	3.073	3.108	3.230	2.940	3.209	3.158	2.733	3.024
+ (ScaleDegree ⊖ FirstInPhrase) ⊗ FirstInBar	2.568	3.285	2.940	3.071	3.107	3.228	2.942	3.208	3.158	2.733	3.024
+ (Interval ⊗ ScaleDegree) ⊖ FirstInPhrase	2.569	3.284	2.942	3.066	3.107	3.228	2.942	3.208	3.159	2.734	3.024
+ (Interval ⊖ Tactus) ⊗ (ScaleDegree ⊖ FirstInPhrase)	2.569	3.282	2.941	3.068	3.107	3.228	2.943	3.208	3.160	2.734	3.024

Table 5.3: Cross-entropy at each stage of addition or deletion during viewpoint selection of the best version 0 multiple viewpoint system (predicting Duration and Pitch) for BOTH+ using corpus 'A'. Cross-entropies are shown for each of the ten corpus/data set combinations used during ten-fold cross-validation, as well as the mean cross-entropy *($\bar{H}$)*. Red indicates an increase, while blue marks the lowest cross-entropy (changes do not always register to three decimal places).

point selection process; therefore an arbitrary method is proposed. In the typical case above, if viewpoint selection were halted as soon as a viewpoint addition resulted in a cross-entropy reduction of < 0.005 bits/note, then the number of viewpoints in the system would be nearly halved at the expense of < 0.02 bits/note. Similarly, if the halting criterion were < 0.0015 bits/note, then the number of viewpoints would be reduced by just over a third at the expense of < 0.01 bits/note. The latter halting criterion will be used for all viewpoint selection runs below except where otherwise stated.

5.2.7 Comparison of Different Corpora

To investigate differences between corpora, viewpoint selection was carried out for BOTH+ using weighted geometric combination, a bias of 2, an L-S bias of 14 and, separately, 50-hymn (2,023-note) corpus 'A', 50-hymn (2,284-note) corpus 'B' and 100-hymn (4,307-note) corpus 'A+B'. As we can see from Figure 5.21, corpus 'B' produces a lower cross-entropy than corpus 'A' across the board, in spite of the fact that a full (not curtailed) viewpoint selection was carried out using corpus 'A'. In both cases an $\hbar$ of 3 is optimal, giving a cross-entropy of 2.90 bits/note for corpus 'B' compared with 3.02 bits/note for 'A'. Bearing in mind the huge variation in cross-entropy within a single corpus (see Table 5.3), this is not such a large difference; but it may indicate differences between the corpora (in spite of them both originating from Vaughan Williams 1933) which make corpus 'B' easier to learn. Table 5.4 provides some evidence for this: although there is some overlap between the multiple viewpoint systems selected for 'A' and 'B', there are a good many viewpoints unique to one or other of the systems. To be precise, there are 21 viewpoints in the system for 'A' (curtailed for purposes of comparison) and 27 for 'B'; of these, 12 are common to both systems.

Corpus 'A+B' produces very similar results to corpus 'B', even though it is twice the size. It is, in fact, very slightly better than 'B', with an optimal $\hbar$ of 6 giving a cross-entropy of 2.90 bits/note. On the face of it, this would seem to suggest that the addition of 'A' to 'B' provides very little extra information; but the reality is more complex, as will be seen in §5.3. Notice that beyond an $\hbar$ of 3, there is virtually no change in cross-entropy. This is often the case beyond a certain point, and is likely to be due to larger contexts rarely being found in the model during prediction, resulting in back-off to some lower order before a match is found. Notice also that the 'A+B' error bars are generally the smallest of the groups of three. This suggests that corpora 'A' and 'B' are stylistically similar enough to combine well, forming a distribution with a smaller standard error under ten-fold cross-validation.

Table 5.4 shows a considerable overlap between the systems selected for 'B' and 'A+B', suggesting that the 'B' system may be more generally applicable than the 'A' system. In order to test the validity of this hypothesis, the system which learned from corpus 'B' was run using corpus 'A', and vice versa. In each case, after optimisation of the biases, the cross-validated cross-entropy of the corpus was 0.05 bits/note higher

Viewpoint	A	B	A+B
ScaleDegree $\otimes$ Phrase	×	×	
Interval $\otimes$ FirstInPhrase	×		
Duration $\otimes$ Metre	×	×	
ScaleDegree $\otimes$ Metre	×	×	
DurRatio $\otimes$ Phrase	×	×	×
IntFirstInBar $\otimes$ ScaleDegree	×		×
ScaleDegree $\otimes$ Tessitura	×	×	×
Interval $\otimes$ TactusPositionInBar	×		×
Duration $\otimes$ LastInPhrase	×		
(ScaleDegree $\ominus$ FirstInPhrase) $\otimes$ FirstInPiece	×	×	×
DurRatio $\otimes$ TactusPositionInBar	×		
IntFirstInPhrase $\otimes$ ScaleDegree	×	×	×
Interval $\otimes$ LastInPhrase	×		
Duration $\otimes$ (ScaleDegree $\ominus$ LastInPhrase)	×	×	×
(ScaleDegree $\ominus$ FirstInPhrase) $\otimes$ Piece	×	×	×
Pitch $\otimes$ ScaleDegree	×		
Interval $\otimes$ (ScaleDegree $\ominus$ LastInPhrase)	×	×	×
(ScaleDegree $\ominus$ FirstInPhrase) $\otimes$ Tessitura	×	×	×
Interval $\otimes$ (ScaleDegree $\ominus$ FirstInBar)	×	×	×
Duration $\otimes$ PositionInBar	×		
IntFirstInPiece $\otimes$ (ScaleDegree $\ominus$ FirstInPhrase)	×		
Contour $\otimes$ ScaleDegree		×	
Duration $\otimes$ Phrase		×	×
Duration $\otimes$ (ScaleDegree $\ominus$ FirstInPhrase)		×	×
Interval $\otimes$ Phrase		×	×
(Pitch $\ominus$ Tactus) $\otimes$ ScaleDegree		×	×
Interval $\otimes$ Metre		×	
Pitch $\otimes$ (ScaleDegree $\ominus$ FirstInPhrase)		×	×
Duration $\otimes$ TactusPositionInBar		×	×
Interval $\otimes$ (ScaleDegree $\ominus$ Tactus)		×	×
Pitch $\otimes$ FirstInPhrase		×	
DurRatio $\otimes$ Metre		×	×
Interval $\otimes$ IOI		×	
Duration $\otimes$ (Interval $\ominus$ FirstInPhrase)		×	
(Pitch $\ominus$ Tactus) $\otimes$ (ScaleDegree $\ominus$ FirstInPhrase)		×	×
Pitch $\otimes$ (Interval $\ominus$ FirstInPhrase)		×	
ScaleDegree $\otimes$ TactusPositionInBar			×
Pitch $\otimes$ Phrase			×
Interval $\otimes$ LastInPiece			×
Pitch $\otimes$ Metre			×

Table 5.4: Best version 0 multiple viewpoint systems (predicting Duration and Pitch) for BOTH+ comparing corpus ‘A’ *(A)*, corpus ‘B’ *(B)* and corpus ‘A+B’ *(A+B)*.

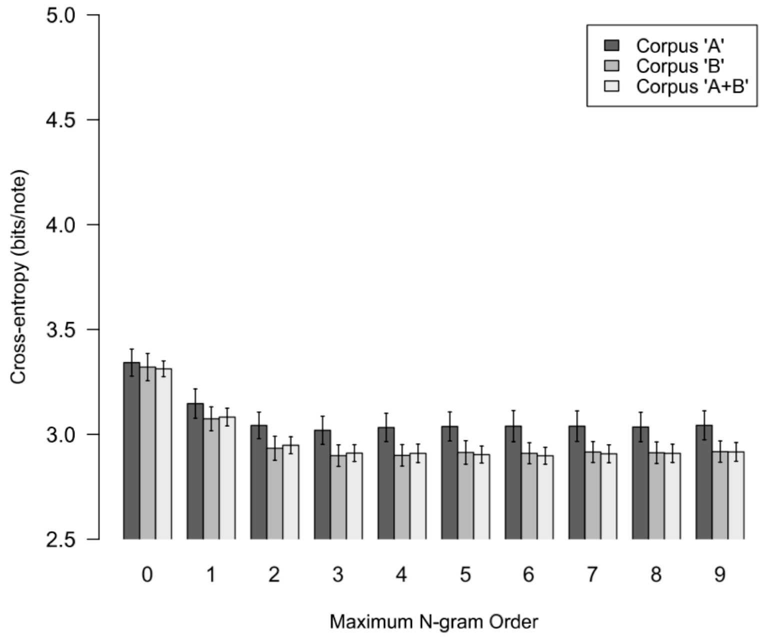


Figure 5.21: Bar chart showing how cross-entropy varies with $\hbar$ for BOTH+ using weighted geometric viewpoint combination, a bias of 2 and an L-S bias of 14 for viewpoint selection. Corpora 'A', 'B' and 'A+B' are compared.

than that of the corpus on which the system was trained, which indicates that neither system has a strong claim to be more generally applicable.

In order to gauge the effect of using a large corpus with a system trained on a smaller corpus, corpus 'A+B' was firstly used in conjunction with the best system trained on corpus 'A'. The optimised bias and L-S bias were 1.4 and 12.9 respectively, giving a cross-entropy of 2.93 bits/note, which was considerably lower than the original 3.02 bits/note when using corpus 'A'. On the other hand, when the large corpus was tried with the best system trained on corpus 'B', the resulting cross-entropy was slightly higher than the original: 2.93 compared with 2.90 bits/note, with an optimised bias and L-S bias of 1.4 and 14.1 respectively. In each case, however, the cross-entropy was only 0.03 bits/note higher than that attained by the system trained on corpus 'A+B'. This suggests that learning from a relatively small corpus (to keep the viewpoint selection process within an acceptable timeframe) and then using a large corpus for prediction is a reasonable compromise.

5.3 Prediction of `Duration` and `Pitch` Separately

We now investigate the use of separately selected and optimised multiple viewpoint systems for the prediction of `Duration` and `Pitch`, rather than employing a single system for the prediction of both attributes, as has been the case so far. BOTH+ will be employed for this investigation, since it has proven to perform best in the prediction of `Duration` and `Pitch` together. Please note that all charts and graphs in this section relating to the separate prediction of `Duration` or `Pitch` have a cross-entropy range from 0.0 to 2.5 bits/note; and all charts and graphs relating to their combined prediction have a cross-entropy range from 2.5 to 5.0 bits/note, unless otherwise stated.

5.3.1 Prediction of `Duration` Only

Viewpoint selection runs were carried out for the prediction of `Duration` alone using BOTH+, weighted geometric combination, LTM-STM combination method LS1, a bias of 2 and an L-S bias of 14, with $\hbar$ varied (the parameters used in the final set of BOTH+ runs predicting `Duration` and `Pitch` together). Corpora 'A', 'B' and 'A+B' are compared. The results, in Figure 5.22, show that for corpus 'A' an $\hbar$ of 3 is optimal (the same as for the prediction of `Duration` and `Pitch` together), producing a cross-entropy of 0.89 bits/note. The fact that the cross-entropy is so much lower than we have seen before (2.90 bits/note was the previous lowest) indicates that `Duration` is more predictable than `Pitch`, which, for this corpus of hymn tunes, is what we would expect. What is particularly interesting, however, is the much lower optimised L-S bias value for the prediction of `Duration` alone: 1.6 compared with 19.4 for the best corpus 'A' system for predicting `Duration` and `Pitch`. Since larger biases increase the relative weight given to distributions deemed to predict with more certainty, and such distributions are usually expected to be found in the LTM (by virtue of the larger number of statistics), a lower L-S bias value effectively gives increased weight to the STM. This makes sense since the rhythmic patterns of an individual hymn tune can be quickly learned, and are particularly relevant to the prediction of that melody. The effect is minuscule, however, as can be seen in Figure 5.23, which shows how cross-entropy varies with L-S bias given the multiple viewpoint system under discussion here. The shape of the graph is clearly different from that in Figure 5.17. For completeness, the optimised bias is 1.8, which is well within the usual range.

Interestingly (returning to Figure 5.22), the best performing corpus 'B' system has a higher cross-entropy than the best corpus 'A' system (0.92 *cf.* 0.89 bits/note). The much lower cross-entropy for the system predicting both `Duration` and `Pitch` using corpus 'B' rather than 'A' (see §5.2.7 above) must therefore be entirely due to `Pitch` being more predictable in corpus 'B'. As with corpus 'A', an $\hbar$ of 3 is optimal; and the optimised L-S bias, at 3.4, is again very low. The optimised bias is 1.2.

Finally, corpus 'A+B' is found to have the lowest cross-entropy of all (0.86 bits/note); but unusually, this occurs at an $\hbar$ of 6. In this case, the optimised L-S bias is 2.6 and the optimised bias 2.2.

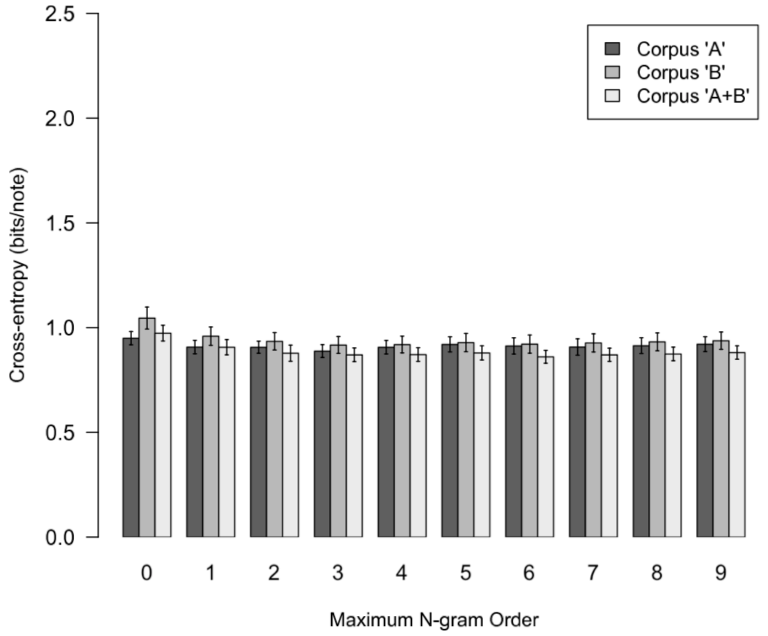


Figure 5.22: Bar chart showing how cross-entropy varies with $\hbar$ for BOTH+ using weighted geometric viewpoint combination, a bias of 2 and an L-S bias of 14 (for viewpoint selection) for the prediction of `Duration` only. Corpora 'A', 'B' and 'A+B' are compared.

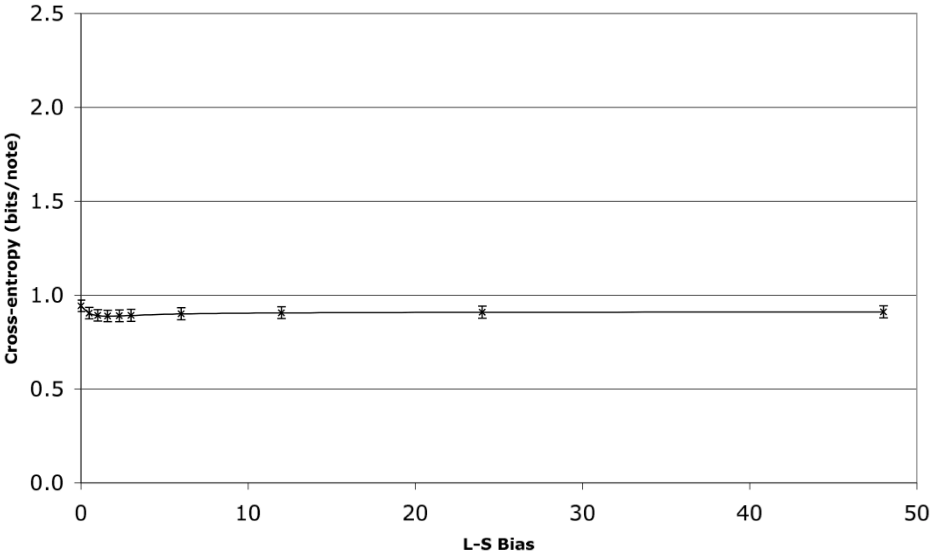


Figure 5.23: Plot of L-S bias against cross-entropy for the best performing BOTH+ using corpus 'A' for the prediction of `Duration` only.

5.3.2 Prediction of Pitch Only

Viewpoint selection runs were then carried out for the prediction of Pitch alone using the same run-time parameters as for the prediction of Duration above. Again, corpora 'A', 'B' and 'A+B' are compared. The resulting bar chart of $\hbar$ against cross-entropy (see Figure 5.24) is unusual compared with other such charts that we have previously seen, in that the cross-entropies in the $\hbar$ range 2 to 5 inclusive are very similar to each other. It is for this reason that further values of $\hbar$ were considered for this chart (and for Figure 5.22) to find out if the trend continues, which it certainly does for Pitch prediction.

For corpus 'A', an unusually high $\hbar$ of 7 is optimal, producing a cross-entropy of 2.11 bits/note. Both the optimised bias and the optimised L-S bias are within the usual range (1.5 and 25.6 respectively). In contrast, the best performing system using corpus 'B' has a much lower cross-entropy, 1.95 bits/note, occurring at an $\hbar$ of 3. The optimised bias and L-S bias are similar to those of the corpus 'A' system (2.0 and 23.8 respectively). The most surprising thing about this bar chart is that the corpus 'A+B' system performs less well than the corpus 'B' system across the board. This is accounted for by the large difference in cross-entropy between corpora 'A' and 'B'; since the 'A+B' cross-entropy is lower than the mean of the other two, we can be assured that the 'A+B' model is, in general, the best of the three. In this case, an $\hbar$ of 3 is optimal (only just better than an $\hbar$ of 4 to 9, however), resulting in a cross-entropy of 2.00 bits/note. The optimised bias and L-S bias are 1.9 and 30.8 respectively.

The best system found by Pearce (2005) for the prediction of Pitch has a cross-entropy of 1.91 bits/note, which is 0.09 bits/note lower than the 'A+B' system above. Bearing in mind that the former utilised a corpus of approximately twice the size (albeit that it included melodies in major and minor keys), and that Pearce (2005) found interpolated smoothing and unbounded order PPM* models (not implemented in this research) to be beneficial, it is highly likely that 1.91 bits/note (for the larger corpus) will be bettered in future work.

In contrast to Figure 5.21, corpus 'B' has by far the smallest error bars here. There are generally large gaps between the 'A' and 'B' error bars, indicating quite distinct distributions. It is perhaps then unsurprising that the combination of corpora 'A' and 'B' has produced a distribution with a relatively large standard error. The fact that Duration is also predicted in the former case seems to have a balancing effect.

5.3.3 Combining Systems Predicting Duration and Pitch Separately

In order to compare the performance of models which predict Duration and Pitch separately with those predicting them together, over a range of $\hbar$, the cross-entropies of the Duration prediction and Pitch prediction models are simply added. The comparison for corpus 'A' over the $\hbar$ range 0 to 9, shown in Figure 5.25, indicates that the selection and optimisation of specialist multiple viewpoint systems for the prediction of individ-

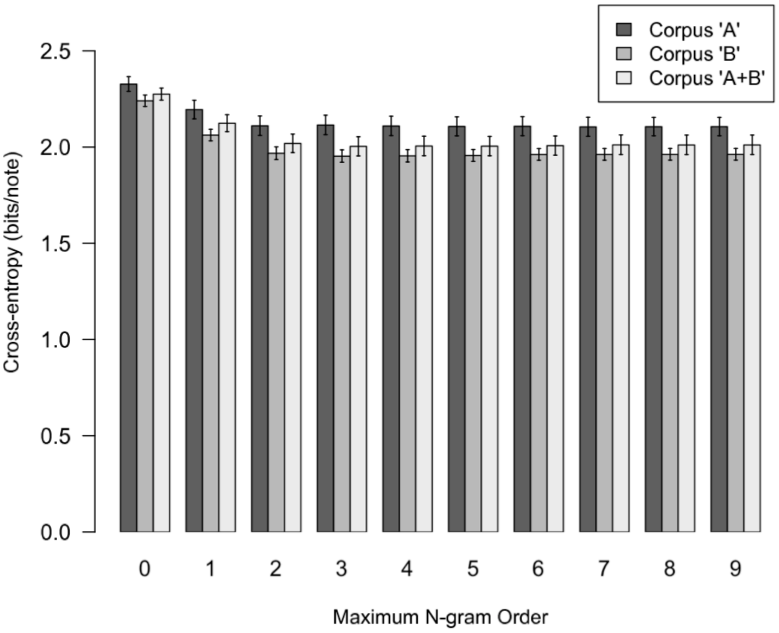


Figure 5.24: Bar chart showing how cross-entropy varies with $\hbar$ for BOTH+ using weighted geometric viewpoint combination, a bias of 2 and an L-S bias of 14 for the prediction of `Pitch` only. Corpora 'A', 'B' and 'A+B' are compared.

ual attributes produces slightly lower cross-entropies across the board, in spite of the fact that viewpoint selection for these systems was curtailed. The best system overall combines the best individual systems from §5.3.1 and §5.3.2 ($\hbar$ of 3 and 7 for `Duration` and `Pitch` respectively), resulting in a cross-entropy of 3.00 bits/note.

The comparison for corpus 'B' (in which all viewpoint selection runs were curtailed), also shows lower cross-entropies for the separate systems: see Figure 5.26. In this case the best system overall, with an $\hbar$ of 3 for both `Duration` and `Pitch`, has a cross-entropy of 2.87 bits/note. Finally, Figure 5.27 tells the same story for corpus 'A+B'. This time the best system overall, with an $\hbar$ of 6 and 3 for `Duration` and `Pitch` respectively, has a cross-entropy of 2.86 bits/note.

The way in which prediction is carried out in practice, that is, `Duration` followed by `Pitch` for each note in turn, means that we are able to split the combined system into two: all viewpoints capable of predicting `Duration` are placed in one subsystem, and all those able to predict `Pitch` are allocated to the other (some viewpoints may appear in both subsystems). In this way, we are able to directly compare separately selected systems with these subsystems.

Table 5.5 shows such a comparison for the prediction of `Duration`, using the best version 0 BOTH+ multiple viewpoint systems for corpora 'A', 'B' and 'A+B'. For each corpus, a separately selected specialist system is compared with the `Duration` predicting subset of a combined system for predicting `Duration` and `Pitch`. The two corpus 'A'

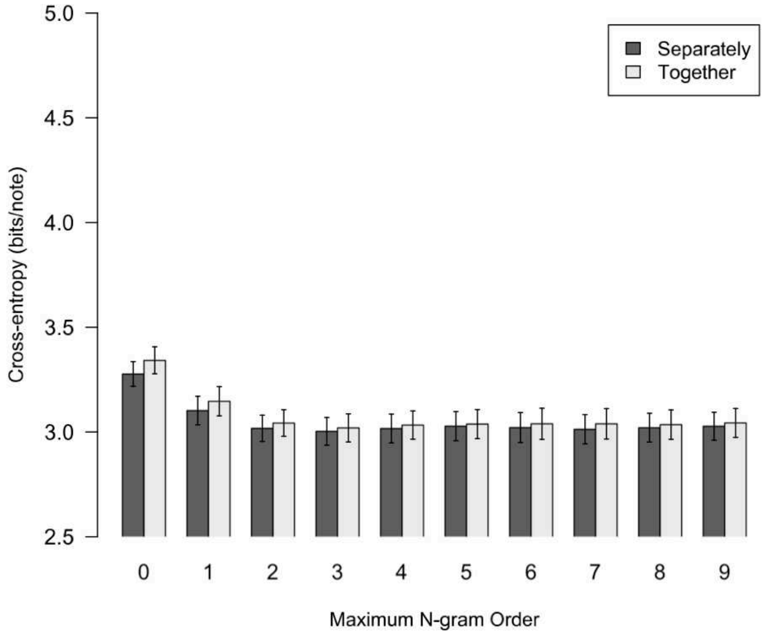


Figure 5.25: Bar chart showing how cross-entropy varies with $\hbar$ for BOTH+ using corpus 'A', weighted geometric viewpoint combination, a bias of 2 and an L-S bias of 14. The prediction of `Duration` and `Pitch` separately (*i.e.*, using separately selected multiple viewpoint systems) and together (*i.e.*, using a single multiple viewpoint system) are compared.

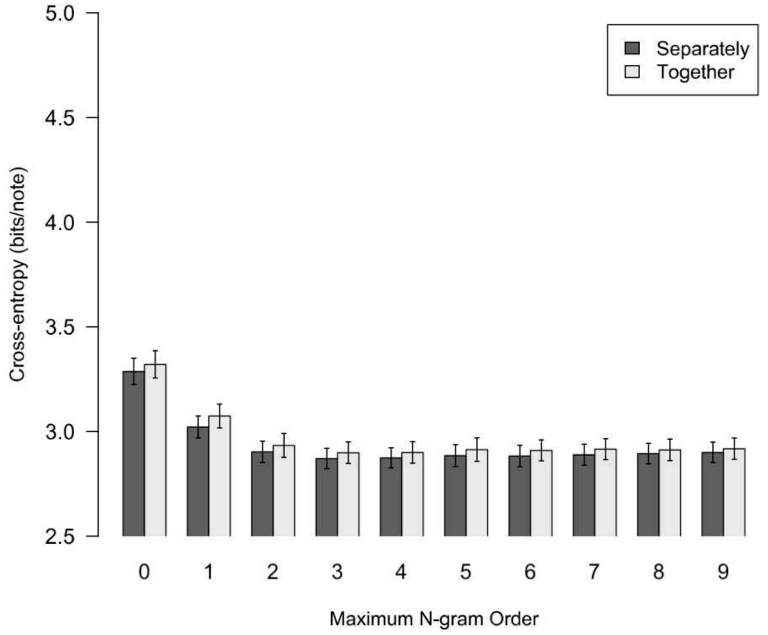


Figure 5.26: Bar chart showing how cross-entropy varies with $\hbar$ for BOTH+ using corpus 'B', weighted geometric viewpoint combination, a bias of 2 and an L-S bias of 14. The prediction of `Duration` and `Pitch` separately (*i.e.*, using separately selected multiple viewpoint systems) and together (*i.e.*, using a single multiple viewpoint system) are compared.

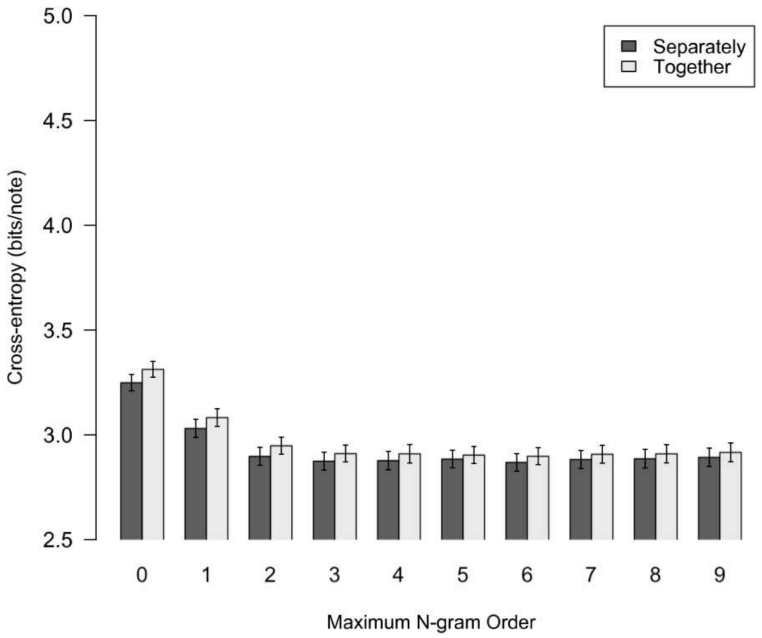


Figure 5.27: Bar chart showing how cross-entropy varies with $\hbar$ for BOTH+ using corpus 'A+B', weighted geometric viewpoint combination, a bias of 2 and an L-S bias of 14. The prediction of `Duration` and `Pitch` separately (*i.e.*, using separately selected multiple viewpoint systems) and together (*i.e.*, using a single multiple viewpoint system) are compared.

systems are almost identical: there is just one viewpoint which appears only in the separate system, namely `Duration` $\otimes$ (`ScaleDegree` $\ominus$ `FirstInPhrase`). The fact that it was not selected for the combined system suggests that it might have been at best weakly beneficial and at worst detrimental to the prediction of `Pitch`. The corpus 'B' systems are completely identical; but they are largely different from the corpus 'A' systems. The corpus 'A+B' systems are the least similar: the separate system contains four more viewpoints than the combined system subset, which again suggests that there are viewpoints beneficial to the prediction of `Duration` which are possibly detrimental to the prediction of `Pitch`.

Table 5.6 shows a similar comparison for the prediction of `Pitch`, using the best version 0 BOTH+ multiple viewpoint systems for corpora 'A', 'B' and 'A+B'. For each corpus, a separately selected specialist system is compared with the `Pitch` predicting subset of a combined system for predicting `Duration` and `Pitch`. The two corpus 'A' systems are largely the same, with some of the differences being viewpoints substituting for other similar ones: `Interval` $\otimes$ `FirstInPhrase` and `Interval` $\otimes$ `LastInPhrase` are missing from the separate system, but `Interval` $\otimes$ `Phrase` appears instead. Of particular interest is the fact that `Duration` $\otimes$ (`ScaleDegree` $\ominus$ `LastInPhrase`) is omitted from the separate system, as is the case for the other two corpora; we can deduce that this viewpoint is generally beneficial for `Duration` prediction, but at best only weakly

	A		B		A+B	
Viewpoint	S	C	S	C	S	C
`Duration` ⊗ `Metre`	×	×	×	×		
`DurRatio` ⊗ `Phrase`	×	×	×	×	×	×
`Duration` ⊗ `LastInPhrase`	×	×			×	
`DurRatio` ⊗ `TactusPositionInBar`	×	×				
`Duration` ⊗ (`ScaleDegree` ⊖ `LastInPhrase`)	×	×	×	×	×	×
`Duration` ⊗ `PositionInBar`	×	×				
`Duration` ⊗ (`ScaleDegree` ⊖ `FirstInPhrase`)	×		×	×	×	×
`Duration` ⊗ `Phrase`			×	×	×	×
`Duration` ⊗ `TactusPositionInBar`			×	×	×	×
`DurRatio` ⊗ `Metre`			×	×	×	×
`Duration` ⊗ (`Interval` ⊖ `FirstInPhrase`)			×	×	×	
`Duration` ⊗ (`ScaleDegree` ⊖ `FirstInBar`)					×	
`Duration` ⊗ `IntFirstInBar`					×	

Table 5.5: Best version 0 BOTH+ multiple viewpoint systems for corpora ‘A’, ‘B’ and ‘A+B’ (*A*, *B* and *A+B* respectively), comparing a separately selected specialist system for the prediction of `Duration` *(S)*, with the `Duration` predicting subset of a combined system for predicting `Duration` and `Pitch` *(C)*.

beneficial for `Pitch` prediction (hence its not being selected before viewpoint selection is curtailed). Its omission in the corpus ‘A’ case means that there are no viewpoints with a `Duration` or `DurRatio` component at all in the separate system.

The corpus ‘B’ systems are fairly similar to each other, but largely different from the corpus ‘A’ systems. Two viewpoints involving `Duration` are missing from the separate system; but one is retained and two involving `DurRatio` are added, showing that such viewpoints can, after all, be useful in the prediction of `Pitch`. It is quite obvious, however, that in all of the systems under consideration here the vast majority of viewpoints do not have a component derived from `Duration`. The corpus ‘A+B’ systems are similar to each other and to the corpus ‘B’ systems. Two viewpoints involving `Duration` are missing from the separate system; but two involving `DurRatio` are added.

5.4 Conclusion

In the early part of this chapter on the prediction of `Duration` and `Pitch` together, we have found that for the LTM, weighted geometric viewpoint combination is much better than weighted arithmetic (*i.e.*, it produces a much lower cross-entropy); also, LTM+ is much better than LTM. For the STM, there is hardly any difference between weighted geometric and weighted arithmetic combination: arithmetic just has the edge. The best method for combining LTM with STM is LS1, which is only slightly better than LS2, and which in turn is much better than LS3. BOTH+ is much better than BOTH; indeed, it is the best performing type of model overall. This being the case, BOTH+

	A		B		A+B	
Viewpoint	S	C	S	C	S	C
ScaleDegree ⊗ Phrase	×	×	×	×	×	
Interval ⊗ FirstInPhrase		×				
ScaleDegree ⊗ Metre	×	×	×	×		
IntFirstInBar ⊗ ScaleDegree	×	×			×	×
ScaleDegree ⊗ Tessitura	×	×		×	×	×
Interval ⊗ TactusPositionInBar	×	×			×	×
(ScaleDegree ⊖ FirstInPhrase) ⊗ FirstInPiece	×	×	×	×	×	×
IntFirstInPhrase ⊗ ScaleDegree	×	×	×	×	×	×
Interval ⊗ LastInPhrase		×			×	
Duration ⊗ (ScaleDegree ⊖ LastInPhrase)		×		×		×
(ScaleDegree ⊖ FirstInPhrase) ⊗ Piece	×	×	×	×	×	×
Pitch ⊗ ScaleDegree	×	×	×			
Interval ⊗ (ScaleDegree ⊖ LastInPhrase)	×	×	×	×	×	×
(ScaleDegree ⊖ FirstInPhrase) ⊗ Tessitura	×	×	×	×	×	×
Interval ⊗ (ScaleDegree ⊖ FirstInBar)	×	×	×	×	×	×
IntFirstInPiece ⊗ (ScaleDegree ⊖ FirstInPhrase)	×	×				
Pitch ⊗ FirstInPhrase	×			×		
Interval ⊗ Phrase	×		×	×		×
IntFirstInPiece ⊗ ScaleDegree	×					
Interval ⊗ FirstInBar	×					
Contour ⊗ ScaleDegree				×		
Duration ⊗ (ScaleDegree ⊖ FirstInPhrase)			×	×		×
(Pitch ⊖ Tactus) ⊗ ScaleDegree			×	×	×	×
Interval ⊗ Metre				×		
Pitch ⊗ (ScaleDegree ⊖ FirstInPhrase)			×	×	×	×
Interval ⊗ (ScaleDegree ⊖ Tactus)			×	×	×	×
Interval ⊗ IOI				×		
Duration ⊗ (Interval ⊖ FirstInPhrase)				×		
(Pitch ⊖ Tactus) ⊗ (ScaleDegree ⊖ FirstInPhrase)			×	×	×	×
Pitch ⊗ (Interval ⊖ FirstInPhrase)				×		
DurRatio ⊗ ScaleDegree			×			
DurRatio ⊗ Interval			×			
Interval ⊗ PositionInBar			×			
Interval ⊗ (ScaleDegree ⊖ FirstInPhrase)			×		×	
ScaleDegree ⊗ TactusPositionInBar						×
Pitch ⊗ Phrase						×
Pitch ⊗ Metre					×	×
Interval ⊗ LastInPiece					×	×
DurRatio ⊗ IntFirstInPiece					×	
DurRatio ⊗ IntFirstInPhrase					×	

Table 5.6: Best version 0 BOTH+ multiple viewpoint systems for corpora ‘A’, ‘B’ and ‘A+B’ (*A*, *B* and *A+B* respectively), comparing a separately selected specialist system for the prediction of Pitch *(S)*, with the Pitch predicting subset of a combined system for predicting Duration and Pitch *(C)*.

was chosen to be the subject of all following investigations. No principled criterion for halting viewpoint selection has yet been found; therefore an arbitrary criterion was adopted for all subsequent investigations which prevents the selection of a large number of viewpoints which do little to improve performance. Corpus 'B' produces much lower cross-entropies than corpus 'A' for no easily identifiable reason; but by a small margin, corpus 'A+B' produces the lowest cross-entropy overall, as would be expected for a larger corpus.

For the prediction of `Duration` alone, corpus 'A' produces lower cross-entropies than corpus 'B' (different from the prediction of `Duration` and `Pitch` together). As expected, corpus 'A+B' generally produces lower cross-entropies than either of the other corpora. On the other hand, for the prediction of `Pitch` alone, corpus 'B' produces much lower cross-entropies than corpus 'A'; and corpus 'A+B' produces slightly higher cross-entropies than corpus 'B'. By combining separately selected systems for the prediction of `Duration` and `Pitch`, it is possible to create slightly better models than those with a single system for the prediction of `Duration` and `Pitch` together. The best system overall, using corpus 'A+B' with an $\hbar$ of 6 and 3 for `Duration` and `Pitch` respectively, has a cross-entropy of 2.86 bits/note.

In the same way that the use of separately selected systems for `Duration` and `Pitch` can enhance performance, it is expected that different systems for LTM+ and STM in BOTH+ could result in an improvement. Preliminary work using version 1 models indicates that combining completely separately selected LTM and STM does not produce a better model; but it is conceivable that selecting an STM given an already selected LTM could improve performance (*i.e.*, the LTM is taken into account during the selection of the STM such that a complementary system emerges). This is interesting work to be followed up in the future.

Chapter 6

Prediction Performance Analysis of Versions 1 to 3

6.1 Introduction

The previous chapter demonstrated that for melodic modelling, BOTH+ using weighted geometric distribution combination performed best. Bearing in mind the time constraints of this research, it is assumed that BOTH+ will perform similarly well with respect to the modelling of harmony; therefore in this chapter we systematically search for the best performing version 1, version 2 and version 3 BOTH+ models of harmony (using weighted geometric combination) for the prediction of `Duration`, `Cont` and `Pitch`. We investigate the prediction of these attributes together and separately, using both the seen and augmented `Pitch` domains. Time constraints also dictate that only corpus ‘A’, a bias of 2 and an L-S bias of 14 are used for viewpoint selection (as for the best melodic BOTH+ runs using corpus ‘A’). Corpus ‘A+B’ is employed in conjunction with systems selected using corpus ‘A’, however. The three versions are compared one with another to ascertain the best performing. Please note that all bar charts in this chapter have a cross-entropy range of 2.5 bits/prediction, often not starting at zero.

Viewpoint selection method VS3 is used (see §3.4.5.2), which recognises that links with specified primitive or threaded viewpoints which have not necessarily been added to the viewpoint set are worth keeping under consideration with respect to linking with other viewpoints. The particular viewpoints kept under consideration here are `DurRatio`, `Interval`, `ScaleDegree`, `ScaleDegree` $\ominus$ `FirstInPhrase` and, for versions 1 and 2 only, `InScale` (since this viewpoint is not implemented for version 3). When predicting `Pitch` only, `Duration` and `Cont` are added to this list (please note that `Duration` was not added to the list in similar circumstances for version 0 viewpoint selection).

Various version 1 models are compared (in terms of performance) in §6.2, followed by version 2 in §6.3. Versions 1 and 2 are then compared in §6.4, followed by versions 1 and 3 in §6.5 and versions 2 and 3 in §6.6. The best version 1 to 3 models are directly compared, as far as is possible, in §6.7; and in §6.8 the best automatically selected version

0 to 3 multiple viewpoint systems are compared. Finally, the chapter is summarised and a conclusion given in §6.9.

6.2 Version 1

6.2.1 Prediction of Duration, Cont and Pitch Together

In the first set of viewpoint selection runs, the seen Pitch domain was used and maximum N-gram order $\hbar$ varied from 0 to 5. As before, for each multiple viewpoint system selected, the biases were optimised (again using the seen domain) and the resulting cross-entropy recorded. It can be seen from Figure 6.1 that an $\hbar$ of 1 produces the lowest cross-entropy, 4.43 bits/chord; but the size of the error bars compared with the cross-entropy differences means that we cannot be confident about this result. The optimised values of bias and L-S bias are 0.8 and 95.0 respectively.

The augmented Pitch domain was used in the second set of viewpoint selection and bias optimisation runs. Figure 6.1 shows that the cross-entropies are much higher than for the seen domain. This is due to the fact that harmonic progressions seen in one key are now valid progressions in any key (by way of viewpoints such as ScaleDegree), subject to voice range restrictions. More elements make their way into the constrained Pitch domains, which means that the probability of an element must, on average, be lower than before. In fact, the probabilities in this case are more realistic; therefore the higher cross-entropy here does not necessarily mean a worse model (*i.e.*, it is not a like for like comparison). Again, the lowest cross-entropy (5.23 bits/chord) is given by an $\hbar$ of 1 (with the usual caveat concerning error bars). The optimised values of bias and L-S bias are 0.8 and 105 respectively.

When the augmented Pitch domain is used in conjunction with the multiple viewpoint system selected using the seen Pitch domain to optimise the biases, the cross-entropies are slightly higher across the board when compared with the latter case. Bearing in mind that viewpoint selection using the augmented domain takes much longer (see Chapter 4), however, this is a reasonable trade-off. Once more, the lowest cross-entropy (5.26 bits/chord) is given by an $\hbar$ of 1. The optimised values of bias and L-S bias are 0.6 and 112 respectively. For the sake of brevity, a multiple viewpoint system for example selected using the seen domain and optimised using the augmented domain will henceforth be referred to as a *seen-aug system.* The better of comparable aug-aug and seen-aug systems will be used for versions 1 and 2 (but not 3, due to time constraints).

6.2.2 Prediction of Duration, Cont and Pitch Separately

In the first set of viewpoint selection runs, systems using seen and augmented Pitch domains are compared for the prediction of Duration alone. Figure 6.2 shows that for any given value of $\hbar$, cross-entropies are virtually identical; and there is also very little change in cross-entropy with $\hbar$. The lowest cross-entropy in each case, 0.68 bits/chord,

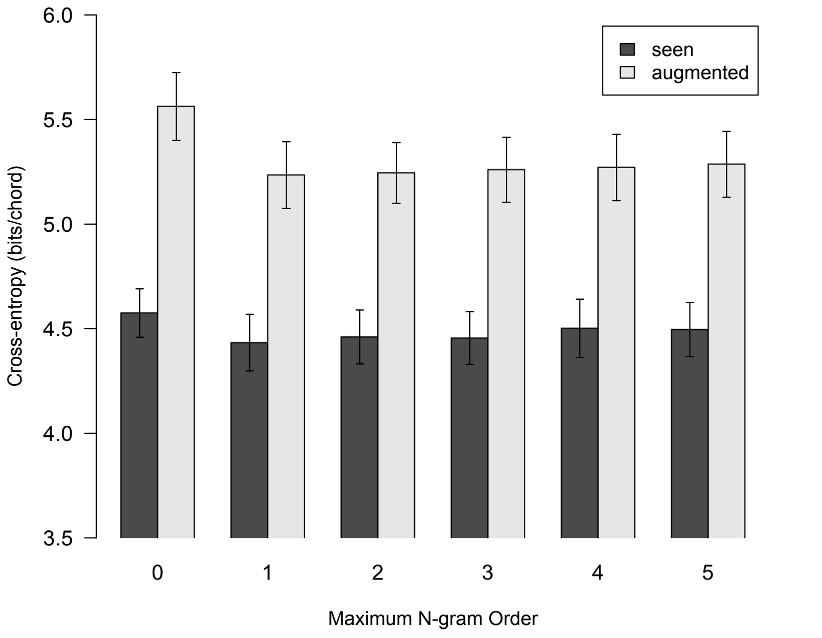


Figure 6.1: Bar chart showing how cross-entropy varies with $\hbar$ for the version 1 prediction of `Duration`, `Cont` and `Pitch` together (*i.e.*, using a single multiple viewpoint system) in the alto, tenor and bass given soprano. Systems using seen and augmented `Pitch` domains are compared.

occurs at an $\hbar$ of 0 (optimised bias and L-S bias are 6.0 and 1.2 respectively in each case).

Next, systems using seen and augmented `Pitch` domains are compared for the prediction of `Cont` alone. It can be seen from Figure 6.3 that cross-entropies for the augmented domain models are higher than those for the seen domain models. This is due to an increase (on average) in the number of elements appearing in constrained `Cont` domains, which in turn is a consequence of the much larger `Pitch` domain. In this case, seen-aug system cross-entropies are very slightly lower than aug-aug system ones. There is very little change in cross-entropy with $\hbar$, but for the seen domain model the minimum cross-entropy (0.60 bits/chord) occurs at an $\hbar$ of 0 (optimised bias and L-S bias are 0.2 and 140 respectively). The minimum cross-entropy for the augmented domain model (0.65 bits/chord) is produced by an $\hbar$ of 1 (optimised bias and L-S bias are 0.2 and 23.0 respectively).

Finally, systems using seen and augmented `Pitch` domains are compared for the prediction of `Pitch` alone. Figure 6.4 shows that the cross-entropies for the augmented domain models are much higher than those for the seen domain models (as noted earlier, the augmented domain probabilities are more realistic). As expected, the seen-aug cross-entropies are higher than the aug-aug ones. The cross-entropy difference is such that it is not a good trade-off with viewpoint selection running time in this case. For the seen domain model, the minimum cross-entropy (3.08 bits/chord) occurs at an $\hbar$ of 1

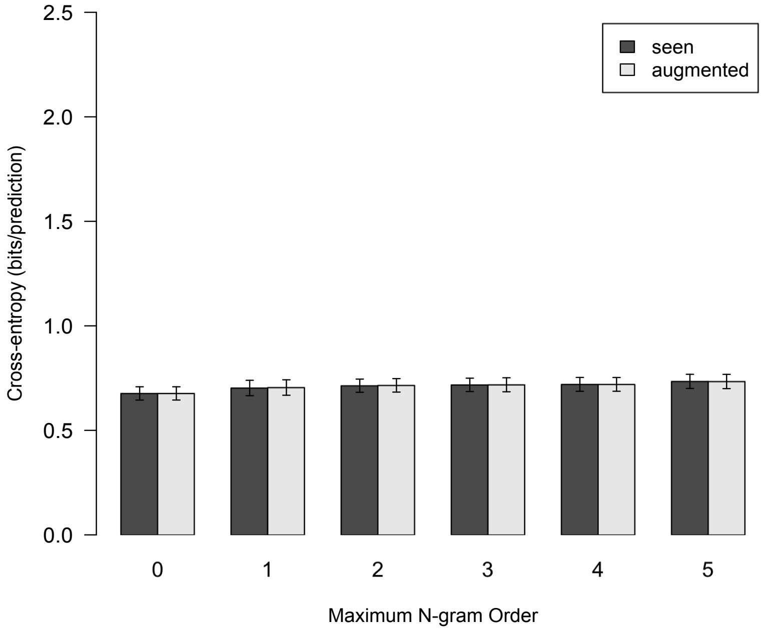


Figure 6.2: Bar chart showing how cross-entropy varies with $\hbar$ for the version 1 prediction of `Duration` only in the alto, tenor and bass given soprano. Systems using seen and augmented `Pitch` domains are compared.

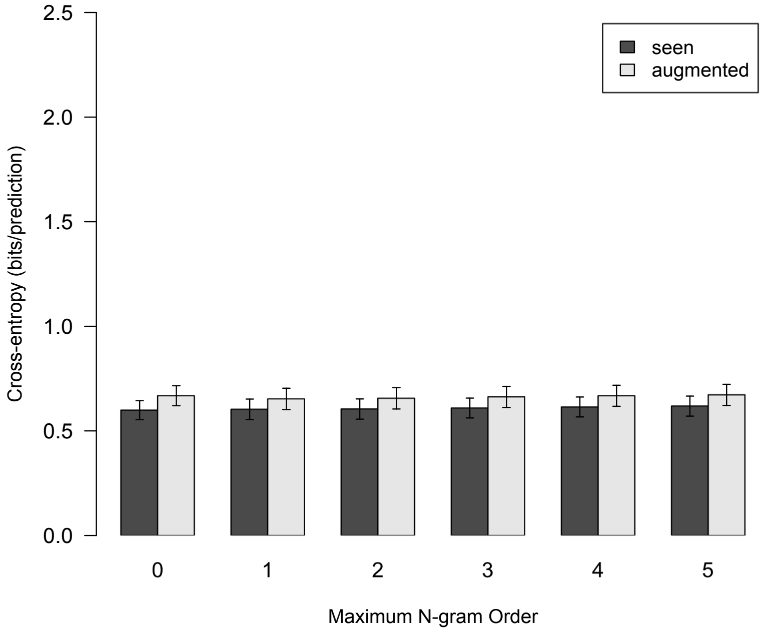


Figure 6.3: Bar chart showing how cross-entropy varies with $\hbar$ for the version 1 prediction of `Cont` only in the alto, tenor and bass given soprano. Systems using seen and augmented `Pitch` domains are compared.

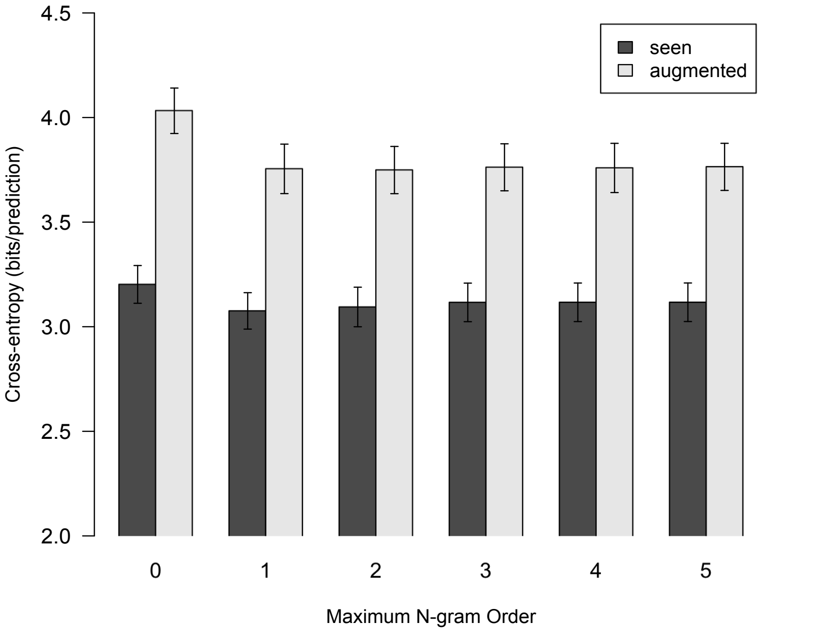


Figure 6.4: Bar chart showing how cross-entropy varies with $\hbar$ for the version 1 prediction of `Pitch` only in the alto, tenor and bass given soprano. Systems using seen and augmented `Pitch` domains are compared.

(optimised bias and L-S bias are 1.0 and 136 respectively). The minimum cross-entropy for the augmented domain model (3.75 bits/chord) is produced by an $\hbar$ of 2 (optimised bias and L-S bias are 0.7 and 140 respectively).

6.2.3 Combining Systems Predicting `Duration`, `Cont` and `Pitch` Separately

In order to compare the performance of models which predict `Duration`, `Cont` and `Pitch` separately with those predicting them together, over a range of $\hbar$, the cross-entropies of the `Duration`, `Cont` and `Pitch` prediction models are simply added. The comparison in Figure 6.5, using seen `Pitch` domain models, shows that the selection and optimisation of specialist multiple viewpoint systems for the prediction of individual attributes produces slightly lower cross-entropies across the board, as was the case for version 0. The lowest cross-entropy, occurring at an $\hbar$ of 1, is 4.38 bits/chord. The very best `Duration` and `Cont` prediction models have an $\hbar$ of 0, however (see §6.2.2), and by combining these with the $\hbar = 1$ Pitch model the cross-entropy is reduced to 4.35 bits/chord (0.08 bits/chord better than the best model predicting the three attributes together).

Figure 6.6 shows that for a similar comparison using the augmented domain, the difference in cross-entropy is greater. Again, the lowest cross-entropy (5.11 bits/chord) occurs at an $\hbar$ of 1. This time, however, the very best `Duration`, `Cont` and `Pitch` models have an $\hbar$ of 0, 1 and 2 respectively (see §6.2.2). Combining these models reduces the cross-entropy to 5.08 bits/chord, which is 0.15 bits/chord better than the best model

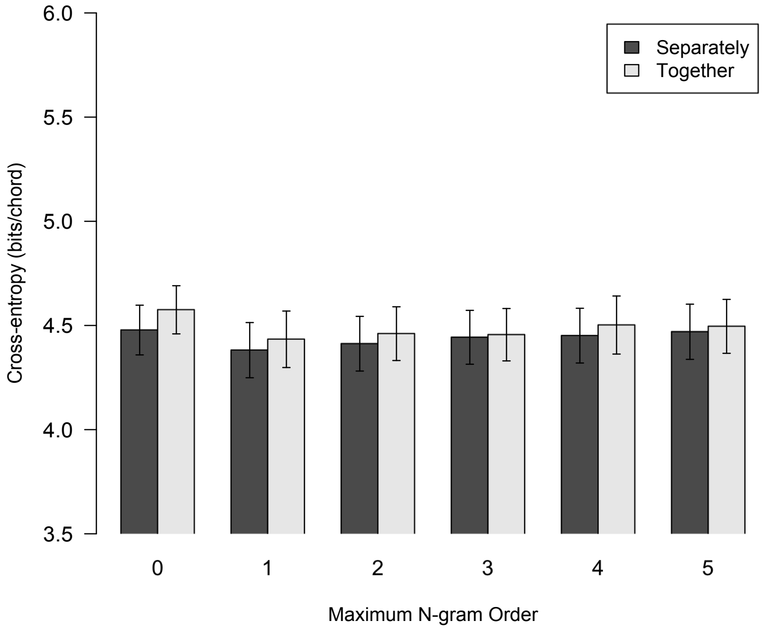


Figure 6.5: Bar chart showing how cross-entropy varies with $\hbar$ for the version 1 prediction of alto, tenor and bass given soprano using the seen `Pitch` domain. The prediction of `Duration`, `Cont` and `Pitch` separately (*i.e.*, using separately selected multiple viewpoint systems) and together (*i.e.*, using a single multiple viewpoint system) are compared.

predicting the three attributes together.

6.2.4 Corpus 'A+B'

The next comparison is carried out in conjunction with the augmented `Pitch` domain. The systems selected using corpus 'A' to predict the three attributes together are employed to compare the effect of using corpus 'A+B' rather than corpus 'A'. It can be seen from Figure 6.7 that a large reduction in cross-entropy results from the use of the larger corpus. Notice also the large reduction in the size of the standard errors. There is an obvious difference in the shape of the curve made by the two sets of bars: the minimum shifts from $\hbar = 1$ for corpus 'A' to $\hbar = 3$ for corpus 'A+B'. This is due to the statistics for higher values of $\hbar$ becoming less sparse as the corpus size increases. The best model, with a bias and L-S bias of 0.9 and 100 respectively,[1] has a cross-entropy of 5.00 bits/chord (an improvement of 0.24 bits/chord over the best model using corpus 'A').

Finally, the composite systems selected using corpus 'A' to predict the attributes separately are also used to compare corpus 'A+B' with corpus 'A'. Figure 6.8 shows a similarly large drop in cross-entropy; but this time the minimum moves from $\hbar = 1$ (only very slightly better than $\hbar = 2$) to $\hbar = 4$. The best model has a cross-entropy of

[1] 100 was the largest value of L-S bias tried, rather than the usual maximum of 140, because of floating point arithmetic errors. The likely difference in cross-entropy is minuscule.

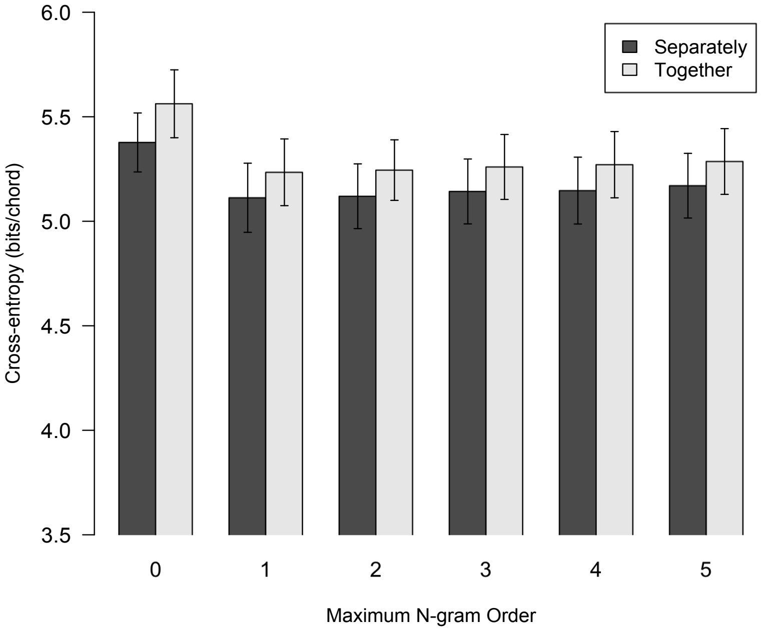


Figure 6.6: Bar chart showing how cross-entropy varies with $\hbar$ for the version 1 prediction of alto, tenor and bass given soprano using the augmented `Pitch` domain. The prediction of `Duration`, `Cont` and `Pitch` separately (*i.e.*, using separately selected multiple viewpoint systems) and together (*i.e.*, using a single multiple viewpoint system) are compared.

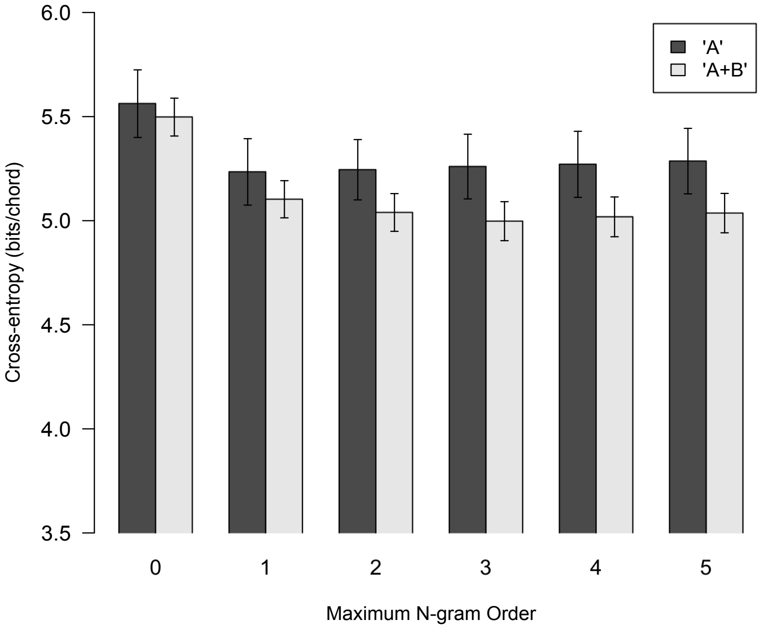


Figure 6.7: Bar chart showing how cross-entropy varies with $\hbar$ for the version 1 prediction of `Duration`, `Cont` and `Pitch` together (*i.e.*, using a single multiple viewpoint system) in the alto, tenor and bass given soprano using the augmented `Pitch` domain. Corpora ‘A’ and ‘A+B’ are compared.

4.87 bits/chord (an improvement of 0.21 bits/chord over the best model using corpus ‘A’), making it the best version 1 model overall. See Table 6.1 for a brief summary of subsystem parameters and cross-entropies for this model.[2]

[2] 100 was the largest value of L-S bias tried for the `Pitch`-predicting system.

predicting	$\hbar$	bias	L-S bias	x-entropy
`Duration`	4	2.1	3.8	0.69
`Cont`	2	0.1	19.3	0.71
`Pitch`	4	1.1	100	3.47

Table 6.1: Summary of subsystem parameters and cross-entropies for the best version 1 model (prediction of `Duration`, `Cont` and `Pitch` using separate multiple viewpoint systems selected in conjunction with the augmented `Pitch` domain and corpus ‘A’, but using corpus ‘A+B’). *Cross-entropy* is abbreviated to *x-entropy*.

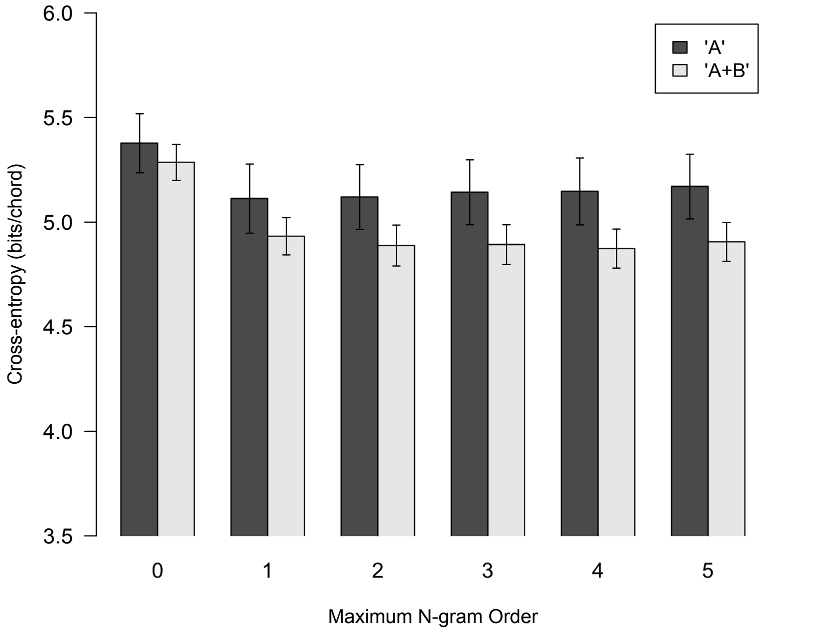


Figure 6.8: Bar chart showing how cross-entropy varies with $\hbar$ for the version 1 prediction of `Duration`, `Cont` and `Pitch` separately (*i.e.*, using separately selected multiple viewpoint systems) in the alto, tenor and bass given soprano using the augmented `Pitch` domain. Corpora 'A' and 'A+B' are compared.

6.3 Version 2

6.3.1 Prediction of `Duration`, `Cont` and `Pitch` Together

6.3.1.1 Seen `Pitch` Domain

In this section we carry out the prediction of bass given soprano, alto/tenor given soprano/bass, tenor given soprano, alto/bass given soprano/tenor, alto given soprano, and tenor/bass given soprano/alto (*i.e.*, prediction in two stages), in order to ascertain the best performing combination for subsequent comparisons. Prediction in three stages is not considered here because of time limitations.

Figure 6.9 compares the prediction of alto given soprano, tenor given soprano, and bass given soprano. Prediction of the alto part has the lowest cross-entropy and prediction of the bass has the highest across the board. This is very likely to be due to the relative number of elements in the `Pitch` domains for the individual parts (*i.e.*, 18, 20 and 23 for alto, tenor and bass respectively). The lowest cross-entropies occur at an $\hbar$ of 1 except for the bass, which has its minimum at an $\hbar$ of 2 (this cross-entropy is only very slightly lower than that for an $\hbar$ of 1, however).

There is a completely different picture for the final stage of prediction. Figure 6.10 shows that, having predicted the alto part with a low cross-entropy, the prediction of tenor/bass has the highest. Similarly, the high cross-entropy for the prediction of the bass is complemented by an exceptionally low cross-entropy for the prediction of

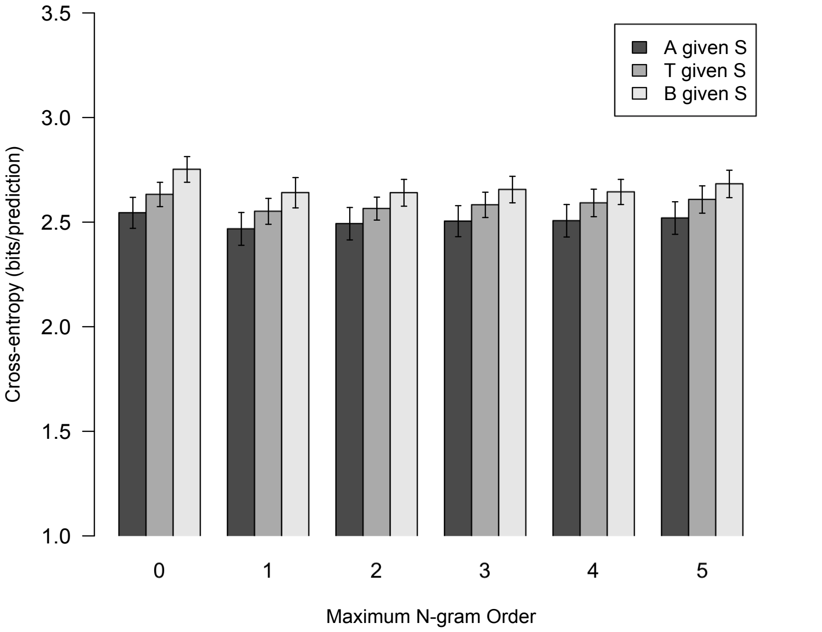


Figure 6.9: Bar chart showing how cross-entropy varies with $\hbar$ for the version 2 prediction of alto given soprano, tenor given soprano, and bass given soprano using the seen `Pitch` domain.

alto/tenor (notice that the error bars do not overlap with those of the other prediction combinations). Once again, this can be explained by the number of elements in the part domains: the sizes of the cross-product domains are 460, 414 and 360 for tenor/bass, alto/bass and alto/tenor respectively. Although we are not using cross-product domains, it is likely that the seen domains are in similar proportion. The lowest cross-entropies occur at an $\hbar$ of 1.

Combining the two stages of prediction, we see in Figure 6.11 that predicting bass first and then alto/tenor is best (*i.e.*, this combination has the lowest cross-entropy), reflecting the usual human approach to harmonisation. The lowest cross-entropy is 4.98 bits/chord, occurring at an $\hbar$ of 1. Although having the same cross-entropy to two decimal places, the very best model combines the bass-predicting model using an $\hbar$ of 2 (optimised bias and L-S bias are 1.9 and 53.2 respectively) with the alto/tenor-predicting model using an $\hbar$ of 1 (optimised bias and L-S bias are 1.3 and 99.6 respectively).

6.3.1.2 Augmented `Pitch` Domain

Figure 6.12 shows that for the prediction of bass given soprano, cross-entropies for the augmented `Pitch` domain models are only a little higher than those for the seen domain models. This is because the vast majority of soprano/bass combinations in the augmented domain have already been seen (*i.e.*, are in the seen domain). The lowest cross-entropy for the augmented domain model is 2.71 bits/prediction at an $\hbar$ of 1 (optimised bias and L-S bias are 1.8 and 36.8 respectively) compared with 2.64

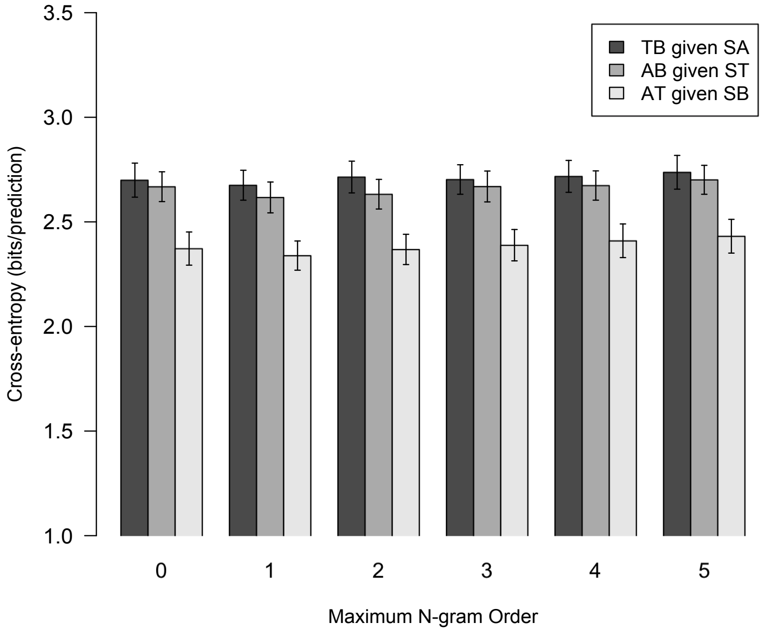


Figure 6.10: Bar chart showing how cross-entropy varies with $\hbar$ for the version 2 prediction of tenor/bass given soprano/alto, alto/bass given soprano/tenor and alto/tenor given soprano/bass using the seen `Pitch` domain.

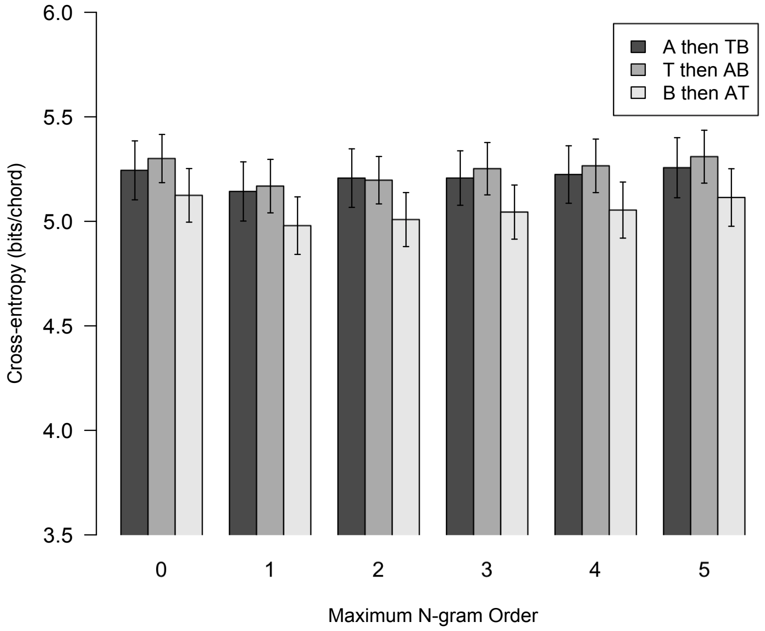


Figure 6.11: Bar chart showing how cross-entropy varies with $\hbar$ for the version 2 prediction of alto then tenor/bass, tenor then alto/bass and bass then alto/tenor given soprano using the seen `Pitch` domain.

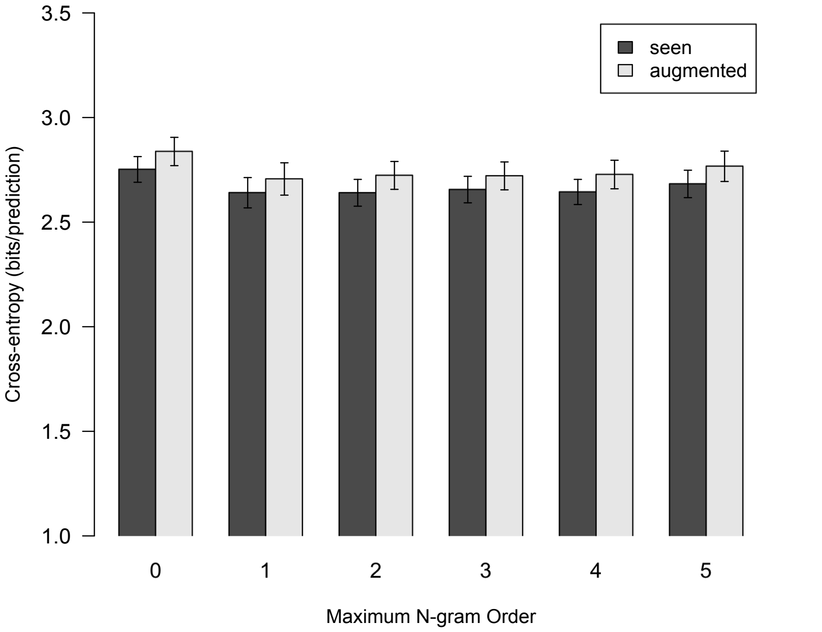


Figure 6.12: Bar chart showing how cross-entropy varies with $\hbar$ for the version 2 prediction of bass given soprano using the seen and augmented `Pitch` domains.

bits/prediction for the seen domain model.

On the other hand, Figure 6.13 shows a wide gulf in cross-entropies for the prediction of alto/tenor given soprano/bass, with those of the augmented `Pitch` domain models very much higher than those of the seen domain models. This is explained by the fact that there are many more alto/tenor combinations (given soprano/bass) in the augmented domain. The lowest cross-entropy for the augmented domain model is 2.91 bits/prediction at an $\hbar$ of 1 (optimised bias and L-S bias are 1.3 and 55.6 respectively) compared with 2.34 bits/prediction for the seen domain model. From this point onwards only the augmented `Pitch` domain is used, on the basis that it produces more realistic prediction probabilities.

6.3.2 Prediction of `Duration`, `Cont` and `Pitch` Separately

We now investigate the use of separately selected and optimised multiple viewpoint systems for the prediction of `Duration`, `Cont` and `Pitch`. Please note that in this subsection, all bar charts have a cross-entropy range of 0.0 to 2.5 bits/prediction.

Figure 6.14 shows that there is a much lower cross-entropy for the prediction of `Duration` in the alto/tenor given soprano/bass than for bass given soprano. This is due to the fact that the largest `Duration` value in the prediction probability distribution for bass given soprano is equal to the total (remaining) length of the soprano note to be harmonised (*i.e.*, including any note continuations), whereas the largest such value for alto/tenor given soprano/bass is equal to the length of the bass note after full expansion

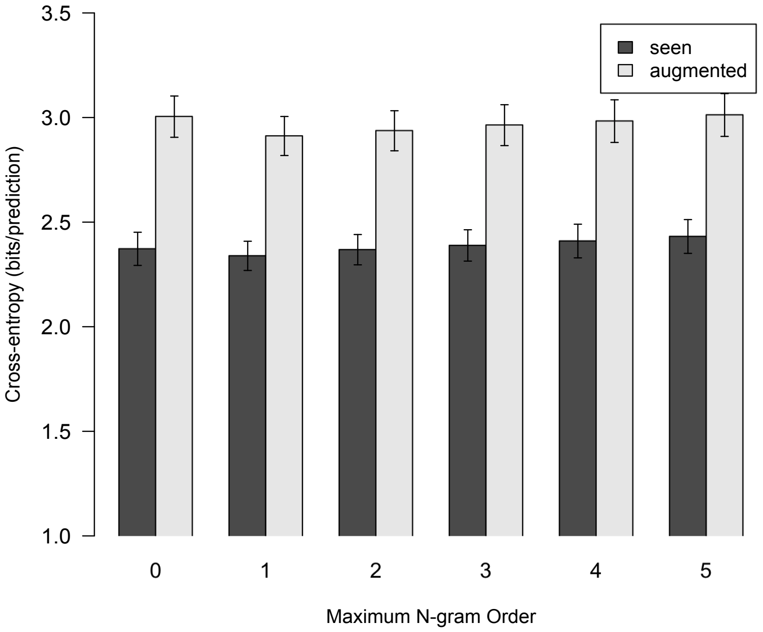


Figure 6.13: Bar chart showing how cross-entropy varies with $\hbar$ for the version 2 prediction of alto/tenor given soprano/bass using the seen and augmented `Pitch` domains.

(*i.e.*, excluding any note continuations). The latter distribution therefore contains fewer predictions of higher probability. Prediction of `Duration` is carried out in this way in order to be consistent with the generation procedure, which utilises an unexpanded melody. An $\hbar$ of 0 is optimal in both cases. The lowest cross-entropy for bass given soprano is 0.67 bits/prediction (optimised bias and L-S bias are 5.7 and 1.2 respectively) compared with 0.47 bits/prediction for alto/tenor given soprano/bass (with an optimised bias and L-S bias of 9.9 and 1.4 respectively).

Figure 6.15, on the other hand, indicates a slightly higher cross-entropy for the prediction of `Cont` in the alto/tenor given soprano/bass than for bass given soprano. This is explicable by the fact that there are at most two predictions in the bass given soprano probability distribution compared with up to four in the alto/tenor given soprano/bass one. Notice how much lower these `Cont` cross-entropies are compared with the `Duration` ones, suggesting that `Cont` is easier to predict than `Duration`. This is very likely true for corpus 'A', which contains relatively few passing notes (and therefore relatively few continuations in other parts). An $\hbar$ of 0 is optimal in each case. The lowest cross-entropy for bass given soprano is 0.26 bits/prediction (optimised bias and L-S bias are 0.1 and 140 respectively) compared with 0.28 bits/prediction for alto/tenor given soprano/bass (with an optimised bias and L-S bias of −0.2 and 3.6 respectively). Note that this is the first time we have seen a negative bias.

In a similar way, Figure 6.16 shows a much higher cross-entropy for the prediction of `Pitch` in the alto/tenor given soprano/bass than for bass given soprano, resulting from the far larger prediction probability distributions in the former case. An $\hbar$ of

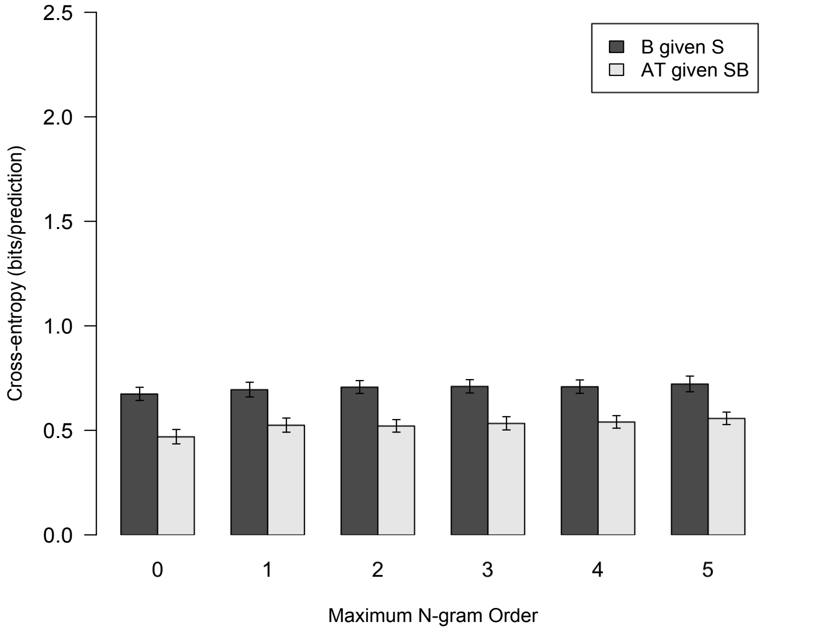


Figure 6.14: Bar chart showing how cross-entropy varies with $\hbar$ for the version 2 prediction of `Duration` only in the bass given soprano and alto/tenor given soprano/bass using the augmented `Pitch` domain.

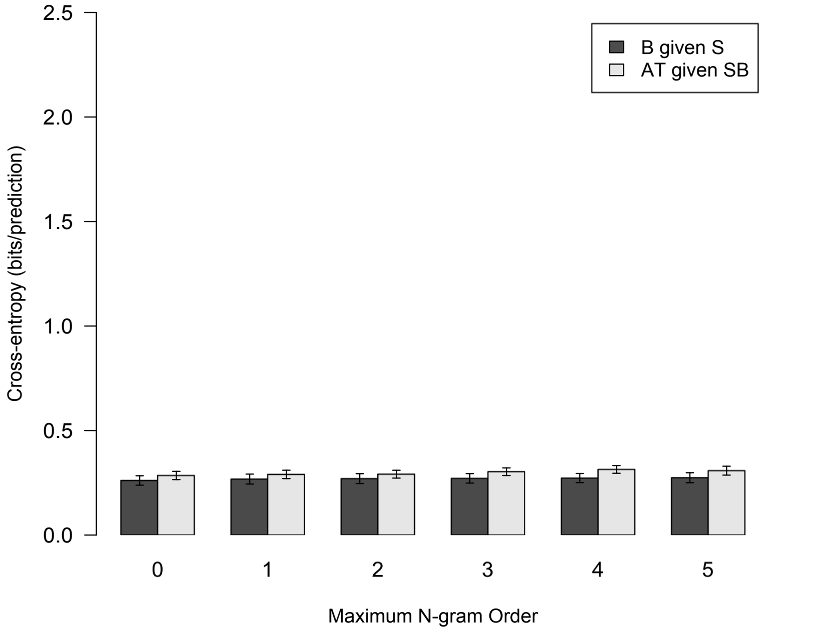


Figure 6.15: Bar chart showing how cross-entropy varies with $\hbar$ for the version 2 prediction of `Cont` only in the bass given soprano and alto/tenor given soprano/bass using the augmented `Pitch` domain.

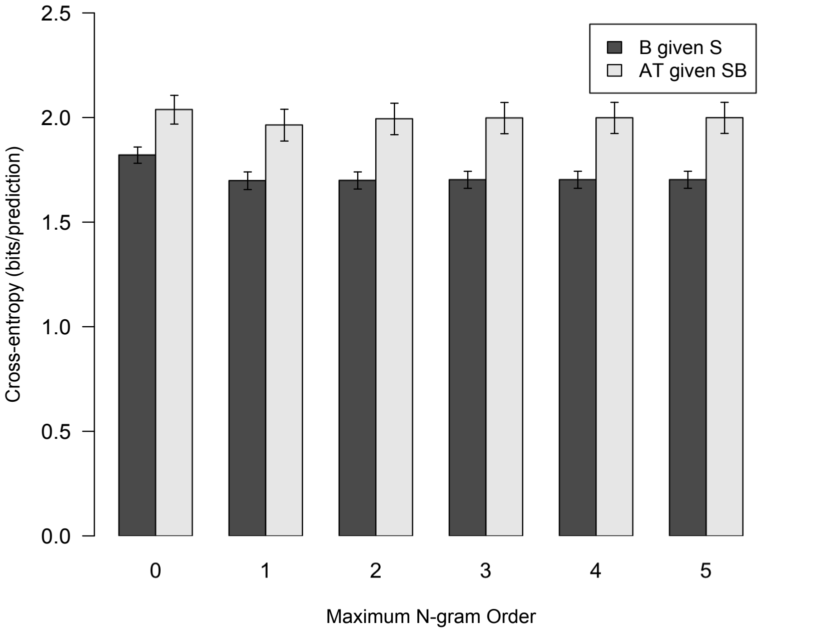


Figure 6.16: Bar chart showing how cross-entropy varies with $\hbar$ for the version 2 prediction of `Pitch` only in the bass given soprano and alto/tenor given soprano/bass using the augmented `Pitch` domain.

1 is optimal in both cases. The lowest cross-entropy for bass given soprano is 1.70 bits/prediction (optimised bias and L-S bias are 1.5 and 54.0 respectively) compared with 1.96 bits/prediction for alto/tenor given soprano/bass (with an optimised bias and L-S bias of 1.6 and 110 respectively).

6.3.3 Combining Systems Predicting `Duration`, `Cont` and `Pitch` Separately

Figure 6.17 shows, once again, that better models can be created by selecting separate multiple viewpoint systems to predict individual attributes, rather than a single system to predict all of them. The difference in cross-entropy is quite marked in this instance. An $\hbar$ of 1 is optimal in both cases. The lowest cross-entropy for separate prediction at $\hbar = 1$ is 5.44 bits/chord, compared with 5.62 bits/chord for prediction together. The very best model for separate prediction, with a cross-entropy of 5.35 bits/chord, comprises the best performing systems (of whatever value of $\hbar$) from §6.3.2.

In the next comparison, the composite systems selected using corpus 'A' to predict the three attributes separately are employed to compare the effect of using corpus 'A+B' rather than corpus 'A'. It can be seen from Figure 6.18 that there is a much smaller reduction in cross-entropy resulting from the use of the larger corpus than occurred in the corresponding version 1 case (see Figure 6.8). Again there is a large reduction in the size of the standard errors, and the minimum shifts from $\hbar = 1$ for corpus 'A' to $\hbar = 2$ for corpus 'A+B'. The best model has a cross-entropy of 5.32 bits/chord (an

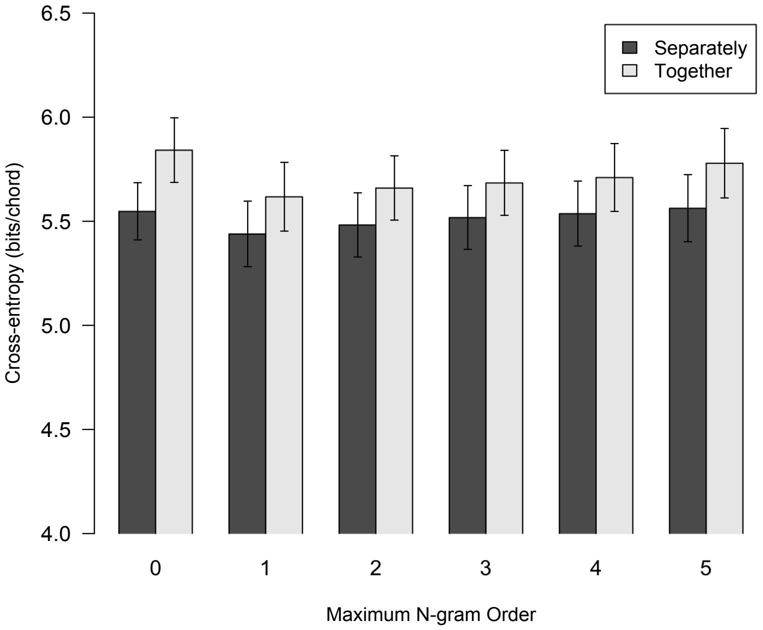


Figure 6.17: Bar chart showing how cross-entropy varies with $\hbar$ for the version 2 prediction of bass given soprano followed by alto/tenor given soprano/bass using the augmented `Pitch` domain. The prediction of `Duration`, `Cont` and `Pitch` separately (*i.e.*, using separately selected multiple viewpoint systems) and together (*i.e.*, using a single multiple viewpoint system) are compared.

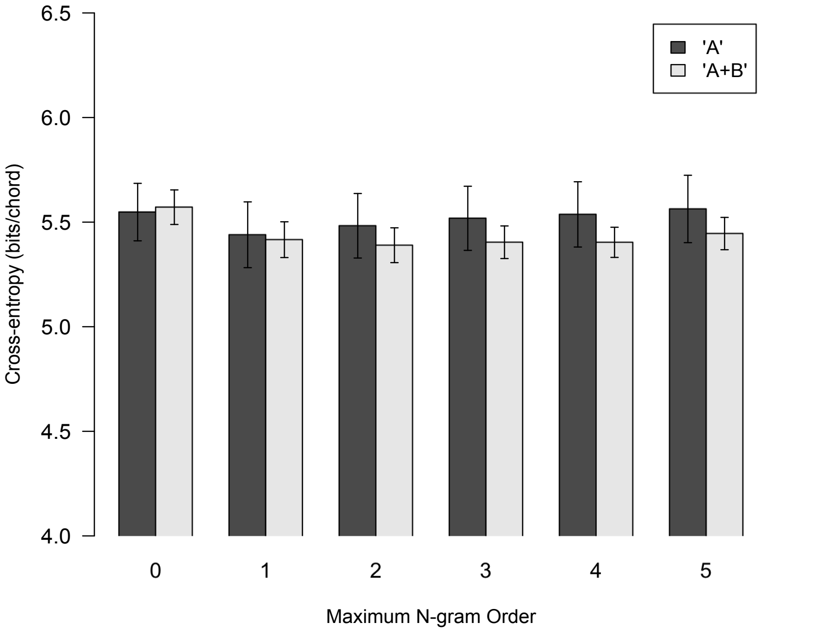


Figure 6.18: Bar chart showing how cross-entropy varies with $\hbar$ for the version 2 prediction of `Duration`, `Cont` and `Pitch` separately (*i.e.*, using separate multiple viewpoint systems selected in conjunction with corpus 'A') in the bass given soprano followed by alto/tenor given soprano/bass using the augmented `Pitch` domain. Corpora 'A' and 'A+B' are compared.

improvement of only 0.03 bits/chord over the best model using corpus 'A'), making it the best version 2 model overall. See Table 6.2 for a brief summary of subsystem parameters and cross-entropies for the model. There is a big improvement in the `Pitch` prediction cross-entropy for the larger corpus, but this is almost completely negated by a deterioration in the `Cont` prediction cross-entropy. This could be due to corpus 'B' containing many more passing notes than corpus 'A'.

6.4 Comparison of Version 1 with Version 2

6.4.1 Prediction of `Duration`, `Cont` and `Pitch` Together

Figure 6.19 shows that version 2 has a substantially higher cross-entropy than version 1. This is to be expected due to the fact that whereas the duration of an entire chord is predicted only once in version 1, it is effectively predicted twice[3] in version 2. Prediction of `Duration` is set up such that, for example, a minim may be generated in the bass given soprano generation stage, followed by a crotchet in the final generation stage, whereby the whole of the chord becomes a crotchet. This is different from the prediction and generation of `Cont` and `Pitch`, where elements generated in the first stage are not subject to change in the second. The way in which the prediction of `Duration` is treated, then,

[3] `Duration` is predicted three times if the lower parts are predicted individually.

predicting	stage	$\hbar$	bias	L-S bias	x-entropy
`Duration`	B given S	4	2.2	3.0	0.68
`Duration`	AT given SB	0	140	1.3	0.46
`Cont`	B given S	1	0.4	39.6	0.28
`Cont`	AT given SB	2	0.4	19.1	0.43
`Pitch`	B given S	2	1.3	56.5	1.62
`Pitch`	AT given SB	1	1.5	82.6	1.84

Table 6.2: Summary of subsystem parameters and cross-entropies for the best version 2 model (prediction of `Duration`, `Cont` and `Pitch` using separate multiple viewpoint systems selected in conjunction with the augmented `Pitch` domain and corpus ‘A’, but using corpus ‘A+B’). *Cross-entropy* is abbreviated to *x-entropy*.

means that versions 1 and 2 are not directly comparable when predicting the three attributes using a single subtask model. We shall, however, be able to make meaningful comparisons in §6.4.2.

6.4.2 Prediction of `Duration`, `Cont` and `Pitch` Separately

Figure 6.20 confirms that there is a huge difference in cross-entropy between versions 1 and 2 because of the way `Duration` is predicted. In each case, an $\hbar$ of 0 is optimal. Notice that version 1 has by far the tighter error range. Moving on to Figure 6.21 we see the first meaningful comparison, for the prediction of `Cont`. Version 2 clearly has lower cross-entropies; and judging from the lack of overlap between the error bars, it seems likely that this is a real improvement over version 1. The optimal version 2 cross-entropy ($\hbar = 0$) is 0.11 bits/prediction lower than that of version 1 ($\hbar = 1$).

The situation is not so clear cut with respect to the `Pitch` prediction comparison in Figure 6.22. The version 2 cross-entropies do appear to be a little lower; but the large overlap in the error bars suggests that we cannot rely on this generally being the case. The optimal version 2 cross-entropy ($\hbar = 1$) is 0.09 bits/prediction lower than that of version 1 ($\hbar = 2$).

6.4.3 Combining Systems Predicting `Cont` and `Pitch` Separately

By ignoring `Duration` prediction, and combining only the directly comparable `Cont` and `Pitch` cross-entropies, we can make a judgement on the overall relative performance of these two versions. Figure 6.23 is strongly indicative of version 2 performing better than version 1. As one might expect, it would appear that the selection of specialist multiple viewpoint systems for the prediction of different parts is beneficial in rather the same way as specialist systems for the prediction of the various attributes. The optimal version 2 cross-entropy, using the best subtask models irrespective of the value of $\hbar$, is 0.19 bits/prediction lower than that of version 1.

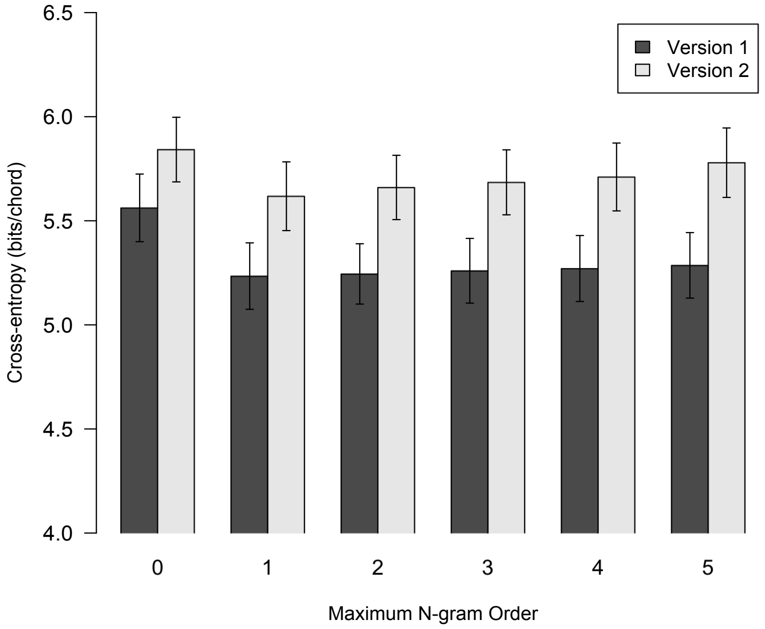


Figure 6.19: Bar chart showing how cross-entropy varies with $\hbar$ for the prediction of alto, tenor and bass given soprano using the augmented `Pitch` domain, comparing version 1 with version 2.

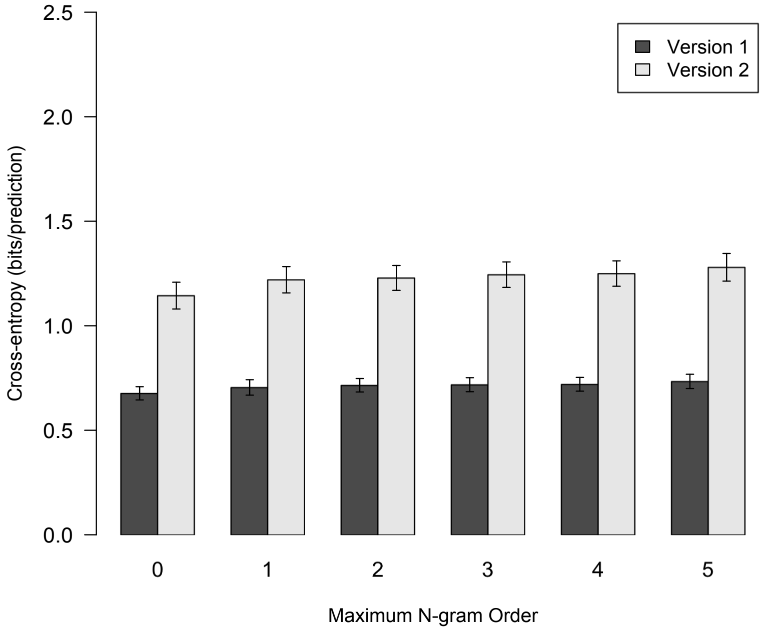


Figure 6.20: Bar chart showing how cross-entropy varies with $\hbar$ for the prediction of `Duration` in the alto, tenor and bass given soprano using the augmented `Pitch` domain, comparing version 1 with version 2.

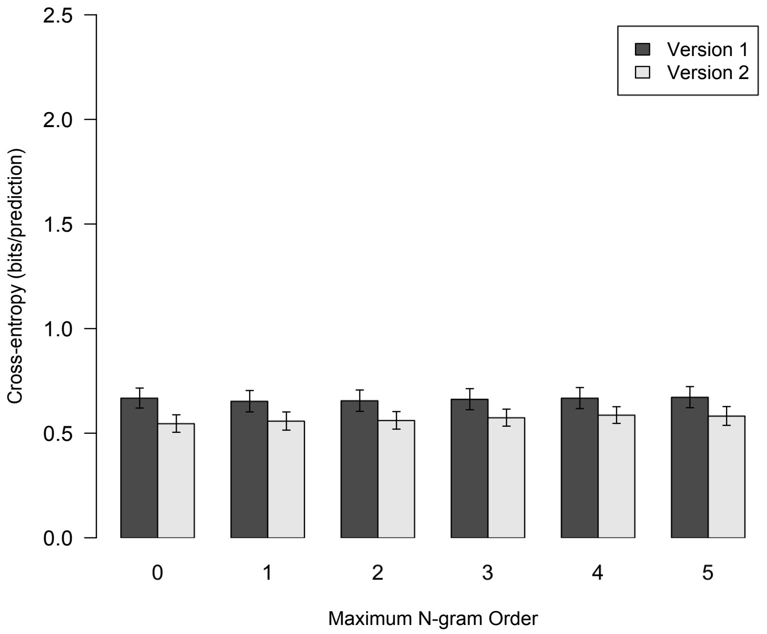


Figure 6.21: Bar chart showing how cross-entropy varies with $\hbar$ for the prediction of Cont in the alto, tenor and bass given soprano using the augmented Pitch domain, comparing version 1 with version 2.

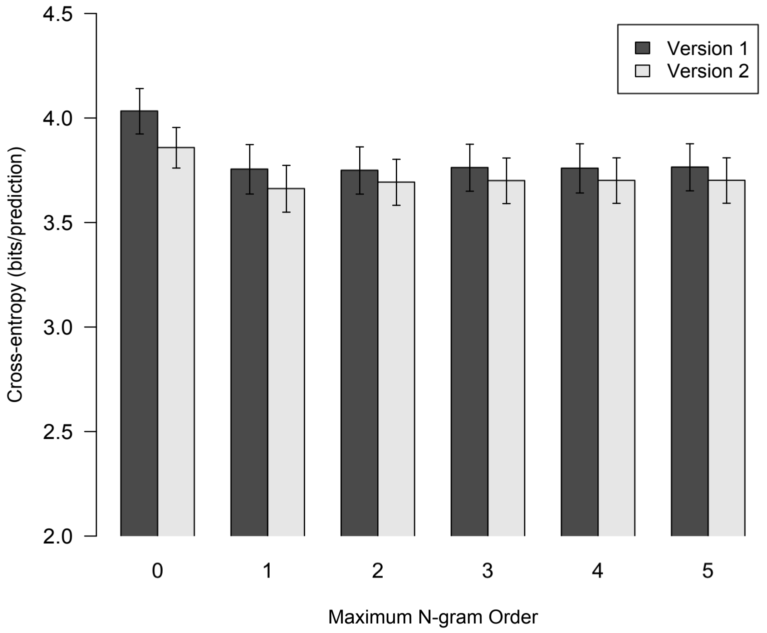


Figure 6.22: Bar chart showing how cross-entropy varies with $\hbar$ for the prediction of Pitch in the alto, tenor and bass given soprano using the augmented Pitch domain, comparing version 1 with version 2.

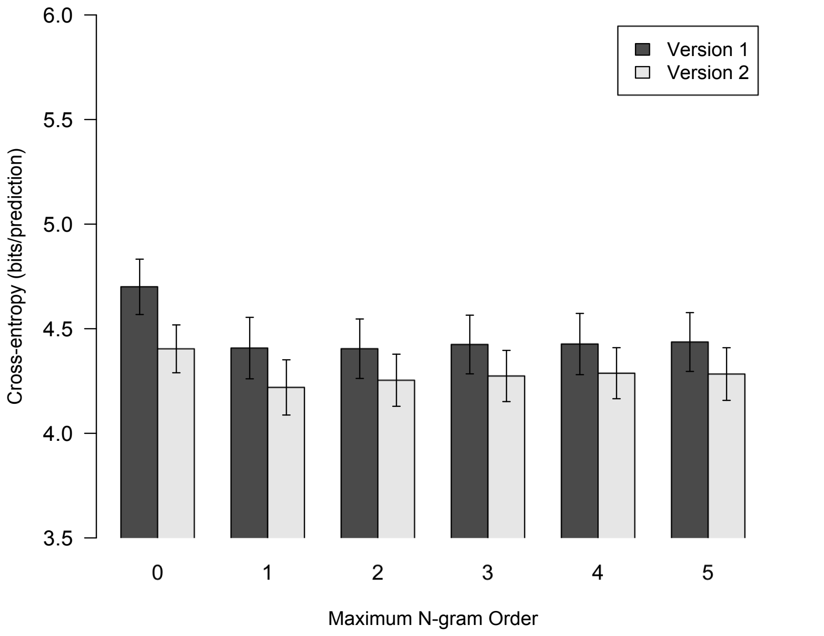


Figure 6.23: Bar chart showing how cross-entropy varies with $\hbar$ for the separate prediction of `Cont` and `Pitch` in the alto, tenor and bass given soprano using the augmented `Pitch` domain, comparing version 1 with version 2.

6.4.4 Corpus ‘A+B’

Finally, the systems selected using corpus ‘A’ are used in conjunction with corpus ‘A+B’. Compared with Figure 6.23, Figure 6.24 shows a much larger drop in cross-entropy for version 1 than for version 2: indeed, the minimum cross-entropies are exactly the same. The only saving grace for version 2 is that the error bars are slightly smaller. We can infer from this that version 1 creates more general models, better able to scale up to larger corpora which may deviate somewhat from the characteristics of the original corpus. Conversely, version 2 is capable of constructing models which are more specific to the corpus for which they are selected. This hypothesis can easily be tested by carrying out viewpoint selection in conjunction with corpus ‘A+B’ (although this would be a very time-consuming process).

6.5 Comparison of Version 1 with Version 3

In this section we investigate the version 3 prediction of alto/tenor/bass given soprano, since this is directly comparable with version 1. From this point onwards prediction of basic attributes together is no longer considered, since separate prediction has been shown to have a performance advantage.

It is important to note at this point that version 3 software errors were discovered

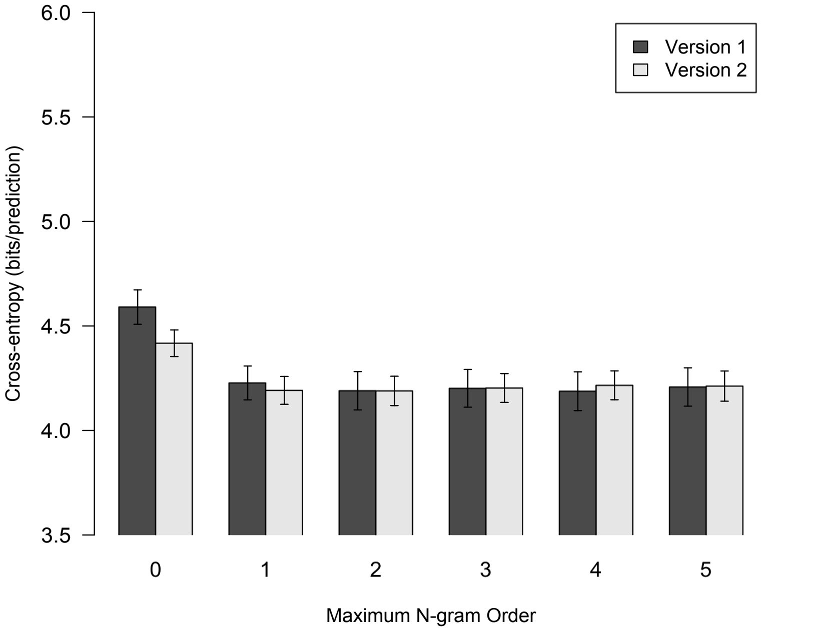


Figure 6.24: Bar chart showing how cross-entropy varies with $\hbar$ for the separate prediction of `Cont` and `Pitch` in the alto, tenor and bass given soprano using the augmented `Pitch` domain and corpus 'A+B' with systems selected using corpus 'A', comparing versions 1 and 2.

after the completion of extremely time-consuming[4] viewpoint selection runs. The errors meant that models were not properly updated during prediction, and undefined viewpoints were often not recognised as such. In order to avoid a further lengthy period of viewpoint selection following correction of the errors, it was decided to use the already selected multiple viewpoint systems as a starting point. Each system went through an additional viewpoint deletion process such that all viewpoint deletions resulting in a cross-entropy reduction were made permanent. The bias and L-S bias were then re-optimised.

6.5.1 Combining Systems Predicting `Duration`, `Cont` and `Pitch` Separately

Please note that comparisons for the individual prediction of `Duration`, `Cont` and `Pitch` can be found in Appendix C (§C.1).

Figure 6.25 shows that for the overall model, version 3 is convincingly better than version 1. The minimum version 3 cross-entropy at a single $\hbar$ value is 4.70 bits/chord at an $\hbar$ of 3, compared with 5.11 bits/chord ($\hbar = 1$) for version 1. Employing the best subtask models irrespective of the value of $\hbar$ reduces the version 3 cross-entropy to 4.65

[4] In spite of the smaller pool of viewpoints (see §3.4.3) and the fact that the criterion for removing viewpoints from further consideration during viewpoint selection was changed to a margin equal to the smaller of 0.01 or 0.4δ bits/symbol (see §3.4.5.3) for version 3.

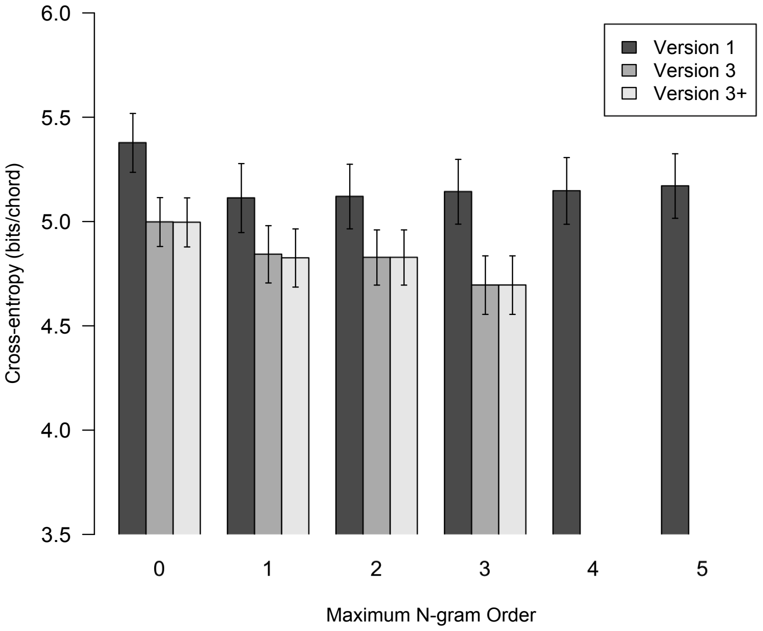


Figure 6.25: Bar chart showing how cross-entropy varies with $\hbar$ for the separate prediction of `Duration`, `Cont` and `Pitch` in the alto, tenor and bass given soprano using the augmented `Pitch` domain, comparing versions 1, 3 and 3+ (a hybrid of versions 1 and 3).

bits/chord and the version 1 cross-entropy to 5.08 bits/chord.

Bearing in mind that the set of possible version 1 systems is a subset of the set of possible version 3 systems, we can theoretically further improve the overall models by using the better of the version 1 and 3 subtask models.[5] The resulting models are referred to as version 3+. Here, the `Cont` predicting subtask models are version 1 for $\hbar$ values of 0 and 1. We can see from Figure 6.25 that the cross-entropy differences are minute in this case, and the optimal version 3+ model (using the best subtask models irrespective of the value of $\hbar$) is exactly the same as that of version 3. Although the substitution of subtask models is of no use here, we shall see that it pays off later.

6.5.2 Corpus 'A+B'

Finally, the systems selected using corpus 'A' are used in conjunction with corpus 'A+B'. Figure 6.26 reveals a situation akin to, but not as bad as, that of §6.4.4. Compared with Figure 6.25, it shows a large drop in cross-entropy for version 1, while the best of the version 3 cross-entropies reduces only a little. Although version 3 is still convincingly better than version 1, the relative change in cross-entropies tends to corroborate the earlier finding that version 1 creates more general models, better able to scale up to larger corpora which may deviate somewhat from the characteristics of the original

[5] In theory but not yet in practice, since not all viewpoints in the version 1 subtask models are implemented in version 3.

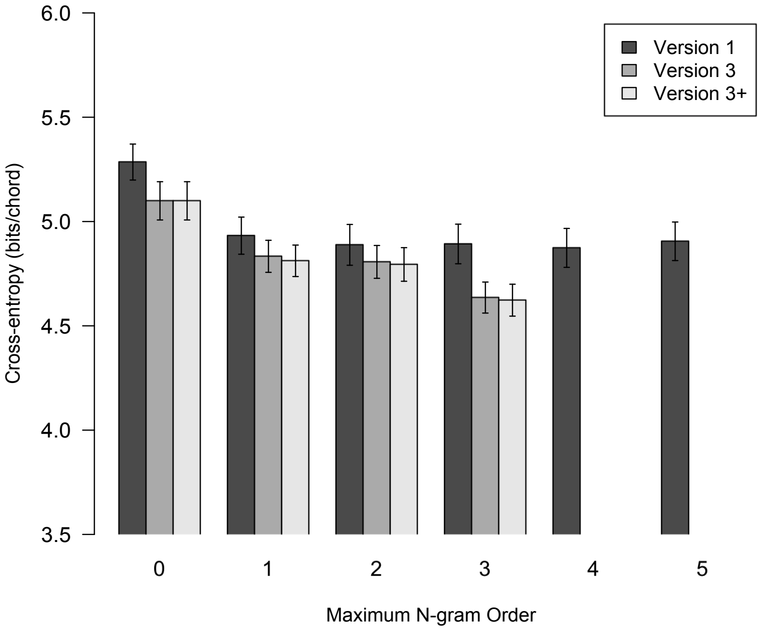


Figure 6.26: Bar chart showing how cross-entropy varies with $\hbar$ for the separate prediction of `Duration`, `Cont` and `Pitch` in the alto, tenor and bass given soprano using the augmented `Pitch` domain and corpus 'A+B' with systems selected using corpus 'A', comparing versions 1, 3 and 3+ (a hybrid of versions 1 and 3).

corpus. Version 3, like version 2 (although in this case for slightly different reasons), is capable of constructing models which are more specific to the corpus for which they are selected. The optimal version 3 model (using the best subtask models irrespective of the value of $\hbar$) has a cross-entropy of 4.63 bits/chord, which is 0.24 bits/chord lower than version 1.

As one might expect, the construction of version 3+ models results in a very slight improvement over version 3. We can see from Figure 6.26, however, that it is by no means clear that the improvement is real. The optimal version 3+ model (using the best subtask models irrespective of the value of $\hbar$) has a cross-entropy of 4.62 bits/chord, which is 0.01 bits/chord lower than version 3. See Table 6.3 for a brief summary of subsystem parameters and cross-entropies for the version 3 model. Replacement version 1 subsystems for version 3+ can be found by inspection of Table 6.1.

6.6 Comparison of Version 2 with Version 3

In this section we investigate the version 3 prediction of bass given soprano followed by alto/tenor given soprano/bass, since this is directly comparable with version 2.

predicting	$\hbar$	bias	L-S bias	x-entropy
`Duration`	3	3.2	2.1	0.62
`Cont`	2	1.2	28.0	0.72
`Pitch`	3	1.5	85.7	3.30

Table 6.3: Summary of subsystem parameters and cross-entropies for the best version 3 model (prediction of `Duration`, `Cont` and `Pitch` in alto/tenor/bass given soprano using separate multiple viewpoint systems selected in conjunction with the augmented `Pitch` domain and corpus 'A', but using corpus 'A+B'). *Cross-entropy* is abbreviated to *x-entropy.*

6.6.1 Combining Systems Predicting `Duration`, `Cont` and `Pitch` Separately

Please note that comparisons for the individual prediction of `Duration`, `Cont` and `Pitch` can be found in Appendix C (§C.2).

Figure 6.27 shows that for the overall model, the performance of version 3 is far superior to that of version 2. The minimum version 3 cross-entropy at a single $\hbar$ value is 5.21 bits/chord at an $\hbar$ of 2, compared with 5.44 bits/chord ($\hbar = 1$) for version 2. Employing the best subtask models irrespective of the value of $\hbar$ reduces the version 3 cross-entropy to 5.07 bits/chord, and the version 2 cross-entropy to 5.35 bits/chord (0.28 bits/chord higher than version 3).

The assembling of version 3+ models results in quite a large improvement over version 3, as can be seen in Figure 6.27. The optimal version 3+ model (using the best subtask models irrespective of the value of $\hbar$) has a cross-entropy of 4.92 bits/chord, which is 0.15 bits/chord lower than version 3.

6.6.2 Corpus 'A+B'

Finally, the systems selected using corpus 'A' are used in conjunction with corpus 'A+B'. In Figure 6.28, version 2 shows a small improvement when compared with Figure 6.27. On the other hand, the version 3 cross-entropy is a little worse overall. Although version 3 is still much better than version 2, a large enough corpus could possibly reverse this situation. The optimal version 3 model (using the best subtask models irrespective of the value of $\hbar$) has a cross-entropy of 5.12 bits/chord, which is 0.20 bits/chord lower than version 2.

We can see from Figure 6.28 that the construction of version 3+ models results in a small improvement over version 3. The optimal version 3+ model (using the best subtask models irrespective of the value of $\hbar$) has a cross-entropy of 5.05 bits/chord, which is 0.07 bits/chord lower than version 3. See Table 6.4 for a brief summary of subsystem parameters and cross-entropies for the version 3 model. Replacement version 2 subsystems for version 3+ can be found by inspection of Table 6.2.

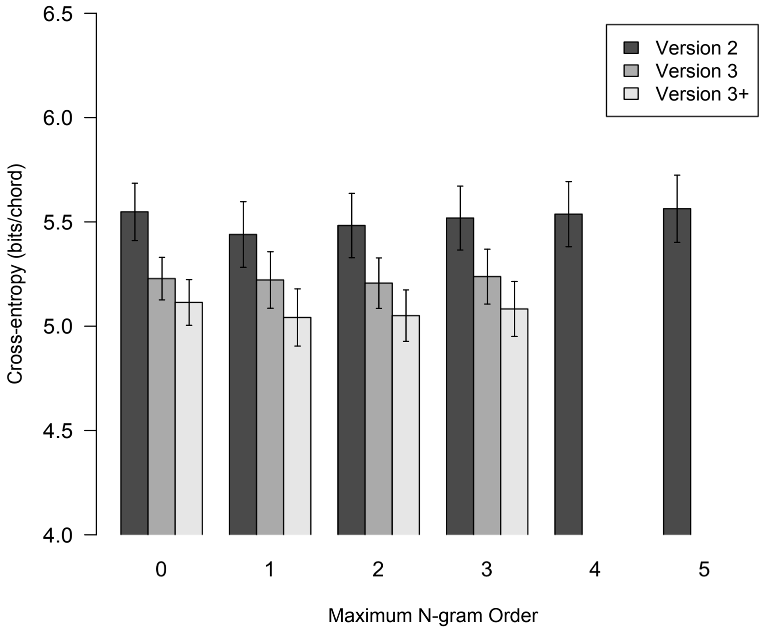


Figure 6.27: Bar chart showing how cross-entropy varies with $\hbar$ for the separate prediction of `Duration`, `Cont` and `Pitch` in the bass given soprano followed by alto/tenor given soprano/bass using the augmented `Pitch` domain, comparing versions 2, 3 and 3+ (a hybrid of versions 2 and 3).

predicting	stage	$\hbar$	bias	L-S bias	x-entropy
`Duration`	B given S	0	48.5	1.1	0.62
`Duration`	AT given SB	0	140	1.8	0.43
`Cont`	B given S	1	1.3	14.6	0.27
`Cont`	AT given SB	2	1.2	100	0.45
`Pitch`	B given S	3	1.9	20.5	1.67
`Pitch`	AT given SB	1	1.4	68.9	1.68

Table 6.4: Summary of subsystem parameters and cross-entropies for the best version 3 model (prediction of `Duration`, `Cont` and `Pitch` in bass given soprano followed by alto/tenor given soprano/bass using separate multiple viewpoint systems selected in conjunction with the augmented `Pitch` domain and corpus ‘A’, but using corpus ‘A+B’). *Cross-entropy* is abbreviated to *x-entropy.*

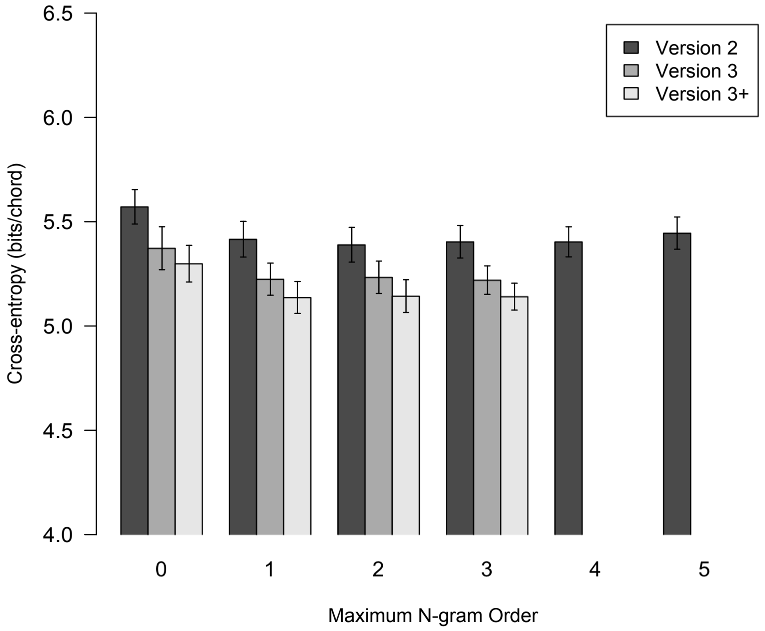


Figure 6.28: Bar chart showing how cross-entropy varies with $\hbar$ for the separate prediction of `Duration`, `Cont` and `Pitch` in the bass given soprano followed by alto/tenor given soprano/bass using the augmented `Pitch` domain and corpus ‘A+B’ with systems selected using corpus ‘A’, comparing versions 2, 3 and 3+ (a hybrid of versions 2 and 3).

version	corpus ‘A’	corpus ‘A+B’	difference
3.2+	3.90	4.01	+0.11
3.2	4.05	4.07	+0.02
3.1+	4.05	4.00	−0.05
3.1	4.05	4.01	−0.04
2	4.21	4.18	−0.03
1	4.40	4.18	−0.22

Table 6.5: Performance (cross-entropy) comparison of the best version 1 to 3 models predicting `Cont` and `Pitch`, lowest corpus ‘A’ cross-entropy first. Corpus ‘A+B’ cross-entropy and cross-entropy difference are also tabulated.

6.7 Final Overall Performance Comparisons

In this section we directly compare, as far as is possible, the best version 1 to 3 models. To avoid any possible confusion, the version 3 model which predicts alto, tenor and bass in a single stage (comparable with version 1) will here be termed version 3.1. Similarly, the version 3 two-stage prediction model, comparable with version 2, will be referred to as version 3.2. Such models employing better performing version 1 and 2 subtask models then become versions 3.1+ and 3.2+ respectively.

6.7.1 Prediction of `Cont` and `Pitch` Separately

By removing `Duration` from consideration, it is possible to directly compare version 1, version 2 and all of the various version 3 models, as shown in Table 6.5. The best models from each version or sub-version are shown in order of corpus ‘A’ cross-entropy, lowest first. On this basis, the best performing model overall, by a fair margin, is version 3.2+, with a cross-entropy of 3.90 bits/chord. Notice that all of the version 3 sub-versions are better than versions 1 and 2. The version 1 model performs least well, having a cross-entropy 0.50 bits/chord higher than that of version 3.2+.

The picture changes somewhat when precisely the same systems are used in conjunction with corpus ‘A+B’, however. This time, with version 3.2+ having suffered the largest rise in cross-entropy, version 3.1+ performs best by a very small margin over versions 3.1 and 3.2+. Versions 1 and 2 are still the worst performing; but version 1 has benefitted from the most dramatic increase in performance. Assuming that this trend continues, a further modest increase in corpus size would see version 1 having the lowest overall cross-entropy. It is expected that for viewpoint selection using larger corpora, however, version 3.2+ would remain preeminent.

6.7.2 Prediction of `Duration`, `Cont` and `Pitch` Separately

On reinstating the prediction of `Duration`, we find that essentially nothing changes. In Tables 6.6 and 6.7 the models are in the same order as in the `Cont` and `Pitch` prediction

version	corpus ‘A’	corpus ‘A+B’	difference
3.1+	4.65	4.62	−0.03
3.1	4.65	4.63	−0.02
1	5.08	4.87	−0.21

Table 6.6: Performance (cross-entropy) comparison of the best version 1 and 3 models predicting Duration, Cont and Pitch, lowest corpus ‘A’ cross-entropy first. Corpus ‘A+B’ cross-entropy and cross-entropy difference are also tabulated.

version	corpus ‘A’	corpus ‘A+B’	difference
3.2+	4.92	5.05	+0.13
3.2	5.07	5.12	+0.05
2	5.35	5.32	−0.03

Table 6.7: Performance (cross-entropy) comparison of the best version 2 and 3 models predicting Duration, Cont and Pitch, lowest corpus ‘A’ cross-entropy first. Corpus ‘A+B’ cross-entropy and cross-entropy difference are also tabulated.

case. The cross-entropy differences are very similar to before.

6.8 Comparison of Selected Multiple Viewpoint Systems

In this section, the best version 1 to 3 (and where applicable, version 0) multiple viewpoint systems selected using corpus ‘A’ for the prediction of Duration, Cont and Pitch separately are compared. The fact that there are systems which predict soprano from scratch, bass given soprano, alto/tenor given soprano/bass, and alto/tenor/bass given soprano means that certain viewpoints must be considered equivalent for the purposes of any meaningful comparison; for example, $(\texttt{Pitch})_S$, $(\texttt{Pitch})_{SB}$, $(\texttt{Pitch})_{AT}$, $(\texttt{Pitch})_{SAT}$, $(\texttt{Pitch})_{ATB}$ and $(\texttt{Pitch})_{SATB}$ are here deemed to be equivalent.

6.8.1 Prediction of Duration

A comparison of the best version 0, 1, 2 and 3 Duration-predicting multiple viewpoint systems is shown in Table 6.8. There is a huge amount of overlap between the version 0, 1 and 2 systems: indeed, the version 1 system and the version 2 bass-predicting system are exactly the same. Eight of the ten viewpoints in these systems also appear in the alto/tenor-predicting version 2 system, while six of them occur in the version 0 system. In fact, these six viewpoints are found in all four systems. It is interesting to note that only one of these six contains a primitive viewpoint derived from Pitch; in fact, only six of the fourteen tabulated viewpoints covering versions 0 to 2 have anything to do with Pitch, while none of them contain Cont.

Moving on to the comparison of the best version 3 Duration-predicting multiple viewpoint systems, we find that the alto/tenor/bass- and bass-predicting systems are

	0	1	2		3		
Viewpoint	S	ATB	B	AT	ATB	B	AT
$(\texttt{DurRatio} \otimes \texttt{Phrase})_{SATB/SB/S}$	×	×	×	×			
$(\texttt{Duration} \otimes \texttt{PositionInBar})_{SATB/SB/S/AT}$	×	×	×	×			×
$(\texttt{Duration} \otimes \texttt{LastInPhrase})_{SATB/SB/S}$	×	×	×	×			
$(\texttt{DurRatio} \otimes (\texttt{IOI} \ominus \texttt{FirstInBar}))_{SATB/SB}$		×	×				
$(\texttt{DurRatio} \otimes \texttt{TactusPositionInBar})_{SATB/SB/S}$	×	×	×	×			
$(\texttt{DurRatio} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{SATB/SB}$		×	×	×			
$(\texttt{Duration} \otimes \texttt{Metre})_{SATB/SB/S/AT}$	×	×	×	×			×
$(\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{SATB/SB/S}$	×	×	×	×			
$(\texttt{Duration} \otimes (\texttt{IOI} \ominus \texttt{FirstInBar}))_{SATB/SB}$		×	×				
$(\texttt{DurRatio} \otimes (\texttt{Interval} \ominus \texttt{FirstInBar}))_{SATB/SB}$		×	×	×			
$(\texttt{DurRatio} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInBar}))_{SATB}$				×			
$(\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInBar}))_{SATB}$				×			
$(\texttt{Duration} \otimes (\texttt{Interval} \ominus \texttt{FirstInBar}))_{SATB}$				×			
$(\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInPhrase}))_{S}$	×						
$(\texttt{TactusPositionInBar})_{S} \otimes (\texttt{DurRatio} \otimes \texttt{LastInPhrase})_{ATB/B}$					×	×	
$(\texttt{PositionInBar} \otimes \texttt{LastInPhrase})_{S} \otimes (\texttt{Duration} \otimes \texttt{Metre})_{ATB/B}$					×	×	
$(\texttt{Cont} \otimes \texttt{LastInPhrase})_{S} \otimes (\texttt{Duration} \otimes \texttt{Metre})_{ATB/B}$					×	×	
$(\texttt{TactusPositionInBar} \otimes \texttt{FirstInPhrase})_{S} \otimes (\texttt{DurRatio} \otimes \texttt{LastInPhrase})_{ATB/B}$					×	×	
$(\texttt{LastInPhrase} \otimes \texttt{FirstInPiece})_{S} \otimes (\texttt{Duration} \otimes \texttt{Metre})_{ATB/B}$					×	×	
$(\texttt{DurRatio} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Phrase})_{B}$							×
$(\texttt{DurRatio} \otimes \texttt{FirstInBar})_{AT} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_{B}$							×
$(\texttt{DurRatio} \otimes \texttt{FirstInBar})_{AT} \otimes (\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{B}$							×
$(\texttt{Cont})_{S} \otimes (\texttt{DurRatio} \otimes \texttt{FirstInBar})_{AT} \otimes (\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{B}$							×
$(\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_{S} \otimes (\texttt{Duration} \otimes \texttt{PositionInBar})_{AT}$							×
$((\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}) \otimes \texttt{TactusPositionInBar})_{S} \otimes (\texttt{Duration} \otimes \texttt{PositionInBar})_{AT}$							×
$(\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_{S} \otimes (\texttt{DurRatio} \otimes \texttt{TactusPositionInBar})_{AT}$							×
$((\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}) \otimes \texttt{FirstInBar})_{S} \otimes (\texttt{DurRatio} \otimes \texttt{TactusPositionInBar})_{AT}$							×
$(\texttt{Cont} \otimes \texttt{FirstInPiece})_{S} \otimes (\texttt{DurRatio} \otimes \texttt{FirstInBar})_{AT} \otimes (\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{B}$							×
$((\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}) \otimes \texttt{TactusPositionInBar})_{S} \otimes (\texttt{Duration} \otimes \texttt{PositionInBar})_{AT} \otimes (\texttt{FirstInBar})_{B}$							×
$(\texttt{Cont})_{S} \otimes (\texttt{Duration})_{AT} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_{B}$							×
$(\texttt{Cont})_{S} \otimes (\texttt{DurRatio})_{AT} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_{B}$							×
$(\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_{S} \otimes (\texttt{Duration} \otimes \texttt{Metre})_{AT}$							×

Table 6.8: Best version 0 (S), version 1 (ATB given S), version 2 (B given S and AT given SB) and version 3 (ATB given S, B given S and AT given SB) multiple viewpoint systems selected using corpus ‘A’ for the prediction of `Duration`.

exactly the same, which mirrors the finding for versions 1 and 2. On the other hand, the alto/tenor-predicting system has no viewpoints in common with the other version 3 systems (and only two in common with versions 0 to 2). This is unsurprising given the added flexibility of the bass layer becoming independent of the others with respect to viewpoint assignment, resulting in some complex inter-layer linked viewpoints. A larger proportion of version 3 viewpoints, twelve out of a total of twenty, contain primitives derived from `Pitch`. This is to be expected, considering that the number of primitive viewpoints in an inter-layer linked viewpoint is no longer limited to two. Perhaps surprisingly, however, in view of the additional flexibility, `Cont` appears in only four of these viewpoints. This seems to suggest that there is little correlation between `Duration` and `Cont`.

6.8.2 Prediction of `Cont`

The best version 1, 2 and 3 `Cont`-predicting multiple viewpoint systems are compared in Table 6.9 (`Cont` is not present in version 0). There is, again, a good deal of overlap between the three version 1 and version 2 systems, which are far more parsimonious than the corresponding `Duration`-predicting systems. Indeed, the version 2 bass-predicting system comprises only three viewpoints. This is probably largely due to the fact that there are no primitive viewpoints derived from `Cont`, whereas `Duration` or `DurRatio` may occur in `Duration`-predicting viewpoints.

Only one of the version 3 viewpoints (in the bass-predicting system) also occurs in version 1 and 2 systems; and there is almost no overlap between the version 3 systems (only one viewpoint appears twice), which is very different from the `Duration`-predicting case. It would appear that there are differences between the parts with respect to `Cont` which version 3 is able to exploit to construct systems more specialised for the task at hand. Sixteen out of a total of twenty-seven tabulated viewpoints contain primitives derived from `Pitch`, whereas only four contain either `Duration` or `DurRatio`. This corroborates the finding in §6.8.1 that there is little correlation between `Duration` and `Cont` (at least in the relatively small corpus ‘A’).

6.8.3 Prediction of `Pitch`

The best version 0, 1 and 2 `Pitch`-predicting multiple viewpoint systems are shown in Table 6.10 along with partial version 3 systems for purposes of comparison (the latter systems are completed in Table 6.11). There is proportionately less overlap between version 0 to 2 systems than we have seen with respect to `Duration` and `Cont`. The plethora of primitive viewpoints derived from `Pitch` seems to enable the construction of more specialist systems for the prediction of `Pitch` even within the confines of these less flexible versions. Again, there is very little overlap between version 3 and other systems. There are two viewpoints, however, which occur in four out of the six systems to which they could possibly belong: `Cont` ⊗ `ScaleDegree` and `Cont` ⊗ `Interval`. Although only

	1	2		3		
Viewpoint	ATB	B	AT	ATB	B	AT
$(\texttt{Cont} \otimes \texttt{Interval})_{SATB}$	×		×			
$(\texttt{Cont} \otimes \texttt{Metre})_{SATB/SB}$	×	×	×			
$(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{SATB/SB}$	×	×	×			
$(\texttt{Cont} \otimes (\texttt{Contour} \ominus \texttt{Tactus}))_{SATB}$	×					
$(\texttt{Cont} \otimes \texttt{Tactus})_{SATB}$	×					
$(\texttt{Cont} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInPhrase}))_{SATB/SB/B}$	×	×			×	
$(\texttt{Cont} \otimes \texttt{PositionInBar})_{SATB/B}$			×			
$(\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$				×		
$(\texttt{Cont} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$				×		
$(\texttt{Duration} \otimes \texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$				×		
$(\texttt{ScaleDegree} \ominus \texttt{FirstInPhrase})_S \otimes (\texttt{Cont})_{ATB/AT}$				×		×
$(\texttt{Cont} \otimes \texttt{Metre})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$				×		
$(\texttt{Cont} \otimes \texttt{Interval})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$				×		
$(\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$				×		
$(\texttt{Interval} \otimes \texttt{ScaleDegree})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$				×		
$(\texttt{Cont} \otimes \texttt{ScaleDegree})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$				×		
$(\texttt{ScaleDegree} \ominus \texttt{FirstInPhrase})_S \otimes (\texttt{Cont} \otimes \texttt{LastInPiece})_{ATB}$				×		
$(\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{Metre})_B$					×	
$(\texttt{DurRatio} \otimes \texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{Metre})_B$					×	
$(\texttt{Cont} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_S \otimes (\texttt{Cont} \otimes \texttt{Metre})_B$					×	
$(\texttt{ScaleDegree} \otimes \texttt{FirstInPhrase})_S \otimes (\texttt{Cont} \otimes \texttt{Metre})_B$					×	
$(\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Cont})_B$						×
$(\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Cont} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_B$						×
$(\texttt{Cont} \otimes \texttt{ScaleDegree})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Cont})_B$						×
$(\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Duration} \otimes \texttt{Cont})_B$						×
$(\texttt{DurRatio} \otimes \texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Cont} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_B$						×
$(\texttt{Cont} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Cont})_B$						×

Table 6.9: Best version 1 (ATB given S), version 2 (B given S and AT given SB) and version 3 (ATB given S, B given S and AT given SB) multiple viewpoint systems selected using corpus ‘A’ for the prediction of `Cont`.

three out of twenty-two relevant viewpoints in Table 6.10 contain Cont, it is nevertheless clearly important to the prediction of Pitch. On the other hand, only one system contains a single viewpoint containing Duration. This strongly suggests that Duration is much less important for the prediction of Pitch than *vice versa*. This again leads to the conclusion that changing the attribute prediction order such that Pitch is predicted before Duration could well prove beneficial.

Looking now at Tables 6.10 and 6.11 for a complete picture of the version 3 systems, we find that, as with Cont, there is hardly any overlap between the systems. What is particularly interesting is that the more flexible linking of this version has enabled the appearance of Cont in twelve out of twenty-eight viewpoints, which is a huge increase over versions 1 and 2. Conversely, Duration now completely disappears, emphasising the potential benefit of investigating alternative attribute prediction orders in future work.

6.9 Conclusion

The version 1 comparisons in §6.2 demonstrate that, as expected, the use of the augmented Pitch domain results in far higher cross-entropies than those produced by the seen domain, especially for the prediction of Pitch. Since this is not a like for like comparison, the higher cross-entropies are not necessarily an indication of worse performance; indeed, the larger domain, being more representative of a larger (though hypothetical) corpus, produces more realistic probabilities. Performance is enhanced by using separately selected multiple viewpoint systems to predict individual basic attributes rather than predicting them together using a single system. The effect is more pronounced for the augmented Pitch domain. Using the best systems selected using corpus ‘A’ in conjunction with corpus ‘A+B’ results in a large improvement in prediction performance.

The first set of version 2 viewpoint selection runs in §6.3, for attribute prediction together using the seen Pitch domain, compare different combinations of two-stage prediction. By far the best performance is obtained by predicting the bass part first followed by the inner parts together, reflecting the usual human approach to harmonisation. It is interesting to note that this heuristic, almost universally followed during harmonisation, therefore has an information theoretic explanation for its success. Generally, other comparisons produce results similar to those obtained for version 1; exceptionally, however, the use of the larger corpus ‘A+B’ leads to a much smaller reduction in cross-entropy.

From this point on, only separate attribute prediction using the augmented Pitch domain is investigated. In comparing version 1 with version 2 (see §6.4), only Cont and Pitch are taken into consideration, since the prediction of Duration is not directly comparable. On this basis, version 2 is better than version 1 when using corpus ‘A’; but when corpus ‘A+B’ is used, their performance is identical. Similarly, §6.5 shows that version 3 (predicting alto/tenor/bass given soprano) performs much better than version

	0	1	2		3		
Viewpoint	S	ATB	B	AT	ATB	B	AT
$(\texttt{Cont} \otimes \texttt{ScaleDegree})_{SATB/SB/AT}$		×	×	×			×
$(\texttt{Cont} \otimes \texttt{Interval})_{SATB/ATB/AT}$		×		×	×		×
$(\texttt{ScaleDegree} \otimes \texttt{LastInPhrase})_{SATB}$		×					
$(\texttt{InScale} \otimes \texttt{Tessitura})_{SATB}$		×		×			
$((\texttt{Pitch} \ominus \texttt{Tactus}) \otimes \texttt{InScale})_{SATB/SB}$		×	×				
$((\texttt{ScaleDegree} \ominus \texttt{FirstInPhrase}) \otimes \texttt{FirstInPiece})_{SATB/SB/S}$	×	×	×				
$(\texttt{Interval} \otimes \texttt{InScale})_{SATB/SB}$		×	×				
$(\texttt{Cont} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{SATB/SB}$		×	×				
$((\texttt{Contour} \ominus \texttt{Tactus}) \otimes \texttt{InScale})_{SATB}$		×					
$((\texttt{ScaleDegree} \ominus \texttt{Tactus}) \otimes \texttt{Piece})_{SATB}$		×					
$((\texttt{ScaleDegree} \ominus \texttt{Tactus}) \otimes \texttt{Metre})_{SATB}$		×					
$((\texttt{ScaleDegree} \ominus \texttt{Tactus}) \otimes \texttt{FirstInPiece})_{SATB}$		×					
$(\texttt{Interval} \otimes \texttt{ScaleDegree})_{SB}$			×				
$(\texttt{Interval} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_{SB}$			×			×	
$(\texttt{Interval} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInBar}))_{SB/S}$	×		×				
$(\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{SB}$			×				
$((\texttt{ScaleDegree} \ominus \texttt{FirstInPhrase}) \otimes \texttt{Tessitura})_{SATB/SB/S}$	×		×	×			
$(\texttt{Interval} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{SB/S}$	×		×				
$((\texttt{ScaleDegree} \ominus \texttt{FirstInPhrase}) \otimes \texttt{Piece})_{SB/S}$	×		×				
$(\texttt{Pitch})_{SATB/ATB/SB}$				×		×	×
$(\texttt{ScaleDegree} \ominus \texttt{Tactus})_{SATB}$				×			
$(\texttt{ScaleDegree} \otimes \texttt{FirstInPhrase})_{SATB}$				×			
$(\texttt{ScaleDegree} \otimes \texttt{Phrase})_{S}$	×						
$(\texttt{ScaleDegree} \otimes \texttt{Metre})_{S}$	×						
$(\texttt{IntFirstInBar} \otimes \texttt{ScaleDegree})_{S}$	×						
$(\texttt{ScaleDegree} \otimes \texttt{Tessitura})_{S}$	×						
$(\texttt{Interval} \otimes \texttt{TactusPositionInBar})_{S}$	×						
$(\texttt{IntFirstInPhrase} \otimes \texttt{ScaleDegree})_{S}$	×						
$(\texttt{Pitch} \otimes \texttt{ScaleDegree})_{S}$	×						
$(\texttt{IntFirstInPiece} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInPhrase}))_{S}$	×						
$(\texttt{Pitch} \otimes \texttt{FirstInPhrase})_{S}$	×						
$(\texttt{Interval} \otimes \texttt{Phrase})_{S}$	×						
$(\texttt{IntFirstInPiece} \otimes \texttt{ScaleDegree})_{S}$	×						
$(\texttt{Interval} \otimes \texttt{FirstInBar})_{S}$	×						

Table 6.10: Best version 0 (S), version 1 (ATB given S) and version 2 (B given S and AT given SB) multiple viewpoint systems selected using corpus ‘A’ for the prediction of `Pitch`. Partial version 3 (ATB given S, B given S and AT given SB) systems are also shown for purposes of comparison. The latter systems are completed in Table 6.11.

Viewpoint	ATB	B	AT
(ScaleDegree ⊗ Phrase)$_S$ ⊗ (Cont ⊗ ScaleDegree)$_{ATB}$	×		
(ScaleDegree ⊗ Piece)$_{ATB}$	×		
(ScaleDegree)$_S$ ⊗ (Interval ⊗ ScaleDegree)$_{ATB/B}$	×	×	
(FirstInPhrase)$_S$ ⊗ (Cont ⊗ Interval)$_{ATB}$	×		
(Cont ⊗ ScaleDegree)$_S$ ⊗ (ScaleDegree)$_{ATB}$	×		×
(ScaleDegree ⊖ FirstInPhrase)$_S$ ⊗ (ScaleDegree ⊗ Piece)$_{ATB}$	×		
(ScaleDegree ⊖ LastInPhrase)$_S$ ⊗ (ScaleDegree)$_{ATB/B}$	×	×	
(ScaleDegree ⊗ Metre)$_S$ ⊗ (ScaleDegree ⊗ Piece)$_B$		×	
(Cont ⊗ ScaleDegree)$_S$ ⊗ (Interval ⊗ ScaleDegree)$_B$		×	
(Phrase)$_S$ ⊗ (Interval ⊗ (ScaleDegree ⊖ Tactus))$_B$		×	
((ScaleDegree ⊖ Tactus) ⊗ Phrase)$_S$ ⊗ (Interval ⊗ (ScaleDegree ⊖ Tactus))$_B$		×	
(ScaleDegree ⊗ FirstInBar)$_S$ ⊗ (ScaleDegree)$_B$		×	
(ScaleDegree ⊖ LastInPhrase)$_S$ ⊗ (ScaleDegree ⊗ FirstInBar)$_B$		×	
(Cont ⊗ Phrase)$_S$ ⊗ (Cont ⊗ Interval)$_B$		×	
(ScaleDegree ⊖ LastInPhrase)$_S$ ⊗ (ScaleDegree ⊗ LastInPiece)$_B$		×	
(ScaleDegree)$_S$ ⊗ (Cont ⊗ ScaleDegree)$_{AT}$ ⊗ (ScaleDegree)$_B$			×
(Interval)$_S$ ⊗ (Cont ⊗ Interval)$_{AT}$			×
(ScaleDegree ⊖ FirstInPhrase)$_S$ ⊗ (ScaleDegree)$_{ATB}$			×
(Interval ⊗ ScaleDegree)$_{AT}$ ⊗ (ScaleDegree)$_B$			×
(Cont ⊗ Interval)$_{AT}$ ⊗ (Cont)$_B$			×
(ScaleDegree)$_{SAT}$ ⊗ (Cont ⊗ ScaleDegree)$_B$			×
(ScaleDegree ⊖ LastInPhrase)$_S$ ⊗ (Interval ⊗ ScaleDegree)$_{AT}$ ⊗ (ScaleDegree)$_B$			×
(Pitch ⊗ (ScaleDegree ⊖ FirstInPhrase))$_S$ ⊗ (ScaleDegree)$_{ATB}$			×
(ScaleDegree)$_S$ ⊗ (Cont ⊗ ScaleDegree)$_{ATB}$			×

Table 6.11: Best version 3 (ATB given S, B given S and AT given SB) multiple viewpoint systems selected using corpus ‘A’ for the prediction of Pitch. Note that a few viewpoints are instead shown in Table 6.10 for purposes of comparison.

1 when using corpus ‘A’; but the version 3 performance advantage is greatly reduced on changing to corpus ‘A+B’. We can infer from this that version 1 creates more general models, better able to scale up to larger corpora which may deviate somewhat from the characteristics of the original corpus.

We find in §6.6 that the performance of version 3 (predicting bass given soprano followed by alto/tenor given soprano/bass) is better than that of version 2, although the margin is reduced on using corpus ‘A+B’. Overall models with a superior performance can be constructed by combining the better of the version 2 and 3 subtask models (version 3.2+).

On removing Duration from consideration in order to directly compare version 1, version 2 and all of the various version 3 models, we find that for corpus ‘A’ all of the version 3 sub-versions outperform versions 1 and 2 (see §6.7). Version 3.2+ (version 3+ predicting bass followed by alto/tenor) performs best overall, having a cross-entropy 0.50 bits/chord lower than the worst-performing version 1. The use of corpus ‘A+B’ changes things considerably. Version 1 benefits from the biggest improvement, while version 3.2+ suffers the largest deterioration in performance: although version 3.2+ is still better, the margin has been reduced to 0.17 bits/chord. A further modest increase in corpus size could see version 1 having the lowest overall cross-entropy, although version 3.2+ is likely to remain preeminent for viewpoint selection using larger corpora. On reinstating the prediction of `Duration` and making comparisons to the fullest possible extent, we find nothing which alters these conclusions.

A comparison in §6.8 of the best version 0 to 3 multiple viewpoint systems for the prediction of `Duration`, `Cont` and `Pitch` separately leads to three main conclusions. Firstly, there appears to be little correlation between `Duration` and `Cont`, as evidenced by the scarcity of the primitives `Duration` and `DurRatio` in `Cont`-predicting systems and the dearth of the primitive `Cont` in `Duration`-predicting systems. Secondly, primitives derived from `Pitch` are heavily involved in `Duration`-predicting systems, whereas `Duration` and `DurRatio` are almost absent from `Pitch`-predicting systems. This suggests that a change in the basic attribute prediction order, such that `Pitch` is predicted before `Duration`, could be beneficial. This will be investigated in future work. Finally, since viewpoints `IOI` $\ominus$ `FirstInBar`, `Interval` $\ominus$ `FirstInBar`, `ScaleDegree` $\ominus$ `FirstInBar`, `Contour` $\ominus$ `Tactus`, `Pitch` $\ominus$ `Tactus`, `InScale` and `Tessitura` appear in the best version 1 and 2 systems, it is likely that the version 3 models can be improved by adding these viewpoints to the restricted version 3 pool.

Chapter 7

Analysis of Selected Version 0 Viewpoints

7.1 Introduction

A much larger number of linked viewpoints are available in this research compared with previous comparable work (Conklin and Witten, 1995; Pearce, 2005), where attempts were made to predict which linked viewpoints would perform well from a music theoretic point of view, and only those viewpoints were implemented. We have already established that many viewpoints new to this research have been selected for the best performing multiple viewpoint systems. It would be interesting and instructive to examine, from a music theoretic standpoint, such viewpoints which perform particularly well. This analysis could point the way to fruitful areas of further research. Please note that scores of the melodies (and their harmonisations) referred to in this chapter are included in Appendix D for convenient reference.

It is important to note at this point that software implementation errors affecting `IOI` $\ominus$ `FirstInBar` and `IOI` $\ominus$ `Tactus` were discovered at a very late stage. In each case, models are constructed using the threaded `IOI` value, whereas prediction probability distributions are calculated using the local `IOI` value. Bearing in mind that the errors affect versions 0, 1 and 2 (`IOI` is not implemented in version 3) and that within these versions the problem is not widespread, we can reasonably assume that the performance comparisons in Chapters 5 and 6 are still on an even footing. The software will be corrected as a matter of urgency in future work.

In this chapter we look at version 0 (melodic) multiple viewpoint systems and speculate on why certain viewpoints have been selected. In particular, we examine differences between systems selected for LTM, LTM+, STM, BOTH and BOTH+ for the prediction of `Duration` and `Pitch` together.[1] The first five viewpoints to be selected for each multiple viewpoint system (using corpus 'A' unless otherwise stated) are discussed in

[1] Viewpoints performing well in this prediction paradigm also perform well in systems selected to predict `Duration` and `Pitch` separately; therefore separate prediction is not included.

Multiple viewpoint system	Cross-entropy (bits/note)
{Duration, Pitch}	4.37
+ ScaleDegree ⊗ Tessitura	3.88
+ Duration ⊗ Metre	3.67
+ Interval ⊗ LastInPhrase	3.56
− Pitch	3.52
+ DurRatio ⊗ Phrase	3.43
− Duration	3.40
+ ScaleDegree ⊗ Piece	3.34

Table 7.1: Cross-entropies up to and including the fifth round of viewpoint addition/deletion during viewpoint selection of the best version 0 multiple viewpoint system (predicting Duration and Pitch) for LTM using corpus ‘A’.

detail on their first appearance. LTM and LTM+ are dealt with in §7.2 and §7.3 respectively, while STM is considered in §7.4 and §7.5 (arithmetic and geometric combination respectively). BOTH is investigated in §7.6, followed by BOTH+ in §7.7, §7.8 and §7.9 (corpora ‘A’, ‘B’ and ‘A+B’ respectively). Finally, in §7.10, the chapter is summarised and a conclusion given.

7.2 LTM

The first five viewpoints selected for the best LTM multiple viewpoint system are presented in order of selection in Table 7.1, along with the deletions of Pitch and Duration. Cross-entropies are shown at each stage. A brief examination of the extent to which there may be mutual information between different viewpoint models is presented in Appendix E.

7.2.1 ScaleDegree ⊗ Tessitura

The first viewpoint to be selected is ScaleDegree ⊗ Tessitura, which is also chosen (although at a slightly later stage) for the best LTM+ system. On its own, ScaleDegree is a good viewpoint: whereas the system {Duration, Pitch} gives rise to a cross-entropy of 4.37, the addition of ScaleDegree reduces it to 3.96 bits/note. The reason ScaleDegree performs so well is that Pitch contexts which are different solely because they appear in melodies with different keys become the same ScaleDegree context, leading to a better prediction probability distribution. There is a drawback to ScaleDegree, however, which is that scale degrees at different octaves are considered to be equivalent. This problem is apparent when predicting, for example, the seventh note of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33, which can be found in Appendix D). According to the ScaleDegree prediction probability distribution (see Figure 7.1), the seventh note is most likely to be the tonic, with a probability of 0.5290. There are two possible tonic notes, however: E♭4 and E♭5. Intuitively, we would consider E♭4

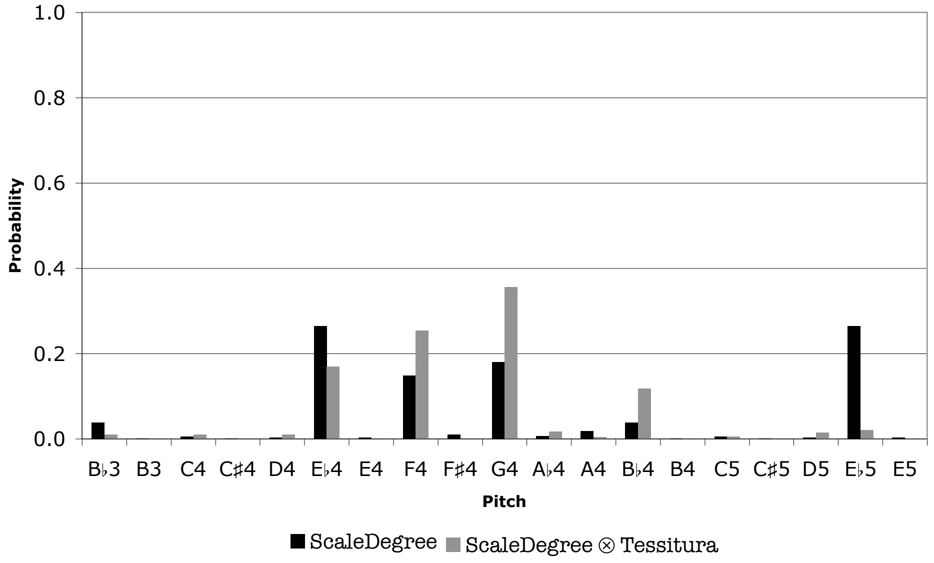


Figure 7.1: Bar chart showing LTM prediction probability distributions for viewpoints `ScaleDegree` and `ScaleDegree` ⊗ `Tessitura`, after conversion to distributions over pitches, predicting the pitch of the seventh note of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33).

to be by far the more likely continuation; but as these notes are indistinguishable by `ScaleDegree`, the probability must be split equally between the possibilities, giving a probability of 0.2645 for each.

The model could be improved by finding a means of distinguishing between octaves, such that more realistic probabilities could be assigned. This can be achieved to a greater or lesser extent by linking `ScaleDegree` with other viewpoints such as `IntFirstInPiece`, `Interval`, `Contour` or `Tessitura`. In this case the pairing is `ScaleDegree` ⊗ `Tessitura`, with {`Duration`, `Pitch`, `ScaleDegree` ⊗ `Tessitura`} reducing the cross-entropy to 3.88 bits/note. In fact, this linked viewpoint is able to completely distinguish between octaves. For the `Pitch` domain derived from corpus 'A', pitches 58 to 64 have a `Tessitura` value of -1; pitches 65 to 72 have a `Tessitura` value of 0; and pitches 73 to 76 have a `Tessitura` value of 1. Each of these three pitch ranges is less than an octave, thereby making all `ScaleDegree` ⊗ `Tessitura` values unique. Figure 7.1 also shows the distribution resulting from the `ScaleDegree` ⊗ `Tessitura` viewpoint model for comparison. As expected, with octaves distinguishable one from the other, E♭5 has a much lower probability than E♭4. What is perhaps surprising, however, is that the distributions are quite different; it is not just a case of redistributing probability mass between different octaves of the same scale degree. E♭4's `ScaleDegree` ⊗ `Tessitura` probability is much lower, while for F4, G4 and B♭4, the probabilities are much higher. In fact, it is G4 which has the highest probability according to the `ScaleDegree` ⊗ `Tessitura` distribution; and as it happens, the hymn tune does indeed continue with a G4.

Although `ScaleDegree` ⊗ `Tessitura` is good at distinguishing between octaves, this ability is only precise within one key. Statistical information does not transfer well between keys; for example, in the key of B♭ major `ScaleDegree` values of 3, 4, 5 and 6 are associated with a `Tessitura` value of 1, while in E major the `ScaleDegree` values are 9, 10, 11 and 0. A new viewpoint which is potentially even better than `ScaleDegree` ⊗ `Tessitura` can be envisaged. Like `ScaleDegree`, the values would be relative to the tonic; unlike `ScaleDegree`, however, this reference note would be fixed at some MIDI value lower than the lowest note in the domain, and all notes above it would have a unique value. The implementation and investigation of this viewpoint is for future work.

7.2.2 `Duration` ⊗ `Metre`

The second viewpoint to be selected is `Duration` ⊗ `Metre`, which is also chosen (at a slightly later stage) for LTM+. The performance of this viewpoint indicates a strong correlation between metrical importance and length of note, which we shall now investigate. Let us, for example, consider a hymn tune with three minims to the bar. Firstly, it is atypical of the corpus for notes to be tied across bar lines; therefore a note longer than a tactus beat is very unlikely to occur on the third beat of the bar. Secondly, syncopation is quite rare, which means that only occasionally would a long note occur on the second beat. Long notes are most likely to occur, therefore, on the first beat of the bar. This fundamental difference between the first beat and the other two beats is precisely reflected in the definition of `Metre` for a bar of three tactus beats (see §3.2.4.2). Similar arguments lead to the greater expectation of long notes on the first beat in all other time signatures, and also on the third of four tactus beats in a bar.

Figure 7.2 shows LTM prediction probability distributions for viewpoint `Duration` ⊗ `Metre`, predicting four note durations in the third bar of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33). All of these notes are minims in the hymnal, and the preceding two notes are also minims; therefore with an $\hbar$ of 2, the duration component of the context is the same in each case. The most striking thing about the distributions is that a minim is overwhelmingly most probable in each case. We see that a minim is less likely on the first and third beats than on the second and fourth, whereas semibreves are more likely to occur on the first and third beats. A dotted semibreve is most likely on the first beat (it is unlikely on the third beat because it would need to be tied over a bar line). This is all in line with expectations. Finally, a crotchet is a little more likely to occur on the fourth beat than any other.

It is conceivable that the definition of `Metre` could be improved; but the experimentation that this would entail will be left for future work.

7.2.3 `Interval` ⊗ `LastInPhrase`

The third viewpoint to be selected is `Interval` ⊗ `LastInPhrase`. This viewpoint is not chosen for LTM+; but a similar viewpoint, `Interval` ⊗ `Phrase`, is chosen at a slightly

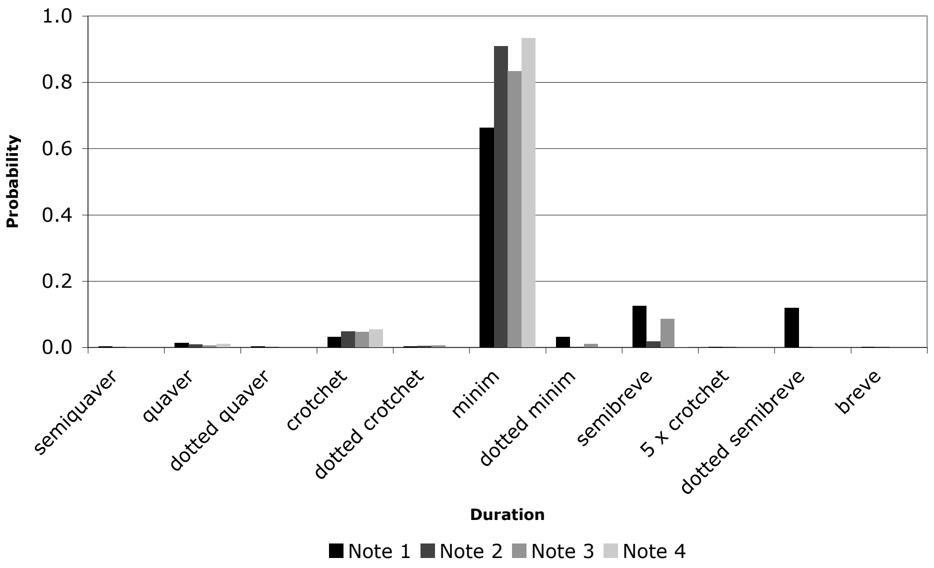


Figure 7.2: Bar chart showing LTM prediction probability distributions for viewpoint `Duration` ⊗ `Metre`, predicting four note durations in the third bar of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33).

later stage.[2] The reason `Interval` ⊗ `LastInPhrase` performs so well is that the last note in a phrase is almost always approached by step; that is, the interval between the penultimate note and the last note is usually either a tone or a semitone, ascending or descending. This means that when the value of `LastInPhrase` is T (indicating a phrase ending), `Interval` values from amongst 2, 1, -1 and -2 (depending on the context) will have relatively high probabilities.

Figure 7.3 shows LTM prediction probability distributions for viewpoints `Interval` and `Interval` ⊗ `LastInPhrase`, after conversion to distributions over pitches, predicting the pitch of the last note of the first phrase of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36). The penultimate note of the phrase is F4; therefore the highest probabilities in the `Interval` ⊗ `LastInPhrase` distribution are in the expected pitch range, E♭4 to G4. G4, which happens to be the last note of the phrase, has the highest probability, followed by E♭4. G4 is only a little less probable according to `Interval`; but E♭4 is far less probable. Notice also that repeating the F4 is an order of magnitude less likely according to `Interval` ⊗ `LastInPhrase`.

It should be realised that linking with `LastInPhrase` probably also improves prediction a little at positions other than last in phrase, due to the fact that last in phrase intervals are not taken into account in those statistics, other than as part of the context (which may particularly improve prediction at the beginning of phrases).

[2] `Interval` ⊗ `FirstInPhrase` is added to LTM at a late stage in the selection process; together with `Interval` ⊗ `LastInPhrase`, the information provided is very similar to that garnered by `Interval` ⊗ `Phrase`.

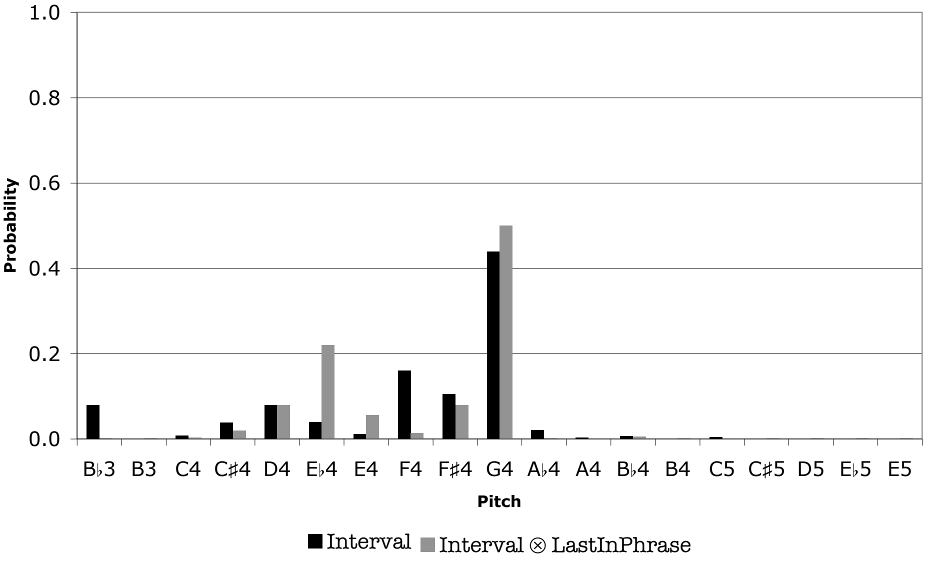


Figure 7.3: Bar chart showing LTM prediction probability distributions for viewpoints `Interval` and `Interval` ⊗ `LastInPhrase`, after conversion to distributions over pitches, predicting the pitch of the last note of the first phrase of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36).

7.2.4 `DurRatio` ⊗ `Phrase`

The fourth viewpoint to be chosen is `DurRatio` ⊗ `Phrase`, which is also selected (at a slightly later stage) for LTM+. The reason this viewpoint is so beneficial is that the last note of a phrase is often long. This means that when the value of `Phrase` is -1 (end of phrase), the value of `DurRatio` is likely to be > 1; and conversely, when the value of `Phrase` is 1 (beginning of phrase), the value of `DurRatio` is likely to be < 1.

Figures 7.4 and 7.5 show LTM prediction probability distributions for viewpoints `DurRatio` and `DurRatio` ⊗ `Phrase`, respectively predicting note durations at the end of the first and at the beginning of the second phrase of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37). The four notes immediately preceding the first note to be predicted (last in phrase) are crotchets. Whereas a crotchet is by far the most likely prediction according to the `DurRatio` distribution, `DurRatio` ⊗ `Phrase` indicates that a minim is a little more probable than a crotchet, as expected; also a dotted minim has a much higher probability in the `DurRatio` ⊗ `Phrase` distribution than in the `DurRatio` one. A dotted crotchet, however, is slightly less likely according to `DurRatio` ⊗ `Phrase`. The last note of the first phrase is a minim, which forms part of the context for the prediction of the next note. According to `DurRatio`, the first note of the next phrase is a little more likely to be a minim than a crotchet; however, a crotchet is much more probable than a minim from the point of view of `DurRatio` ⊗ `Phrase`. The melody actually continues with a dotted crotchet, which has a slightly higher probability in the `DurRatio` ⊗ `Phrase` distribution than in the `DurRatio` one.

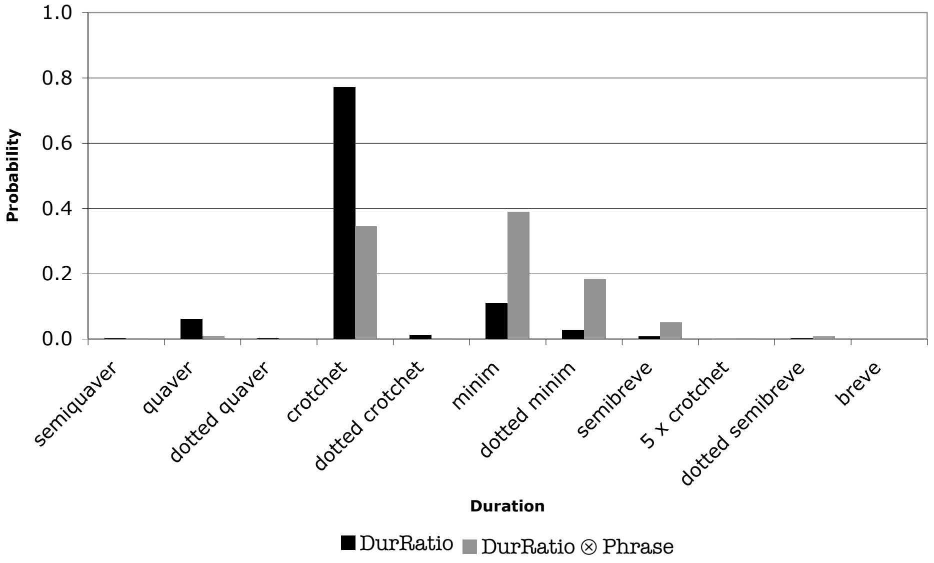


Figure 7.4: Bar chart showing LTM prediction probability distributions for viewpoints `DurRatio` and `DurRatio` ⊗ `Phrase`, predicting note durations at the end of the first phrase of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37).

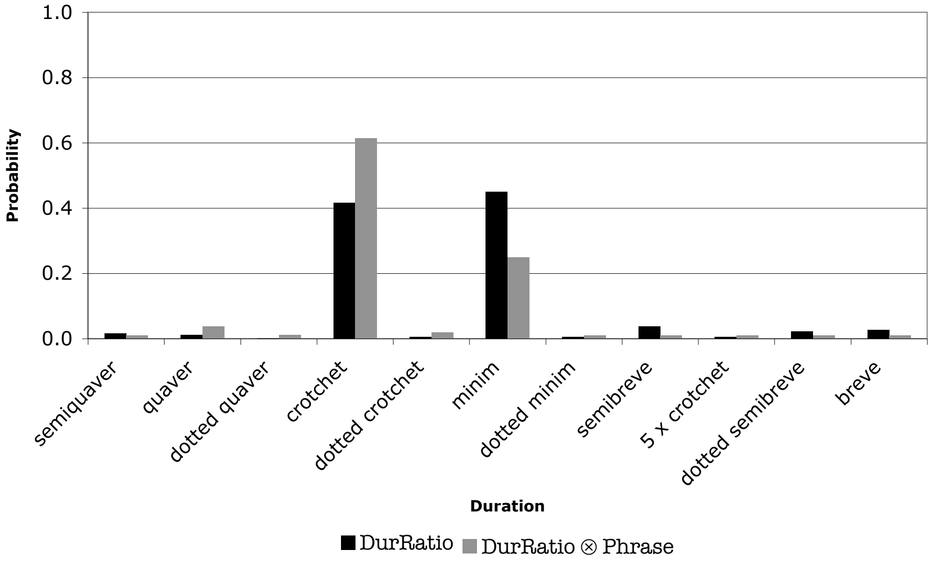


Figure 7.5: Bar chart showing LTM prediction probability distributions for viewpoints `DurRatio` and `DurRatio` ⊗ `Phrase`, predicting note durations at the beginning of the second phrase of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37).

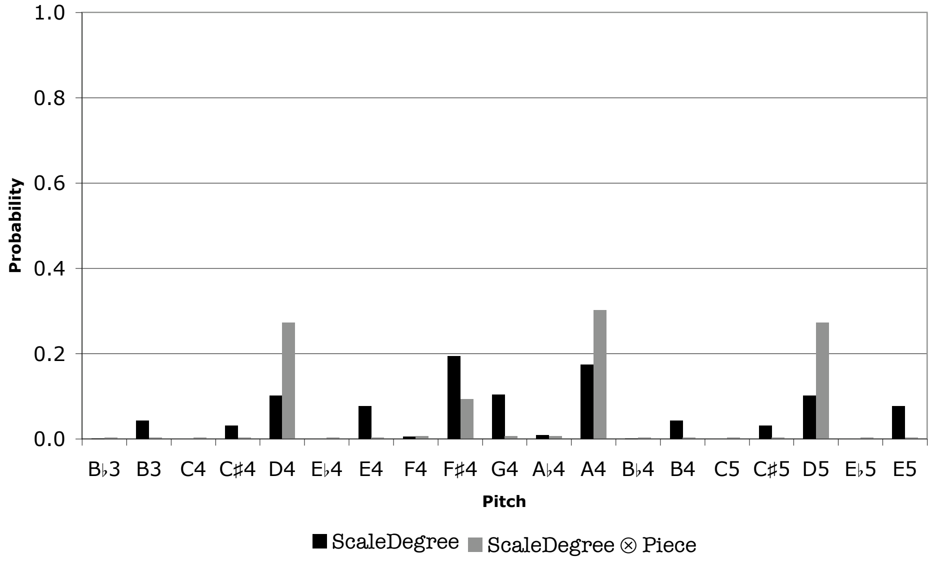


Figure 7.6: Bar chart showing LTM prediction probability distributions for viewpoints `ScaleDegree` and `ScaleDegree` ⊗ `Piece`, after conversion to distributions over pitches, predicting the pitch of the first note of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37).

7.2.5 `ScaleDegree` ⊗ `Piece`

The fifth viewpoint to be selected is `ScaleDegree` ⊗ `Piece` (which is not chosen for LTM+). A hymn tune almost always begins on either the tonic or the dominant, and almost exclusively ends on the tonic. Consequently, when the value of `Piece` is 1 (beginning of piece), `ScaleDegree` values of 0 and 7 will have high probabilities; and when the value of `Piece` is -1 (end of piece), a `ScaleDegree` value of 0 will have a particularly high probability.

Figures 7.6 and 7.7 show LTM prediction probability distributions for `ScaleDegree` and `ScaleDegree` ⊗ `Piece`, after conversion to distributions over pitches, respectively predicting the pitch of the first and last notes of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37). Knowing that this melody is in the key of D major, we can see from the `ScaleDegree` ⊗ `Piece` distribution that the first note is most likely to be the tonic, but is also highly likely to be the dominant, as expected. The third highest probability is assigned to the mediant, which is the actual first note of *Innocents*. In contrast, a good deal of the `ScaleDegree` probability mass is divided much more evenly between the tonic, mediant, dominant and supertonic. Moving on to the last note of the piece, we find that the tonic has a staggering 0.9566 probability according to `ScaleDegree` ⊗ `Piece`, compared with 0.5290 for `ScaleDegree`. The latter distribution contains moderately high probabilities for the mediant, supertonic and dominant.

Although `ScaleDegree` ⊗ `Piece` is demonstrably a highly effective viewpoint, it suffers from the same drawback as `ScaleDegree` alone; that is, different octaves are not distinguishable. It could be improved, for example, by linking it with `Tessitura`, such

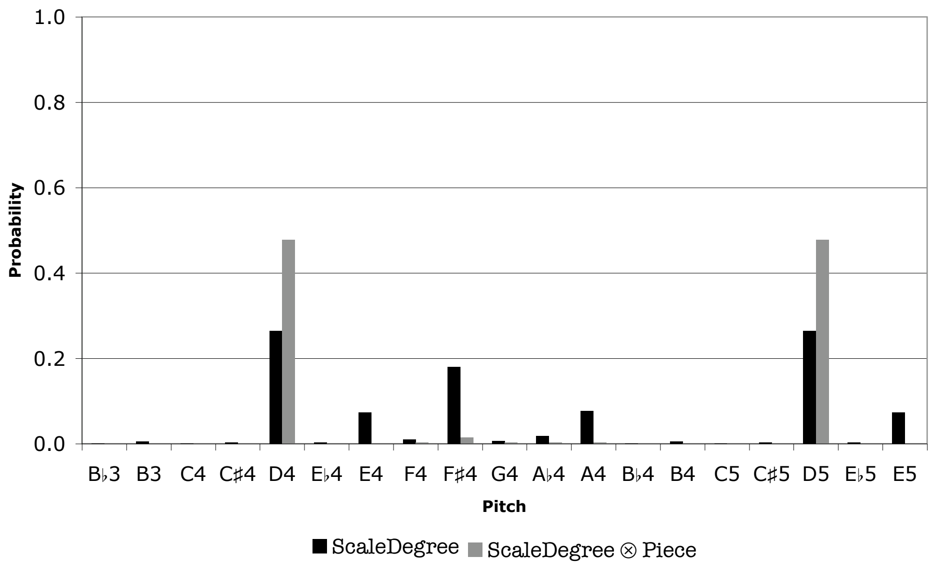


Figure 7.7: Bar chart showing LTM prediction probability distributions for viewpoints `ScaleDegree` and `ScaleDegree` ⊗ `Piece`, after conversion to distributions over pitches, predicting the pitch of the last note of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37).

that it becomes `ScaleDegree` ⊗ `Tessitura` ⊗ `Piece`. Alternatively, the proposed new viewpoint (introduced at the end of the discussion about `ScaleDegree` ⊗ `Tessitura` above) could be linked with `Piece`; this would obviate the need to exceed the current limit of two constituent viewpoints per linked viewpoint.

7.3 LTM+

The first five viewpoints selected for the best LTM+ multiple viewpoint system are presented in order of selection in Table 7.2, along with the deletion of `Duration`. Cross-entropies are shown at each stage. Before looking at some of the better performing viewpoints in detail, it is worth mentioning that the superiority of the updated model is indicated by the performance of the multiple viewpoint system used as the starting point for viewpoint selection, that is, {`Duration`, `Pitch`}. In the case of LTM, a cross-entropy of 4.37 bits/note results, whereas LTM+ improves this to 4.15 bits/note.

7.3.1 `ScaleDegree` ⊗ `Phrase`

The first viewpoint selected for the best LTM+ multiple viewpoint system is `ScaleDegree` ⊗ `Phrase` (not chosen for LTM). There is a strong tendency in the corpus for phrases to begin and end on the tonic, mediant or dominant; therefore when the value of `Phrase` is other than 0 (*i.e.*, at the beginning or end of a phrase), `ScaleDegree` values of 0, 4 and 7 will have high probabilities.

Figures 7.8 and 7.9 show LTM+ prediction probability distributions for viewpoints

Multiple viewpoint system	Cross-entropy (bits/note)
{Duration, Pitch}	4.15
+ ScaleDegree $\otimes$ Phrase	3.71
+ DurRatio $\otimes$ TactusPositionInBar	3.55
+ Interval $\otimes$ ScaleDegree $\ominus$ Tactus	3.43
+ DurRatio $\otimes$ LastInPhrase	3.37
− Duration	3.34
+ Interval $\otimes$ TactusPositionInBar	3.29

Table 7.2: Cross-entropies up to and including the fifth round of viewpoint addition/deletion during viewpoint selection of the best version 0 multiple viewpoint system (predicting Duration and Pitch) for LTM+ using corpus 'A'.

Scale-Degree and ScaleDegree $\otimes$ Phrase, after conversion to distributions over pitches, respectively predicting note pitches at the end of the first and at the beginning of the second phrase of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36). Knowing that this melody is in the key of E$\flat$ major, we can see from the ScaleDegree $\otimes$ Phrase distribution that the last note of the first phrase is most likely to be the tonic, but is also quite likely to be the mediant (the composed continuation of the melody). The supertonic (dominant of the dominant) has the third highest probability, with the dominant trailing in fourth place. The ScaleDegree probability mass is concentrated on the dominant, mediant and supertonic, whereas the tonic is much less likely. Moving on to the first note of the next phrase, we find that ScaleDegree assigns an extremely high probability to the subdominant, which continues the ascending major scale. The supertonic is the only other prediction with a significant likelihood. The ScaleDegree $\otimes$ Phrase distribution is completely different, with much of the probability mass distributed between the tonic (the actual continuation of the melody), mediant and dominant, as expected. The supertonic also has a relatively high probability.

7.3.2 DurRatio $\otimes$ TactusPositionInBar

The second viewpoint to be chosen for the best LTM+ system is DurRatio $\otimes$ TactusPositionInBar (not selected for LTM). This viewpoint is similar to Duration $\otimes$ Metre, inasmuch as it is essentially concerned with how duration is related to metrical position; although there is no attempt in this case to identify in advance metrical positions which may be considered equivalent. Let us take as an example a hymn tune with four minims to the bar. Recalling from §7.2 that syncopation is quite rare, at the second and fourth tactus beats a DurRatio value of 1 is likely to have a high probability. Obviously sub-tactus durations can interfere with this general rule; but such durations in N-gram contexts can be taken into account by learning from the corpus.

Figure 7.10 shows LTM+ prediction probability distributions for viewpoint DurRatio $\otimes$ TactusPositionInBar, predicting four note durations in the third bar of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33). The preceding two notes are minims.

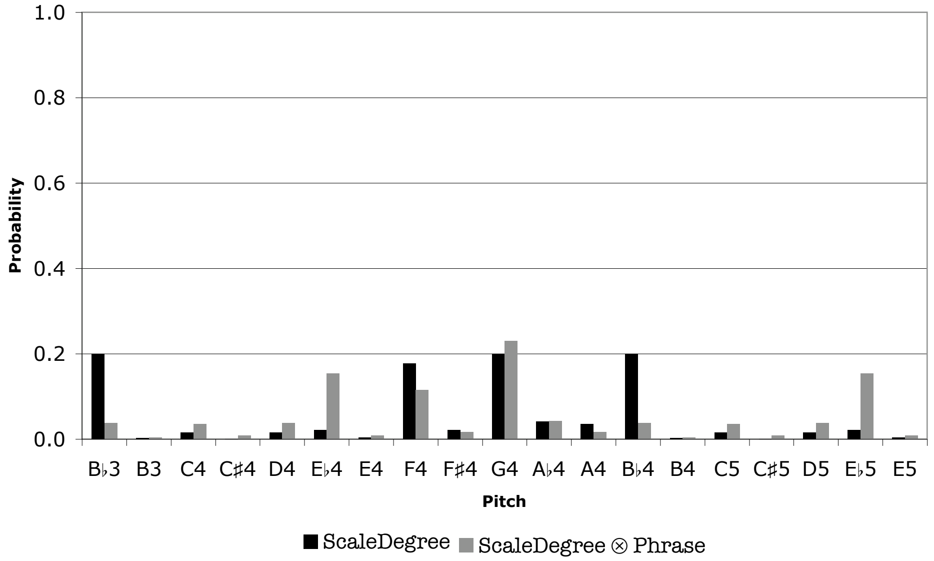


Figure 7.8: Bar chart showing LTM+ prediction probability distributions for viewpoints `ScaleDegree` and `ScaleDegree` ⊗ `Phrase`, predicting the pitch of the note at the end of the first phrase of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36).

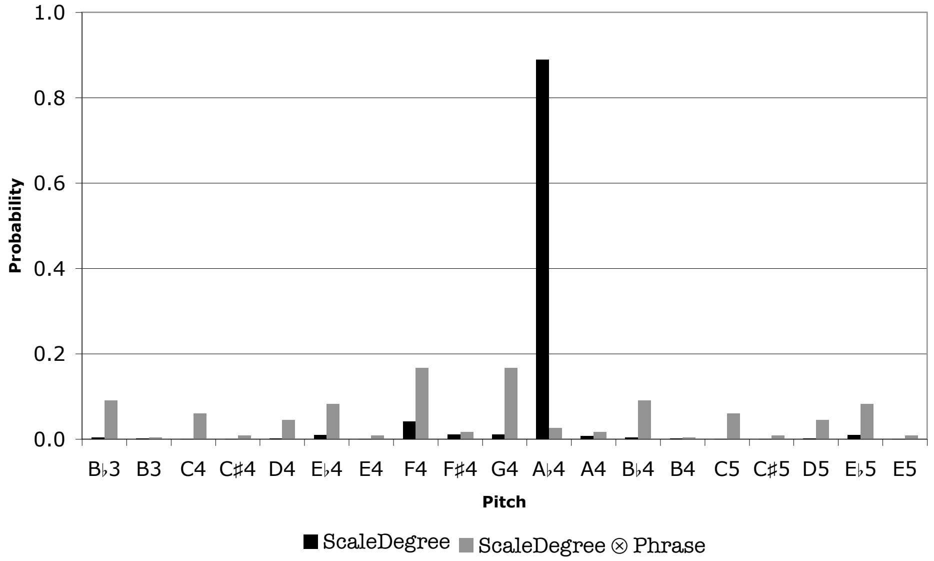


Figure 7.9: Bar chart showing LTM+ prediction probability distributions for viewpoints `ScaleDegree` and `ScaleDegree` ⊗ `Phrase`, predicting the pitch of the note at the beginning of the second phrase of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36).

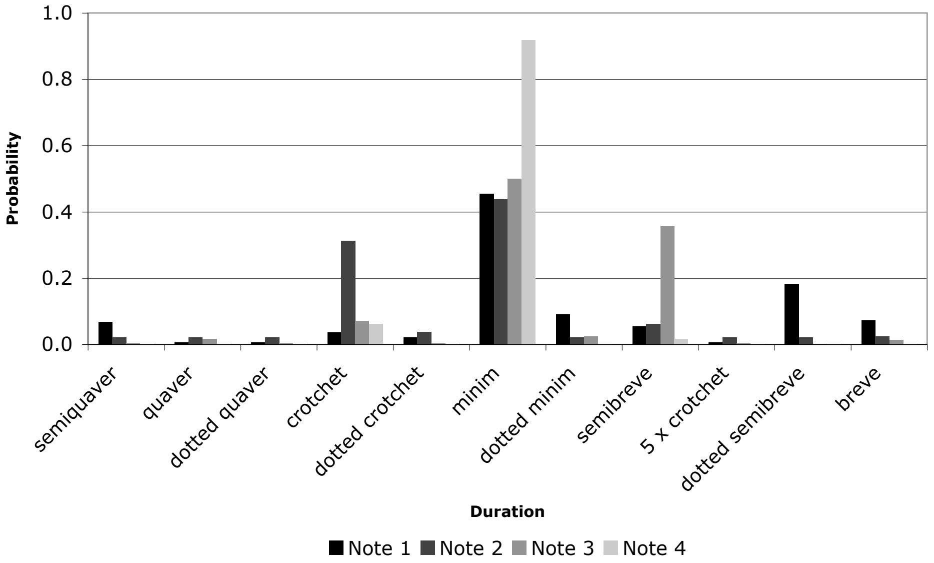


Figure 7.10: Bar chart showing LTM+ prediction probability distributions for viewpoint `DurRatio` ⊗ `TactusPositionInBar`, predicting four note durations in the third bar of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33).

According to the distributions, all four notes are most likely to be minims, which is how the melody was composed. For the first note, durations longer than a minim are generally more likely than those shorter than a minim, with a dotted semibreve having a particularly high probability. For the second note, a crotchet has by far the second-highest probability: in fact, a crotchet is twice as likely as all durations longer than a minim put together. There is a high likelihood of a semibreve for the the third note, which would take up the rest of the bar; and the fourth note is overwhelmingly likely to be a minim.

7.3.3 `Interval` ⊗ (`ScaleDegree` ⊖ `Tactus`)

The third viewpoint to be selected for LTM+ is `Interval` ⊗ (`ScaleDegree` ⊖ `Tactus`), which is not chosen for LTM. The selection of a viewpoint which is threaded at tactus intervals indicates that useful patterns can be found in the data by ignoring events not occurring on tactus beats; many such events are likely to be unessential in the music theoretic sense. As explained in §7.2, a drawback of `ScaleDegree` is that different octaves cannot be distinguished. This state of affairs can be remedied by linking it with `Interval`; in fact, because the dynamic `Interval` domain maps one-to-one to the `Pitch` domain, this linked viewpoint is able to completely distinguish between octaves.

Figure 7.11 shows LTM+ prediction probability distributions for viewpoints `Interval` ⊗ `ScaleDegree` and `Interval` ⊗ (`ScaleDegree` ⊖ `Tactus`), after conversion to distributions over pitches, predicting the pitch of the seventh note of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33). The preceding two notes are a crotchet G4 on a tactus beat followed by a non-tactus crotchet F4. The `Interval` ⊗ `ScaleDegree`

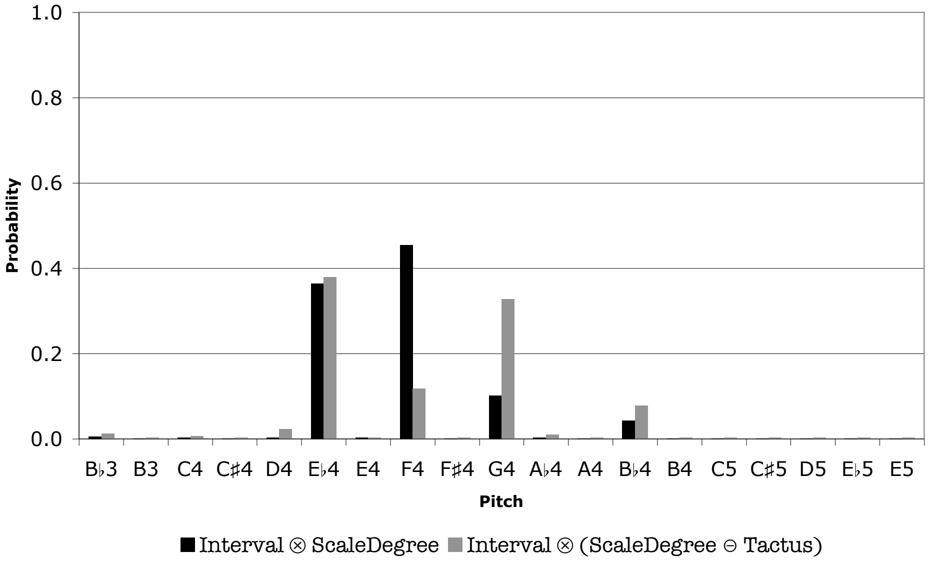


Figure 7.11: Bar chart showing LTM+ prediction probability distributions for viewpoints `Interval` ⊗ `ScaleDegree` and `Interval` ⊗ (`ScaleDegree` ⊖ `Tactus`), after conversion to distributions over pitches, predicting the pitch of the seventh note of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33).

distribution makes F4 the most likely continuation (*i.e.*, the same pitch as the previous note), and E♭4 also very likely; G4 has a much lower probability than either of these other pitches. In contrast, by ignoring the non-tactus F4, `Interval` ⊗ (`ScaleDegree` ⊖ `Tactus`) assigns the highest probability to E♭4; the second-highest to G4 (which is the composed continuation); and the third-highest to F4 (much lower than the other two probabilities, and only about a quarter of the corresponding probability in the `Interval` ⊗ `ScaleDegree` distribution).

7.3.4 `Duration` ⊗ `LastInPhrase`

The fourth viewpoint to be selected for LTM+ is `Duration` ⊗ `LastInPhrase` (not chosen for LTM, although a similar viewpoint, `DurRatio` ⊗ `Phrase`, was). The reason `Duration` ⊗ `LastInPhrase` performs so well is that the last note of a phrase is often long. This means that when the value of `LastInPhrase` is T (indicating a phrase ending), values of `Duration` corresponding to long notes will have relatively high probabilities.

Figure 7.12 shows LTM+ prediction probability distributions for viewpoints `Duration` and `Duration` ⊗ `LastInPhrase`, predicting the duration of the last note of the first phrase of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37). The note to be predicted follows two crotchets. `Duration` alone assigns a very high probability to a crotchet and a high probability to a minim; durations longer than this are deemed very unlikely by comparison. In the `Duration` ⊗ `LastInPhrase` distribution, on the other hand, a minim is given the highest probability; dotted minim and semibreve are fairly likely; and note lengths shorter than a minim are deemed very unlikely to occur.

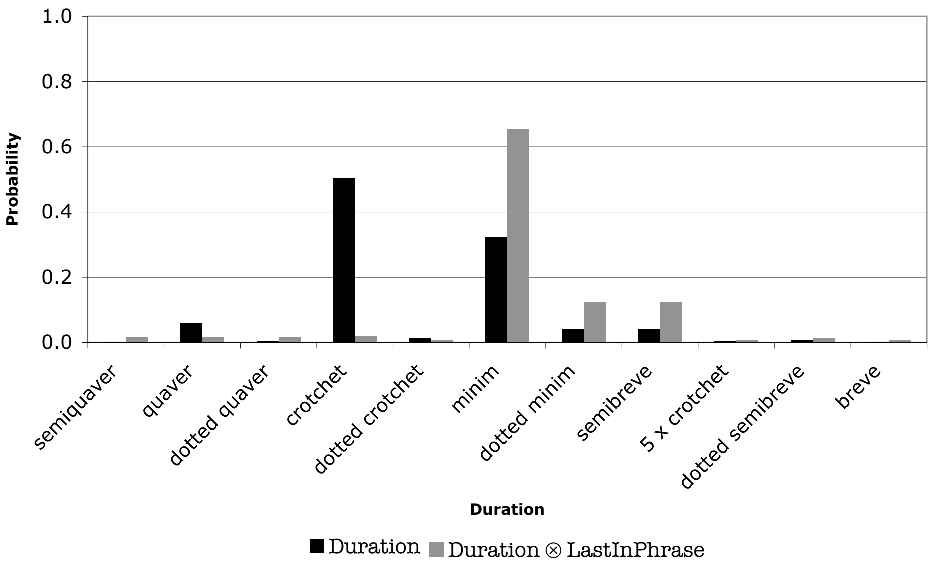


Figure 7.12: Bar chart showing LTM+ prediction probability distributions for viewpoints `Duration` and `Duration` ⊗ `LastInPhrase`, predicting the duration of the last note of the first phrase of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37).

7.3.5 `Interval` ⊗ `TactusPositionInBar`

The fifth viewpoint to be selected for the best LTM+ system is `Interval` ⊗ `TactusPositionInBar` (not chosen for LTM). This viewpoint is able to learn about potential passing notes and auxiliary notes of sub-tactus duration. If a note is sounded halfway through a tactus beat (*e.g.*, when `TactusPositionInBar` has a value of 1.5), it is likely to be one of these unessential notes, which means that `Interval` values from amongst −2, −1, 1 and 2 (depending on the context) will be of high probability. Similarly, if the immediately preceding note is sounded halfway through a tactus beat, then again `Interval` values of −2, −1, 1 and 2 will be highly probable.

Figures 7.13 and 7.14 show LTM+ prediction probability distributions for viewpoints `Interval` and `Interval` ⊗ `TactusPositionInBar`, after conversion to distributions over pitches, respectively predicting the pitch of the sixth and seventh notes of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33). The sixth note is not on a tactus beat, and follows a minim E♭4 and a crotchet G4. The `Interval` distribution indicates that A♭4 is by far the most probable continuation; F4 (the actual continuation) has a much lower probability. In contrast, `Interval` ⊗ `TactusPositionInBar` assigns F4 the highest probability. The relatively high probability given to F♯4 seems strange at first glance; but this sequence of intervals makes sense in a minor key passage (recalling that `Interval` is blind to key).

The seventh note is on a tactus beat. `Interval` reserves the highest probability for F4, the pitch of the previous note; and it also gives a high probability to E♭4, which completes a *me-re-doh* figure. E4 has the third highest probability (albeit much lower than the other two). In the `Interval` ⊗ `TactusPositionInBar` distribution, both E♭4

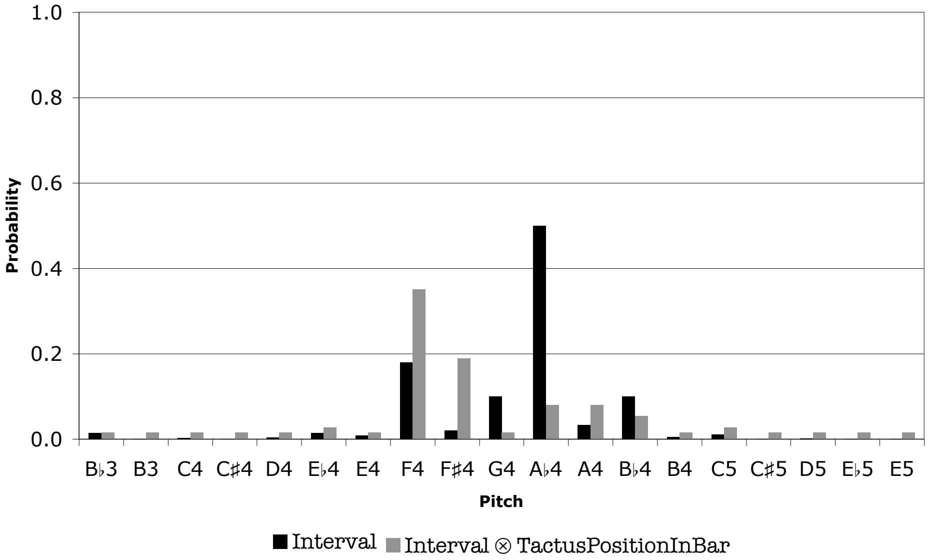


Figure 7.13: Bar chart showing LTM+ prediction probability distributions for viewpoints `Interval` and `Interval` ⊗ `TactusPositionInBar`, after conversion to distributions over pitches, predicting the pitch of the sixth note of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33).

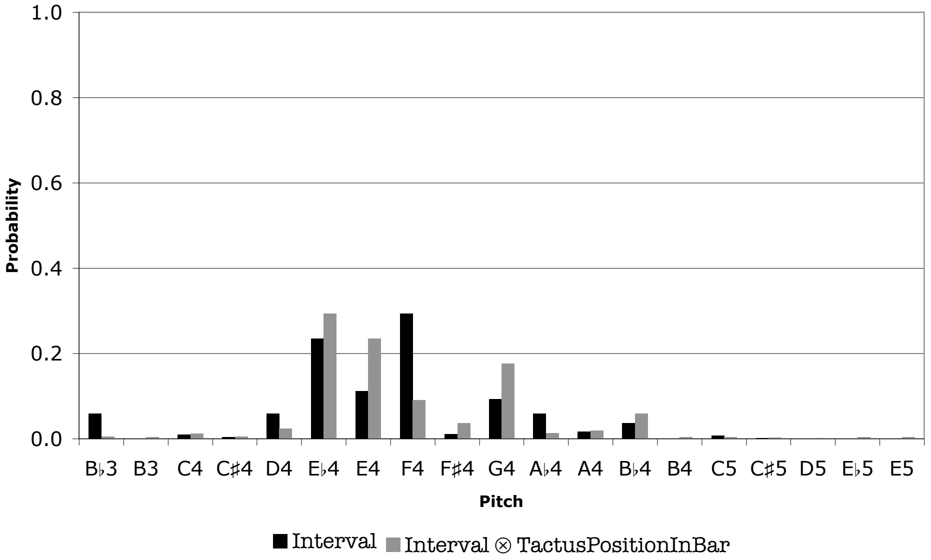


Figure 7.14: Bar chart showing LTM+ prediction probability distributions for viewpoints `Interval` and `Interval` ⊗ `TactusPositionInBar`, after conversion to distributions over pitches, predicting the pitch of the seventh note of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33).

Multiple viewpoint system	Cross-entropy (bits/note)
{Duration, Pitch}	4.82
+ Interval ⊗ InScale	4.55
+ Duration ⊗ LastInPhrase	4.44
− Duration	4.44
+ Duration ⊗ Metre	4.37
+ InScale ⊗ Tessitura	4.32
+ Duration ⊗ (ScaleDegree ⊖ FirstInPhrase)	4.30

Table 7.3: Cross-entropies up to and including the fifth round of viewpoint addition/deletion during viewpoint selection of the best version 0 multiple viewpoint system (predicting Duration and Pitch) for STM using arithmetic combination and corpus 'A'.

and E4 have high probabilities. At first sight, the latter looks rather odd; but an E4 would complete a *so-fa-me* figure in C major. The composed continuation, G4, has a higher probability in the Interval ⊗ TactusPositionInBar distribution than the Interval one.

7.4 STM Using Arithmetic Combination

The first five viewpoints selected for the best STM multiple viewpoint system using arithmetic combination are presented in order of selection in Table 7.3, along with the deletion of Duration. Cross-entropies are shown at each stage. The second viewpoint selected, Duration ⊗ LastInPhrase, has already been discussed in §7.3; while the third, Duration ⊗ Metre, was investigated in §7.2. These viewpoints, also chosen for STM using geometric combination, will therefore not be further analysed in detail. Suffice it to say, in regard to the latter, that the correlation between metrical position and length of note can easily be modelled well within the span of a single hymn tune, as can metrically related rhythmic patterns peculiar to that tune.

7.4.1 Interval ⊗ InScale

The first viewpoint to be selected for the best STM multiple viewpoint system is Interval ⊗ InScale (also chosen for STM using geometric combination). This viewpoint performs well because intervals ending on notes belonging to the scale, which appear more often in hymn tunes, have a relatively high probability. Interval is a particularly good viewpoint for the STM because the data is much less sparse than absolute or relative pitch data; for example, C4 to D4 and D4 to E4 are different from the point of view of pitch, but are the same interval. This means that it is quickly able to model the fact that melodies mostly move by step, and less often by larger intervals. Even within the confines of an STM, with limited data available, it is possible to beneficially modify Interval by linking it with InScale, because the linked domain is the same

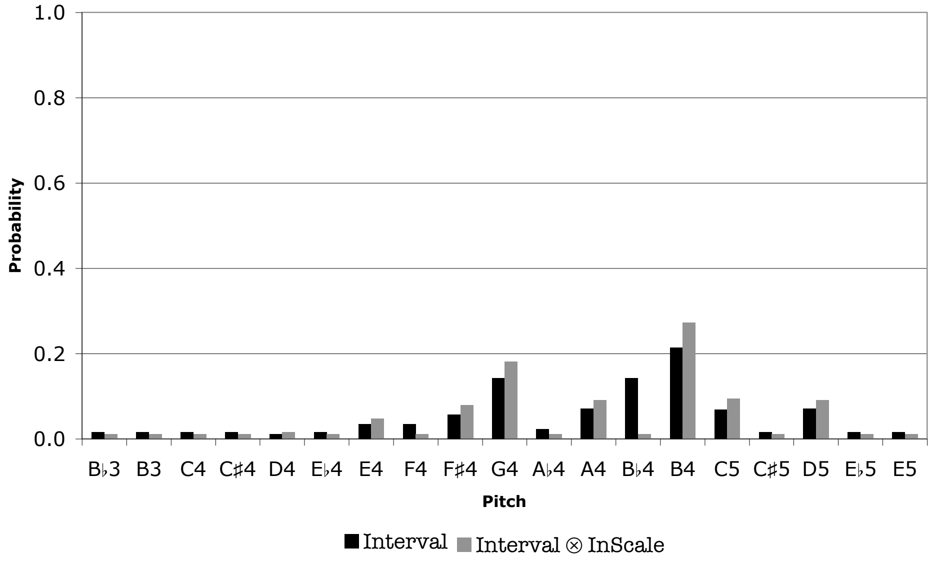


Figure 7.15: Bar chart showing STM prediction probability distributions for viewpoints `Interval` and `Interval` ⊗ `InScale`, after conversion to distributions over pitches, predicting the pitch of the final note of hymn tune *St. Edmund* (Vaughan Williams 1933, hymn no. 47).

size as the `Interval` one (each interval either ends on a note of the relevant scale or it does not).

Figure 7.15 shows STM prediction probability distributions for viewpoints `Interval` and `Interval` ⊗ `InScale`, after conversion to distributions over pitches, predicting the pitch of the final note of hymn tune *St. Edmund* (Vaughan Williams 1933, hymn no. 47). The melody is in the key of G major, and in both distributions B4 has the highest probability, followed by G4. Comparing the two distributions, `Interval` ⊗ `InScale` generally has higher probabilities for notes which belong to G major (including B4 and G4), and *vice versa.* In particular, it has a much lower probability for the non-scale note B♭4.

7.4.2 `InScale` ⊗ `Tessitura`

The fourth viewpoint to be selected for the STM is `InScale` ⊗ `Tessitura` (also chosen for STM using geometric combination). This viewpoint is useful for the STM, where available data is extremely limited, because it has a domain of only six members; some useful statistics can be accumulated very quickly. In particular, an `InScale` value of T linked with a `Tessitura` value of 0 (notes belonging to the scale in the comfortable part of the range) is likely to have a high probability. Conversely, an `InScale` value of F linked with `Tessitura` values of -1 or 1 (non-scale notes at the extremes of the range) will have particularly low probabilities.

Figure 7.16 shows STM prediction probability distributions for `InScale`, `Tessitura` and `InScale` ⊗ `Tessitura` viewpoint models, after conversion to distributions over

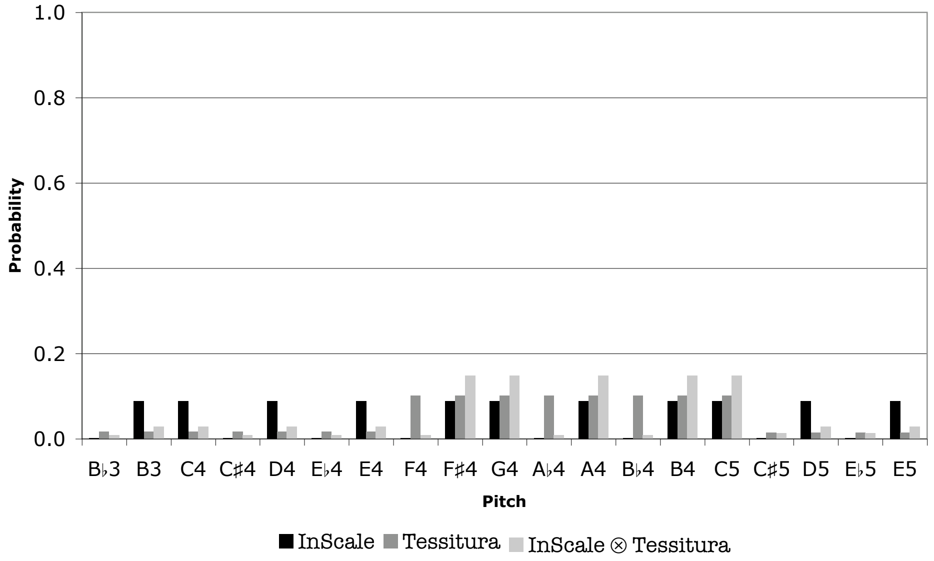


Figure 7.16: Bar chart showing STM prediction probability distributions for viewpoints `InScale`, `Tessitura` and `InScale` ⊗ `Tessitura`, after conversion to distributions over pitches, predicting the pitch of the final note of hymn tune *St. Edmund* (Vaughan Williams 1933, hymn no. 47).

pitches, predicting the pitch of the final note of hymn tune *St. Edmund* (Vaughan Williams 1933, hymn no. 47). The melody is in the key of G major, and as expected the highest `InScale` ⊗ `Tessitura` probabilities are assigned to mid-range notes belonging to this scale (*i.e.*, F♯4, G4, A4, B4 and C5).

7.4.3 `Duration` ⊗ (`ScaleDegree` ⊖ `FirstInPhrase`)

The fifth viewpoint to be selected is `Duration` ⊗ (`ScaleDegree` ⊖ `FirstInPhrase`) (also chosen for STM using geometric combination). The selection of a threaded viewpoint is, on the face of it, rather surprising considering the paucity of the data; many hymn tunes contain only four phrases (although tunes comprising six or more phrases are not unusual). It is the case, however, that in many hymn tunes the first note of all (or most) phrases is of precisely the same length. This means that, phrase by phrase during prediction, the probability of that particular duration increases. Similarly, it is also useful for the prediction of `Pitch` because it is quite usual for several phrases of a melody to begin on the same pitch.

Figure 7.17 shows STM prediction probability distributions for viewpoint `Duration` ⊗ (`ScaleDegree` ⊖ `FirstInPhrase`), predicting the duration of the first note of each of the four phrases of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37). Since all four phrases begin with a dotted crotchet, this note duration becomes ever more probable as the melody progresses.

Figure 7.18 shows such distributions for pitch. The appearance of F♯4 at the beginning of the first phrase means that it has the highest prediction probability at the

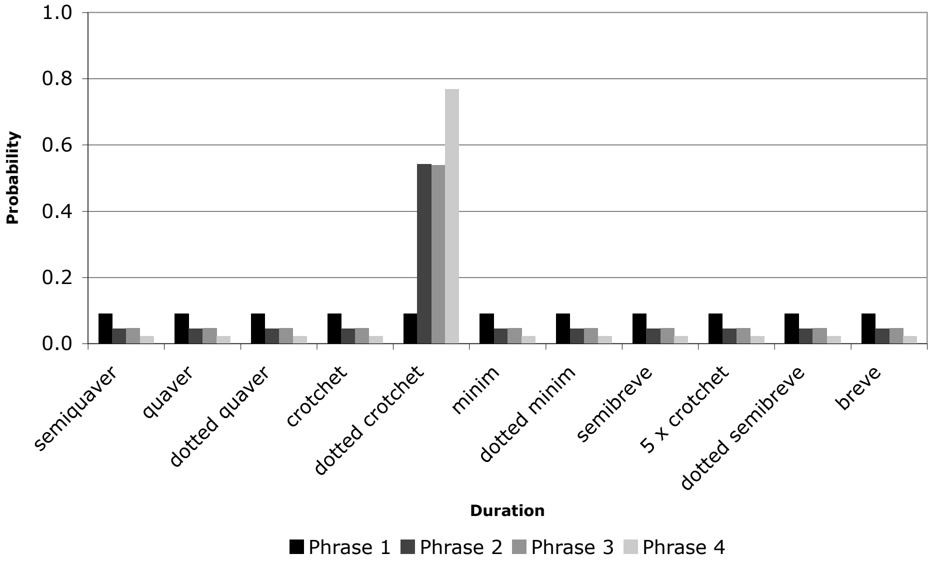


Figure 7.17: Bar chart showing STM prediction probability distributions for viewpoint `Duration` ⊗ (`ScaleDegree` ⊖ `FirstInPhrase`), predicting the duration of the first note of each of the four phrases of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37).

beginning of the second. On this occasion, however, the actual note at this point is D4; but since `ScaleDegree` is blind to octave, probability mass is shared between D4 and D5, which means that the most probable prediction for the third phrase is still F♯4. As it happens, this is the note which appears in the melody at this point. By the final phrase, D4, F♯4 and D5 are equally probable, with D4 being the composed note.

7.5 STM Using Geometric Combination

The first five viewpoints selected for the best STM multiple viewpoint system using geometric combination are presented in order of selection in Table 7.4, along with the deletion of `Duration`. Cross-entropies are shown at each stage. The first and fourth viewpoints selected, `Interval` ⊗ `InScale` and `InScale` ⊗ `Tessitura` respectively, have already been discussed in §7.4. The second viewpoint chosen, `Duration` ⊗ `LastInPhrase`, was investigated in §7.3; while the third, `Duration` ⊗ `Metre`, was examined in §7.2. These viewpoints, also chosen for STM using arithmetic combination, will therefore not be further analysed.

7.5.1 `Interval` ⊗ `Phrase`

The fifth viewpoint to be selected for the STM is `Interval` ⊗ `Phrase` (not chosen for STM using arithmetic combination). A similar viewpoint, `Interval` ⊗ `LastInPhrase`, has been discussed in §7.2; but here we are dealing with much sparser data, seeking regularities within a single piece. Such regularities are most likely to occur if phrases

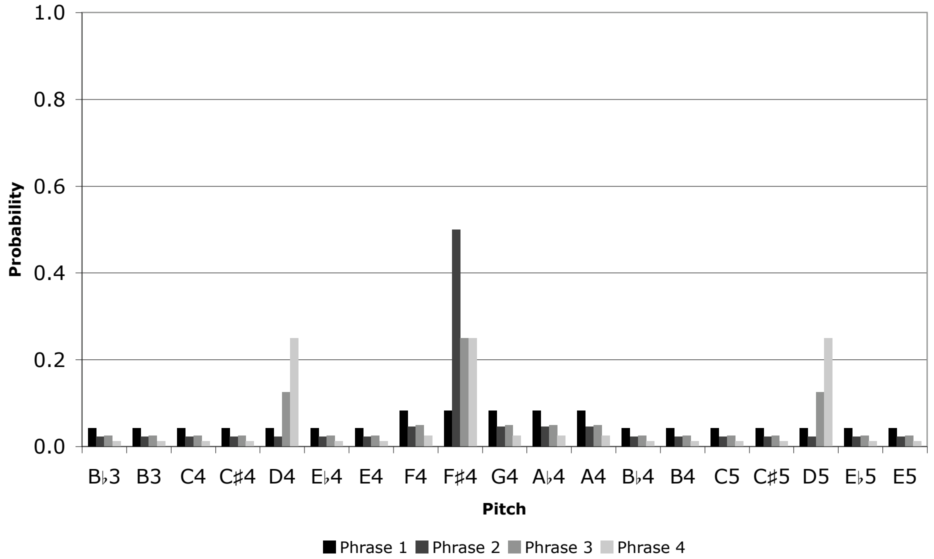


Figure 7.18: Bar chart showing STM prediction probability distributions for viewpoint Duration ⊗ (ScaleDegree ⊖ FirstInPhrase), predicting the pitch of the first note of each of the four phrases of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37).

Multiple viewpoint system	Cross-entropy (bits/note)
{Duration, Pitch}	4.82
+ Interval ⊗ InScale	4.60
+ Duration ⊗ LastInPhrase	4.48
+ Duration ⊗ Metre	4.43
− Duration	4.41
+ InScale ⊗ Tessitura	4.37
+ Interval ⊗ Phrase	4.35

Table 7.4: Cross-entropies up to and including the fifth round of viewpoint addition/deletion during viewpoint selection of the best version 0 multiple viewpoint system (predicting Duration and Pitch) for STM using geometric combination and corpus ‘A’.

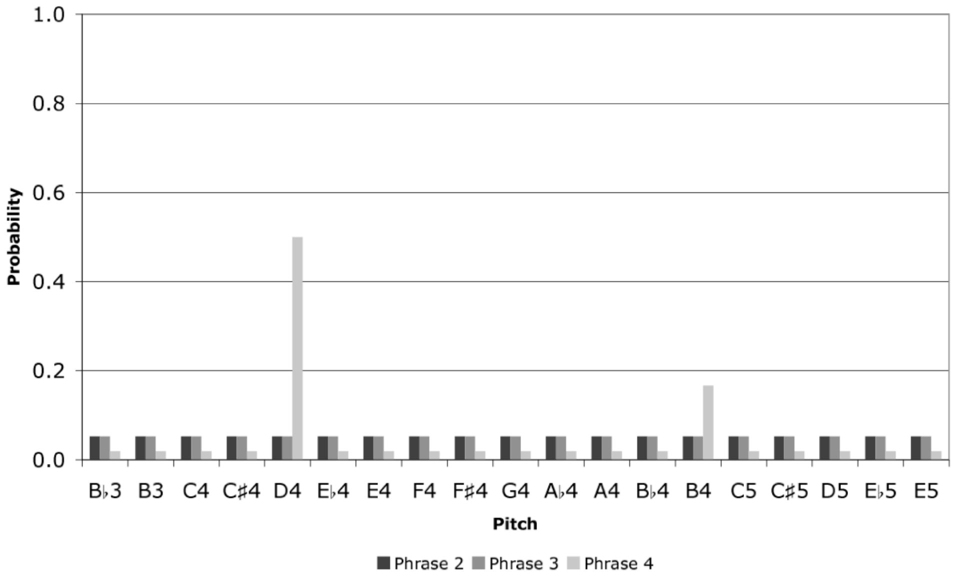


Figure 7.19: Bar chart showing STM prediction probability distributions for viewpoint `Interval` ⊗ `Phrase`, predicting the pitch of the first note of the second, third and fourth phrases of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37).

are repeated, as is the case in Vaughan Williams (1933), hymn no. 37, where the first and third phrases are the same, as are the first halves of the second and fourth phrases.

Figure 7.19 shows STM prediction probability distributions for viewpoint `Interval` ⊗ `Phrase`, predicting the pitch of the first note of the second, third and fourth phrases of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37), bearing in mind that this viewpoint is undefined at the start of the first phrase. There is a uniform distribution for the second phrase, as expected. This is also the case for the third phase, which at first glance seems odd, since the descending perfect fifth of the previous first in phrase should now be in the statistics; but such a leap would take the melody below the soprano range (*i.e.*, outside of the `Pitch` domain) and is disallowed. D4 is by far the most probable prediction for the fourth phrase, which coincides with the composed first of phrase.

Figure 7.20 shows such distributions for the last note of each of the four phrases. In this case, with all four phrases ending with a descending major second, the notes appearing in the hymnal are predicted with the highest possible probability (*i.e.*, the second phrase ends on E4, the third phrase on A4 and the fourth on D4).

7.5.2 Comparison of Viewpoints Selected for Long-term and Short-term Models

Viewpoints `Duration` ⊗ `LastInPhrase` and `Duration` ⊗ `Metre` were amongst the first chosen for both long- and short-term systems, suggesting that they are good all-round viewpoints for the prediction of `Duration`. Viewpoints `Duration` ⊗ (`ScaleDegree` ⊖ `FirstInPhrase`) and `Interval` ⊗ `Phrase`, chosen early for STM systems, were selected

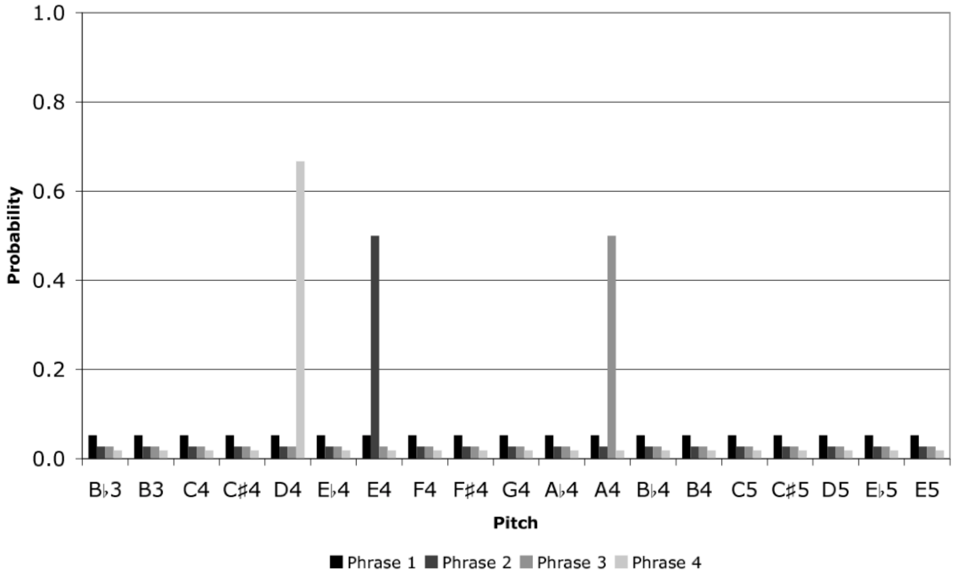


Figure 7.20: Bar chart showing STM prediction probability distributions for viewpoint `Interval` ⊗ `Phrase`, predicting the pitch of the last note of each of the four phrases of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37).

at a later stage for long-term ones. There was no other overlap between long- and short-term models amongst the better performing (first selected) viewpoints.

The paucity of data available for the STM compared with the relative abundance of data available for the LTM and LTM+ is the reason for such different viewpoints being selected. For example, in STM systems, `InScale` appears both on its own and in linked viewpoints `Interval` ⊗ `InScale` and `InScale` ⊗ `Tessitura`. There is a lot of music theoretic knowledge packed into the definition of `InScale` which does not have to be learned from the corpus; therefore it is particularly useful in connection with the data-poor STM. On the other hand, in long-term model systems, `ScaleDegree` appears in viewpoints `ScaleDegree` ⊗ `Tessitura`, `ScaleDegree` ⊗ `Piece`, `ScaleDegree` ⊗ `Phrase`, `Interval` ⊗ (`ScaleDegree` ⊖ `Tactus`) and many more. Whereas `InScale` was only able to assign probabilities on the basis of a note being in a scale or not, `ScaleDegree` is able to learn from the corpus individual probabilities for all degrees of the scale, making it a better performing viewpoint for long-term models.

7.6 BOTH

The first five viewpoints selected for the best BOTH multiple viewpoint system are presented in order of selection in Table 7.5, along with the deletions of `Pitch` and `Duration`. Cross-entropies are shown at each stage. The first viewpoint selected, `ScaleDegree` ⊗ `Phrase`, has already been discussed in §7.3. The second, fourth and fifth viewpoints chosen, `Duration` ⊗ `Metre`, `ScaleDegree` ⊗ `Tessitura` and `DurRatio` ⊗ `Phrase`, were investigated in §7.2. These viewpoints, also chosen for BOTH+, will therefore not be

Multiple viewpoint system	Cross-entropy (bits/note)
{`Duration`, `Pitch`}	4.17
+ `ScaleDegree` ⊗ `Phrase`	3.83
+ `Duration` ⊗ `Metre`	3.65
+ `Interval` ⊗ `FirstInBar`	3.47
− `Pitch`	3.47
+ `ScaleDegree` ⊗ `Tessitura`	3.38
+ `DurRatio` ⊗ `Phrase`	3.32
− `Duration`	3.27

Table 7.5: Cross-entropies up to and including the fifth round of viewpoint addition/deletion during viewpoint selection of the best version 0 multiple viewpoint system (predicting `Duration` and `Pitch`) for BOTH using corpus 'A'.

further analysed.

7.6.1 `Interval` ⊗ `FirstInBar`

The third viewpoint to be selected for BOTH is `Interval` ⊗ `FirstInBar` (also chosen for BOTH+, but at a much later stage). Let us consider the hypothesis that the success of this viewpoint is due more to a correlation between interval and phrase beginnings rather than to a more general correlation involving the beginnings of bars. When there is no anacrusis, `Interval` ⊗ `FirstInBar` and `Interval` ⊗ `FirstInPhrase` are both equipped to model the tendency for an interval larger than a major second to occur between phrases (albeit that the probability distributions of the former viewpoint are contaminated by first in bar data from elsewhere in the phrase). When there is an anacrusis, `Interval` ⊗ `FirstInBar` loses the ability to model this tendency (since the interval occurs before the first in bar), but instead gains the capability of capturing the arguably stronger tendency for a pitch leap to occur between an anacrusis and the first beat of the following bar (an ability which does not exist in `Interval` ⊗ `FirstInPhrase`). It should be noted that `Interval` ⊗ `FirstInPhrase` is in fact almost as good as `Interval` ⊗ `FirstInBar`: the cross-entropy would be 3.48 bits/note if it were selected instead at this stage. A new test viewpoint to identify the primary first in bar of a phrase can be envisaged, which may be used to investigate the validity of this hypothesis.

Figure 7.21 shows LTM prediction probability distributions for viewpoints `Interval` ⊗ `FirstInPhrase` and `Interval` ⊗ `FirstInBar`, after conversion to distributions over pitches, predicting the pitch of the second note of the last phrase of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36). This note occurs on the first beat of a bar. There is good agreement between the distributions, with two glaring exceptions: `Interval` ⊗ `FirstInBar` predicts B♭4 (the same as the preceding note) with a much higher probability (tending to refute the above hypothesis), and E♭5 (an ascending perfect fourth) with a much lower probability. The latter discrepancy is easily explicable in music theoretic terms. The preceding interval is also an ascending perfect

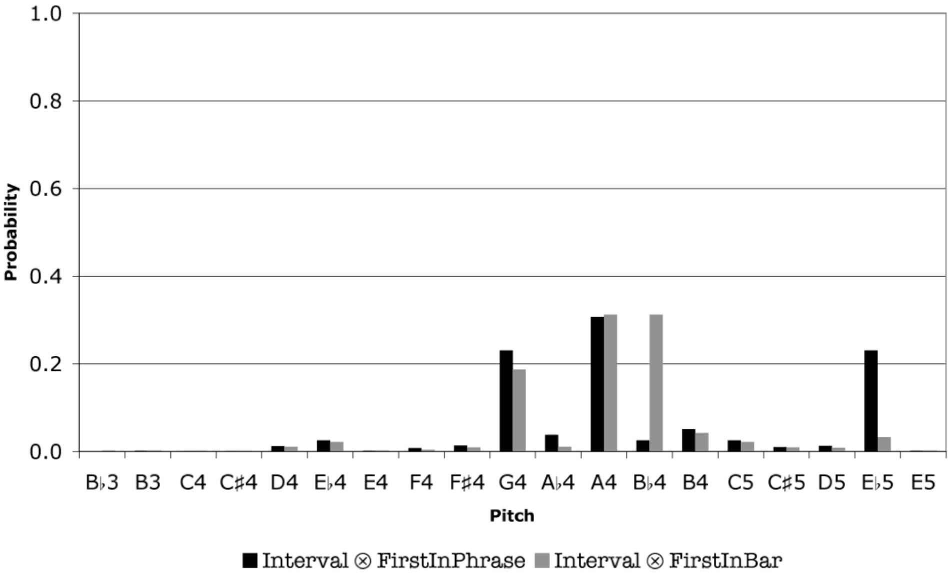


Figure 7.21: Bar chart showing LTM prediction probability distributions for viewpoints `Interval` ⊗ `FirstInPhrase` and `Interval` ⊗ `FirstInBar`, predicting the pitch of the second note of the last phrase of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36).

fourth, making a total of a minor seventh within three note onsets. This would be highly unusual within a phrase; but the interposition of a phrase boundary makes this a likely alternative at this point (*i.e.*, the interval straddling the phrase boundary is disregarded). On the other hand, G4 and A4 are favoured by both distributions, with G4 being the composed continuation.

Figure 7.22 shows these distributions for the STM. Notice that both distributions assign relatively high probabilities to G4 (the composed continuation), thus reinforcing that prediction. Apart from this, peaks in the LTM distributions correspond with troughs in the STM ones, and *vice versa.*

7.7 BOTH+

The first five viewpoints selected for the best BOTH+ multiple viewpoint system are presented in order of selection in Table 7.6, along with the deletion of `Duration`. Cross-entropies are shown at each stage. The first viewpoint selected, `ScaleDegree` ⊗ `Phrase` (also chosen for BOTH and BOTH+ using corpus 'B'), has already been examined in §7.3. The second viewpoint, `Interval` ⊗ `FirstInPhrase` (also chosen for BOTH, but at a much later stage), was discussed in relation to viewpoint `Interval` ⊗ `FirstInBar` in §7.6 above. By virtue of its selection here, `Interval` ⊗ `FirstInPhrase` has proven itself to be a worthwhile viewpoint in its own right. Third viewpoint `Duration` ⊗ `Metre` (also chosen for BOTH and BOTH+ using corpus 'B') was investigated in §7.2, as was fifth viewpoint `DurRatio` ⊗ `Phrase` (also chosen for BOTH, BOTH+ using corpus 'A+B'

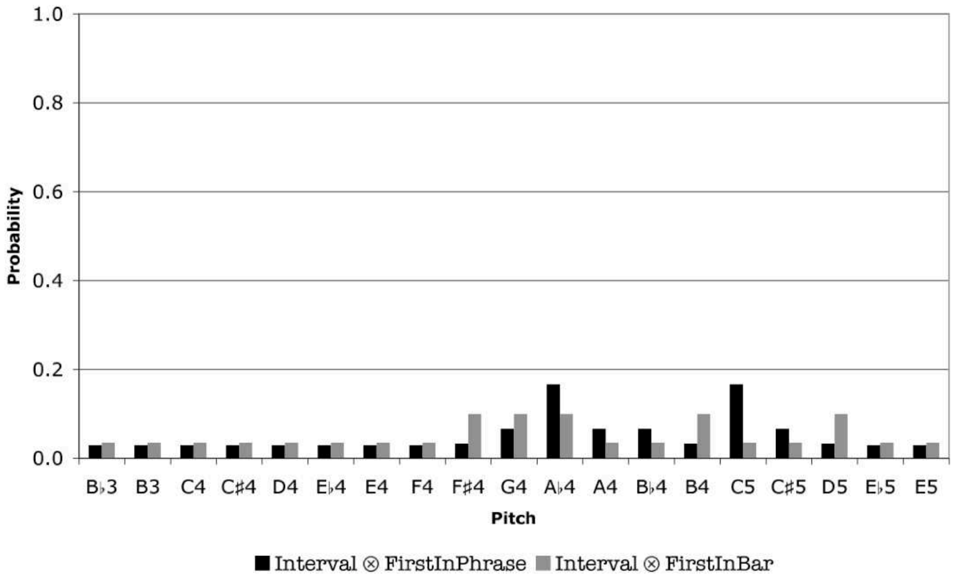


Figure 7.22: Bar chart showing STM prediction probability distributions for viewpoints `Interval` ⊗ `FirstInPhrase` and `Interval` ⊗ `FirstInBar`, predicting the pitch of the second note of the last phrase of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36).

and, at a later stage, BOTH+ using corpus 'B'). These viewpoints will therefore not be further analysed.

7.7.1 `ScaleDegree` ⊗ `Metre`

The fourth viewpoint to be selected for BOTH+ is `ScaleDegree` ⊗ `Metre` (also chosen at a later stage for BOTH and corpus 'B'; not selected for corpus 'A+B'). The performance of this viewpoint is ample evidence that there is a correlation between `ScaleDegree` and metrical importance. Inspection of half-a-dozen hymn tunes from corpus 'A' suggests that `ScaleDegree` values of 0 and 4 (tonic and mediant respectively) are more common than other values on the strongest tactus beat of the bar, while values of 0, 2, 4 and 7

Multiple viewpoint system	Cross-entropy (bits/note)
{`Duration`, `Pitch`}	4.13
+ `ScaleDegree` ⊗ `Phrase`	3.73
+ `Interval` ⊗ `FirstInPhrase`	3.56
+ `Duration` ⊗ `Metre`	3.42
+ `ScaleDegree` ⊗ `Metre`	3.35
+ `DurRatio` ⊗ `Phrase`	3.29
− `Duration`	3.23

Table 7.6: Cross-entropies up to and including the fifth round of viewpoint addition/deletion during viewpoint selection of the best version 0 multiple viewpoint system (predicting `Duration` and `Pitch`) for BOTH+ using corpus 'A'.

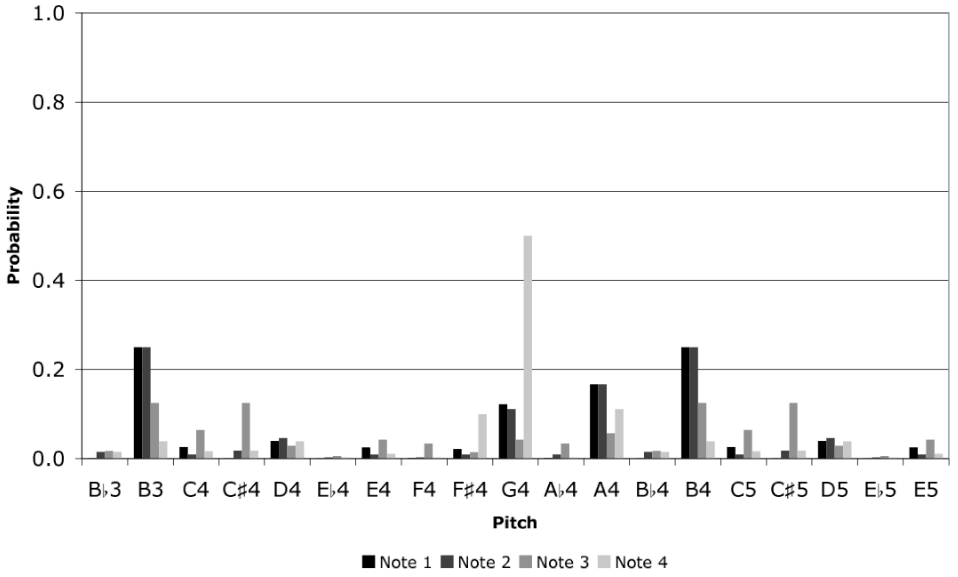


Figure 7.23: Bar chart showing LTM prediction probability distributions for viewpoint `ScaleDegree` ⊗ `Metre`, predicting the pitch of the notes in the penultimate bar of hymn tune *St. Edmund* (Vaughan Williams 1933, hymn no. 47).

(tonic, supertonic, mediant and dominant respectively) are particularly common on the weakest tactus beats.

Figure 7.23 shows LTM prediction probability distributions for viewpoint `ScaleDegree` ⊗ `Metre`, after conversion to distributions over pitches, predicting the pitch of the notes in the penultimate bar of hymn tune *St. Edmund* (Vaughan Williams 1933, hymn no. 47). The melody is in G major, and the previous bar ends on a C5. The predictions with the highest probability for the first note of the bar (with the strongest tactus beat) are B3 and B4 (mediant). This is another example of how a `ScaleDegree`-like viewpoint able to distinguish between octaves would be useful. The note in the hymnal at this point, A4, and G4 (tonic) are the only other predictions with comparable probabilities. The prediction probabilities for the second note of the bar (having a weak tactus beat) are remarkably similar. The composed continuation, D5, has the fifth-highest probability (albeit that it is much lower than the fourth-highest). The third note (a secondary strong beat) has a very different distribution: the highest probabilities are for B3, C♯4, B4 and C♯5, suggesting a strong possibility of a modulation to the dominant here. The actual note, G4, has quite a low probability. The distribution for the fourth note (a weak tactus beat) is different again, with G4 this time being by far the most likely prediction. F♯4 and A4 are reasonably likely; but the composed continuation, C5, is deemed highly unlikely.

Figure 7.24 shows these distributions for the STM. The top four predictions for the first note are the same as those in the LTM distribution (although this time G4 is most likely). A4 is overwhelmingly the most likely prediction for the second and third notes, while for the fourth note G4 and A4 have the highest probabilities (in line with the LTM

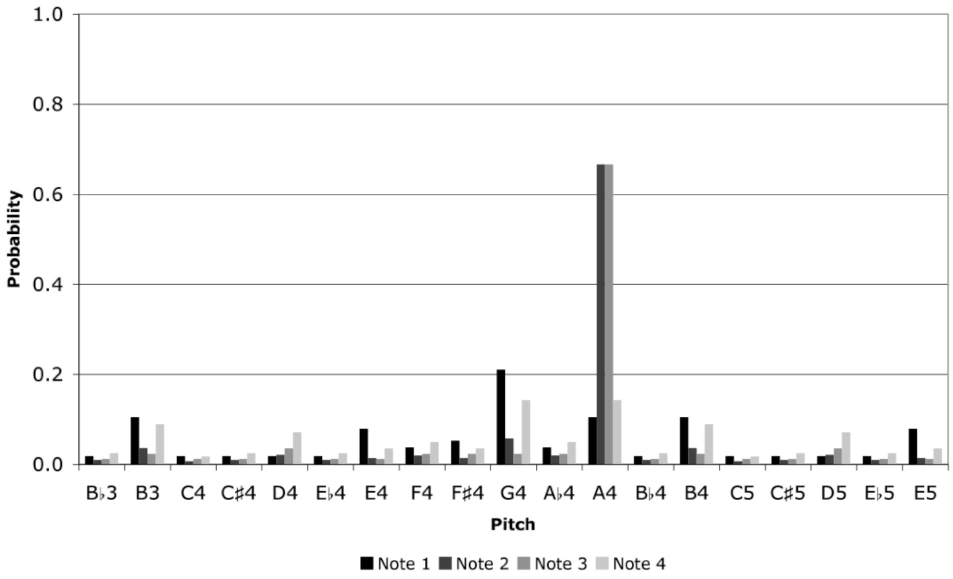


Figure 7.24: Bar chart showing STM prediction probability distributions for viewpoint `ScaleDegree` ⊗ `Metre`, predicting the pitch of the notes in the penultimate bar of hymn tune *St. Edmund* (Vaughan Williams 1933, hymn no. 47).

distribution). The poor performance of this viewpoint with respect to this particular bar is likely to be due to the unusually jumpy nature of the melody at this point.

7.8 BOTH+ (Corpus 'B')

The first five viewpoints selected for the best BOTH+ (corpus 'B') multiple viewpoint system are presented in order of selection in Table 7.7, along with the deletion of `Duration`. Cross-entropies are shown at each stage. The first viewpoint selected, `ScaleDegree` ⊗ `Phrase`, has already been examined in §7.3, while third viewpoint `Duration` ⊗ `Metre` was discussed in §7.2. Both of these viewpoints were also chosen for corpus 'A', but not for 'A+B'. Fifth viewpoint `Duration` ⊗ `Phrase` (chosen later in the process for corpus 'A+B', but not chosen for 'A') is effective, for example, because the last note of a phrase is very often long. Similar viewpoints `DurRatio` ⊗ `Phrase` and `Duration` ⊗ `LastInPhrase` have already been discussed in §7.2 and §7.3 respectively. These viewpoints will therefore not be further analysed.

7.8.1 `DurRatio` ⊗ `Interval`

The second viewpoint to be selected for BOTH+ (Corpus 'B') is `DurRatio` ⊗ `Interval` (chosen for neither corpus 'A' nor corpus 'A+B'). Although this viewpoint performs well, the information it contains is subsumed by a combination of subsequently selected viewpoints, as evidenced by its eventual deletion from the system. Since it was deleted directly after the addition of viewpoint `DurRatio` ⊗ `Phrase`, and since `Duration` ⊗ `Phrase` and `Interval` ⊗ `Phrase` had also been selected, it is reasonable to assume that

Multiple viewpoint system	Cross-entropy (bits/note)
{Duration, Pitch}	3.91
+ ScaleDegree ⊗ Phrase	3.59
+ DurRatio ⊗ Interval	3.42
+ Duration ⊗ Metre	3.30
+ Contour ⊗ ScaleDegree	3.24
+ Duration ⊗ Phrase	3.19
− Duration	3.16

Table 7.7: Cross-entropies up to and including the fifth round of viewpoint addition/deletion during viewpoint selection of the best version 0 multiple viewpoint system (predicting Duration and Pitch) for BOTH+ using corpus 'B'.

DurRatio ⊗ Interval is good at modelling what is happening at phrase boundaries. Prediction of Duration by the STM is likely to be good where there are repeats; but LTM+ is not expected to be a good predictor of Duration. Once the durations are known, since long notes often occur at phrase boundaries the Pitch prediction capability of this viewpoint should approach that of Interval ⊗ Phrase. At last in phrase, a step interval is most likely; whereas at first in phrase, something other than a step interval is to be expected.

Figure 7.25 shows LTM+ and STM prediction probability distributions for viewpoint DurRatio ⊗ Interval, after conversion to distributions over durations, predicting the duration of the note at the end of the fourth phrase of hymn tune *Würzburg* (Vaughan Williams 1933, hymn no. 127). Notice that semiquavers and dotted quavers are absent from corpus 'B', and consequently from the distributions. The third and fourth phrases are exact repeats of the first and second phrases, so it is no surprise that the phrase-concluding semibreve is predicted with the highest probability by the STM. On the other hand, the LTM regards a semibreve to be less probable than either a minim or a crotchet. Figure 7.26 shows such distributions for the beginning of the fifth phrase of the hymn tune. This time the two distributions are in good agreement, with a minim being deemed most likely (as is indeed the case).

Figure 7.27 shows LTM+ and STM prediction probability distributions for viewpoint DurRatio ⊗ Interval, after conversion to distributions over pitches, predicting the pitch of the note at the end of the fourth phrase of hymn tune *Würzburg* (Vaughan Williams 1933, hymn no. 127). Notice that B♭4 and B4 are absent from corpus 'B', but F5 is present (reflected in the distributions). The composed continuation, A4, is (equally) the most probable prediction in both distributions, undoubtedly aided in each case by the repeat. G4 and F4 are assigned quite high probabilities by the LTM+ in spite of F4 not appearing in the scale of G major (the key of the melody); but of course Interval is blind to key.

Figure 7.28 shows such distributions for the beginning of the fifth phrase of the hymn tune. The repeat is almost certainly responsible for the large probabilities associated

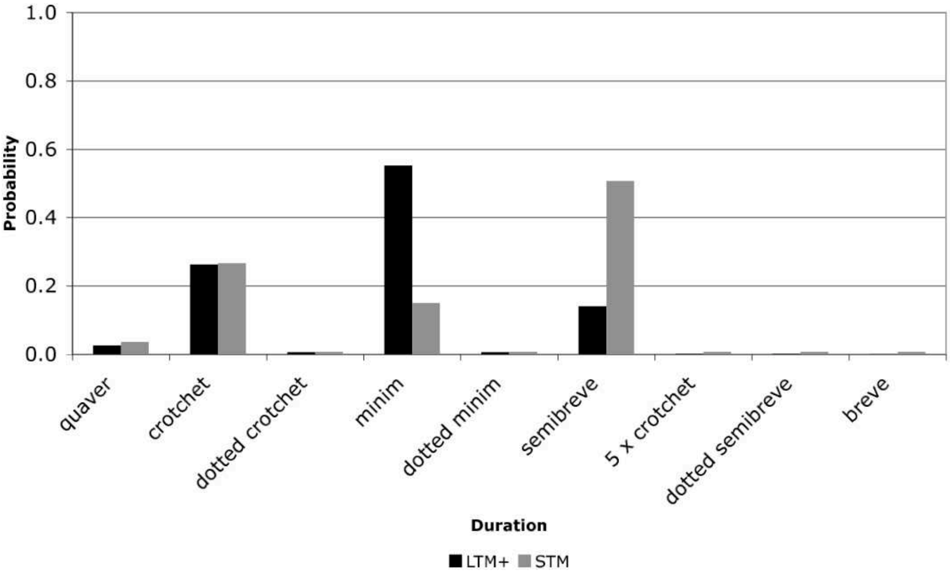


Figure 7.25: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint `DurRatio` ⊗ `Interval`, predicting the duration of the note at the end of the fourth phrase of hymn tune *Würzburg* (Vaughan Williams 1933, hymn no. 127).

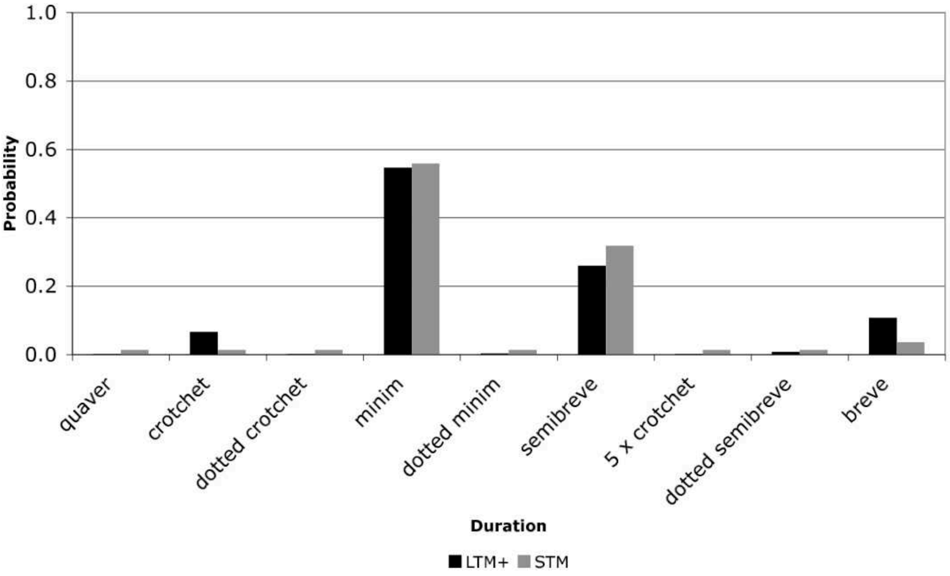


Figure 7.26: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint `DurRatio` ⊗ `Interval`, predicting the duration of the note at the beginning of the fifth phrase of hymn tune *Würzburg* (Vaughan Williams 1933, hymn no. 127).

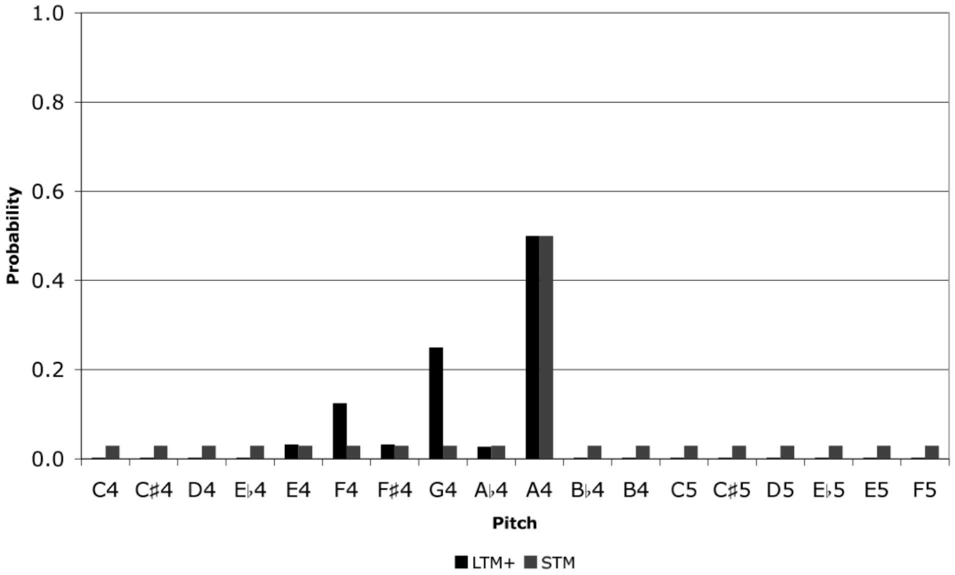


Figure 7.27: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint `DurRatio` ⊗ `Interval`, predicting the pitch of the note at the end of the fourth phrase of hymn tune *Würzburg* (Vaughan Williams 1933, hymn no. 127).

with the prediction of D4, since this is how phrases one and three begin. The fact that this is the only leap assigned a high probability is at first rather surprising; but it can be explained by the fact that this viewpoint is also able to model what could, in a harmonic context, be considered passing or auxiliary notes, where stepwise motion is *de rigueur.*

7.8.2 `Contour` ⊗ `ScaleDegree`

The fourth viewpoint to be selected for BOTH+ (Corpus 'B') is `Contour` ⊗ `ScaleDegree` (not chosen for corpus 'A' nor 'A+B'). It was argued during the discussion on viewpoint `ScaleDegree` ⊗ `Tessitura` (see §7.2) that although `ScaleDegree` is in itself a very effective viewpoint, linking it with `Contour` (amongst other possibilities) improves its performance by providing a means to distinguish between octaves.

Figure 7.29 shows LTM+ and STM prediction probability distributions for viewpoint `Contour` ⊗ `ScaleDegree`, after conversion to distributions over pitches, predicting the pitch of the fifth note of the penultimate bar of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33). The fact that the composed continuation, G4, is by a long way deemed most likely by the STM is due to the fact that it and the preceding four notes of the melody also occur in the ninth bar. The LTM+ assigns an even higher probability to this note, which is a continuation of a descending scale. There is also a fair likelihood of an A♭4 (a repeat of the preceding note) according to the STM.

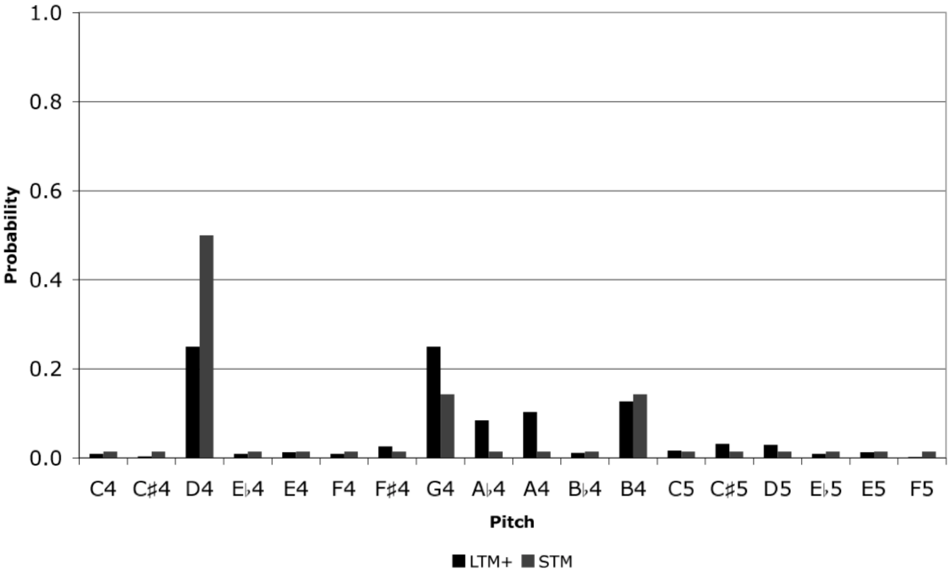


Figure 7.28: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint `DurRatio` ⊗ `Interval`, predicting the pitch of the note at the beginning of the fifth phrase of hymn tune *Würzburg* (Vaughan Williams 1933, hymn no. 127).

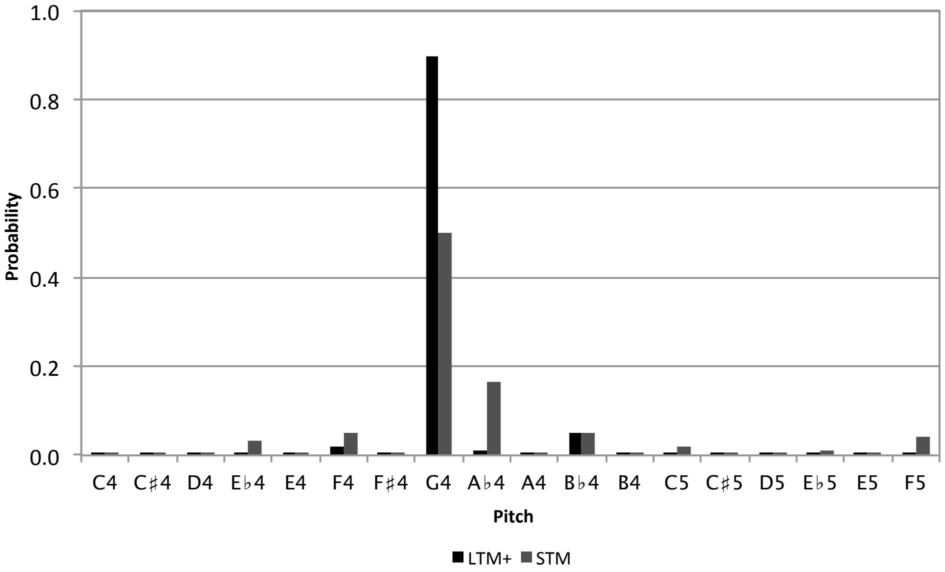


Figure 7.29: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint `Contour` ⊗ `ScaleDegree`, predicting the pitch of the fifth note of the penultimate bar of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33).

Multiple viewpoint system	Cross-entropy (bits/note)
{Duration, Pitch}	3.91
+ Duration ⊗ ScaleDegree	3.58
+ Interval ⊗ Phrase	3.42
+ DurRatio ⊗ Phrase	3.28
+ IntFirstInPhrase ⊗ ScaleDegree	3.21
+ Duration ⊗ TactusPositionInBar	3.15
− Duration	3.14

Table 7.8: Cross-entropies up to and including the fifth round of viewpoint addition/deletion during viewpoint selection of the best version 0 multiple viewpoint system (predicting Duration and Pitch) for BOTH+ using corpus 'A+B'.

7.9 BOTH+ (Corpus 'A+B')

The first five viewpoints selected for the best BOTH+ (corpus 'A+B') multiple viewpoint system are presented in order of selection in Table 7.8, along with the deletion of Duration. Cross-entropies are shown at each stage. The second viewpoint selected, Interval ⊗ Phrase (also chosen for corpus 'B' but not for 'A'), has already been discussed with respect to the STM in §7.5; and similar viewpoint Interval ⊗ LastInPhrase has been discussed in §7.2 regarding the LTM. Third viewpoint DurRatio ⊗ Phrase (also chosen for corpora 'A' and 'B') was examined in §7.2. Fifth viewpoint Duration ⊗ TactusPositionInBar is preferred to Duration ⊗ Metre in this case: viewpoint TactusPositionInBar is similar to Metre, but is defined more simply and is more fine-grained (*i.e.*, it has a larger domain). There is no prescription of metrical equivalence as there is for Metre. Similar viewpoint DurRatio ⊗ TactusPositionInBar has been discussed in §7.3.

7.9.1 Duration ⊗ ScaleDegree

The first viewpoint to be selected for the best BOTH+ (Corpus 'A+B') multiple viewpoint system is Duration ⊗ ScaleDegree (not chosen for corpus 'A' nor 'B'). The addition of this viewpoint to {Duration, Pitch} reduces the cross-entropy from 3.91 to 3.58 bits/note. The likely reason for the effectiveness of this viewpoint is that long notes often occur at phrase boundaries; therefore its behaviour should be similar to that of ScaleDegree ⊗ Phrase, which is already known to be a good viewpoint (see §7.3). In fact, the addition of ScaleDegree ⊗ Phrase instead of Duration ⊗ ScaleDegree at this point results in a cross-entropy of 3.59 bits/note, which is very similar indeed. It is also conceivable that Duration ⊗ ScaleDegree could model the regularities of notes which, in the context of harmony, may be considered passing or auxiliary notes; but other viewpoints, such as Duration ⊗ Interval, are better suited to this task. Viewpoint Duration ⊗ ScaleDegree is deleted later in the viewpoint selection process immediately after the addition of ScaleDegree ⊗ Tessitura, which is known to have

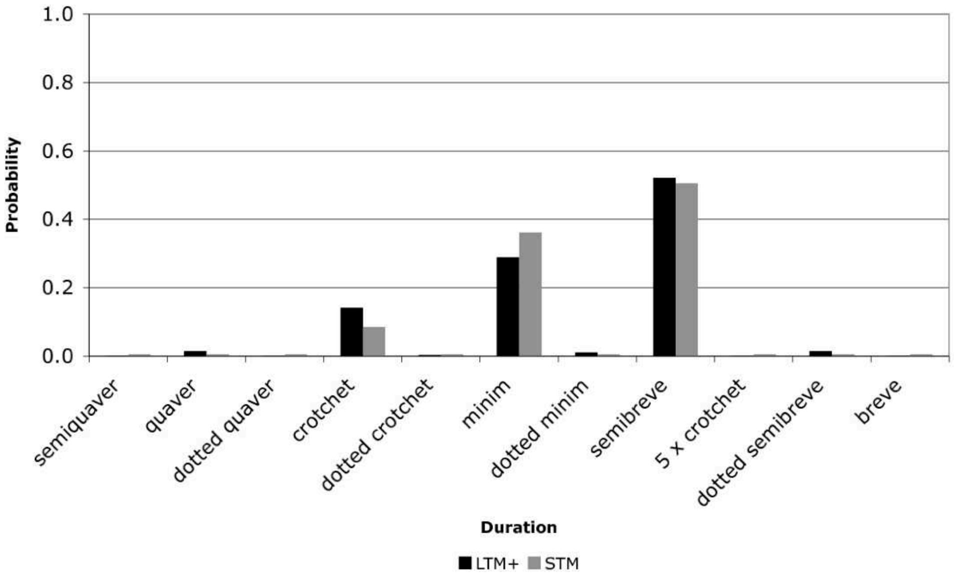


Figure 7.30: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint `Duration` ⊗ `ScaleDegree`, predicting the duration of the note at the end of the fourth phrase of hymn tune *Würzburg* (Vaughan Williams 1933, hymn no. 127).

an advantage over `ScaleDegree` (see §7.2).

Figure 7.30 shows LTM+ and STM prediction probability distributions for viewpoint `Duration` ⊗ `ScaleDegree`, after conversion to distributions over durations, predicting the duration of the note at the end of the fourth phrase of hymn tune *Würzburg* (Vaughan Williams 1933, hymn no. 127). Now that corpus 'A+B' is employed, semiquavers and dotted quavers return to the distributions. The two distributions are well matched, almost certainly by virtue of the fact that the third and fourth phrases are exactly the same as the first and second. The duration in the melody at this point, a semibreve, is deemed the most likely prediction, followed by minim and crotchet.

Figure 7.31 shows such distributions for the beginning of the fifth phrase of the hymn tune. A minim (as in the hymn tune) is far and away the most likely prediction according to both distributions. This is again largely due to the repeat, where a minim occurs at the beginning of the third phrase after an identical context. There is some disagreement between the distributions over which of a crotchet or semibreve is more likely.

Figure 7.32 shows LTM+ and STM prediction probability distributions for viewpoint `Duration` ⊗ `ScaleDegree`, after conversion to distributions over pitches, predicting the pitch of the note at the end of the fourth phrase of hymn tune *Würzburg* (Vaughan Williams 1933, hymn no. 127). With corpus 'A+B' being used, B♭3 and B3 return to the distributions and F5 is retained. The composed continuation, A4, is predicted with high probability by both distributions. This is again mostly due to the repeated musical material. Apart from the prediction of G4 by the LTM+, no other prediction

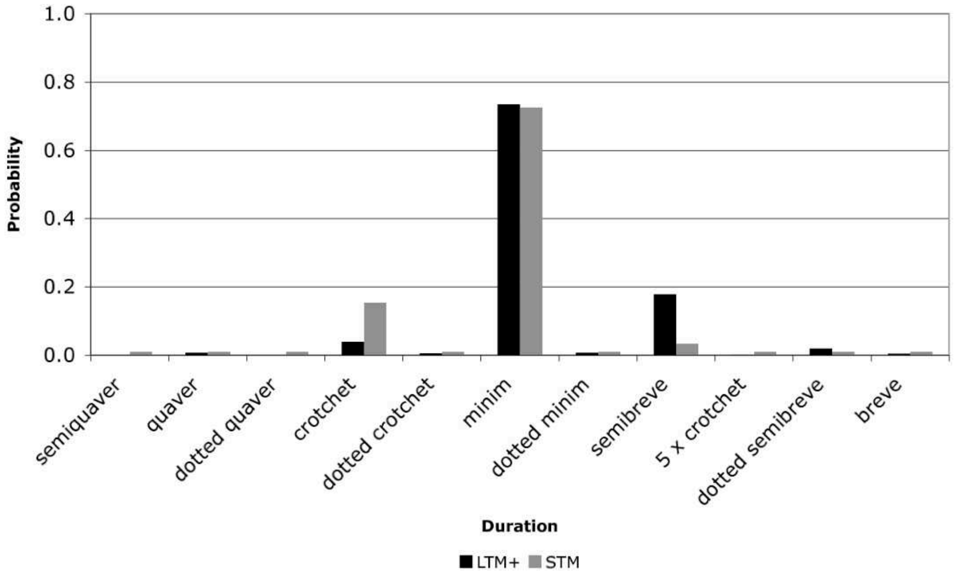


Figure 7.31: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint `Duration` ⊗ `ScaleDegree`, predicting the duration of the note at the beginning of the fifth phrase of hymn tune *Würzburg* (Vaughan Williams 1933, hymn no. 127).

is considered particularly likely. Although it seems rather odd that the LTM+ assigns F♯4 such a low probability, it is conceivable that a semibreve leading note has not been seen in the corpus.

Figure 7.33 shows such distributions for the beginning of the fifth phrase of the hymn tune. D4 and D5 are the most likely predictions according to both distributions, once again largely due to the repeat, where a minim D4 occurs at the beginning of the third phrase after an identical context. This is yet another viewpoint which could be improved by replacing `ScaleDegree` with a similar viewpoint able to distinguish between octaves. Both distributions consider G4 to be fairly likely; in the case of the STM, this must be solely due to the fact that minim G4 occurs eight times within the first four phrases. Similarly, F♯4 occurs six times, according it only a slightly lower STM probability. The composed continuation is A4, which has much lower prediction probabilities than those previously mentioned, but considerably higher than most of the rest of the distributions. For a direct comparison with somewhat similar viewpoint `DurRatio` ⊗ `Interval` (*i.e.*, predicting the same notes), see Figures 7.25 to 7.28.

7.9.2 `IntFirstInPhrase` ⊗ `ScaleDegree`

The fourth viewpoint to be selected for BOTH+ (Corpus ‘A+B’) is `IntFirstInPhrase` ⊗ `ScaleDegree` (also chosen for corpora ‘A’ and ‘B’). The set of likely intervals from first in phrase changes according to the starting note; for example, a descending interval of one tone is more likely if a phrase starts on the dominant rather than the tonic (since in the latter case a note outside of the major scale would result). The problem with

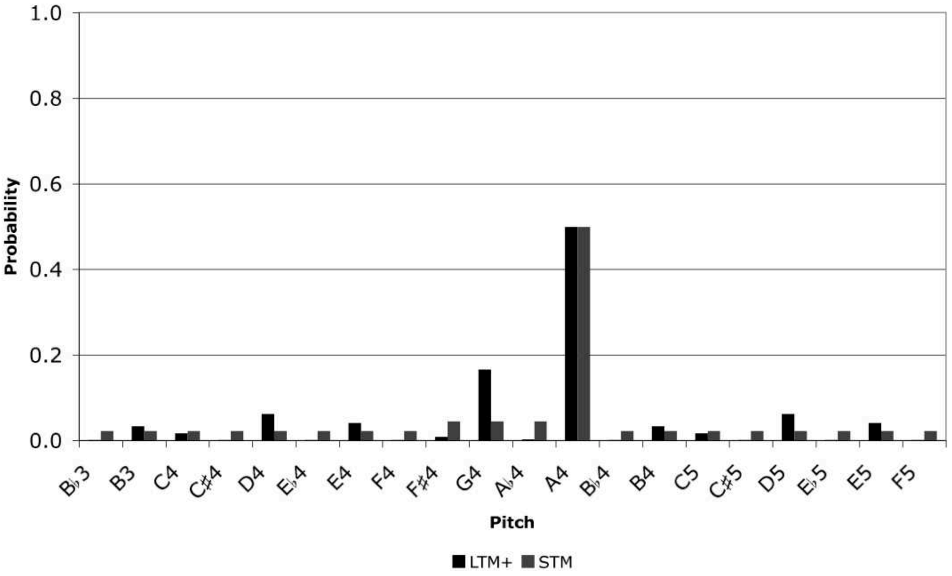


Figure 7.32: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint `Duration` ⊗ `ScaleDegree`, predicting the pitch of the note at the end of the fourth phrase of hymn tune *Würzburg* (Vaughan Williams 1933, hymn no. 127).

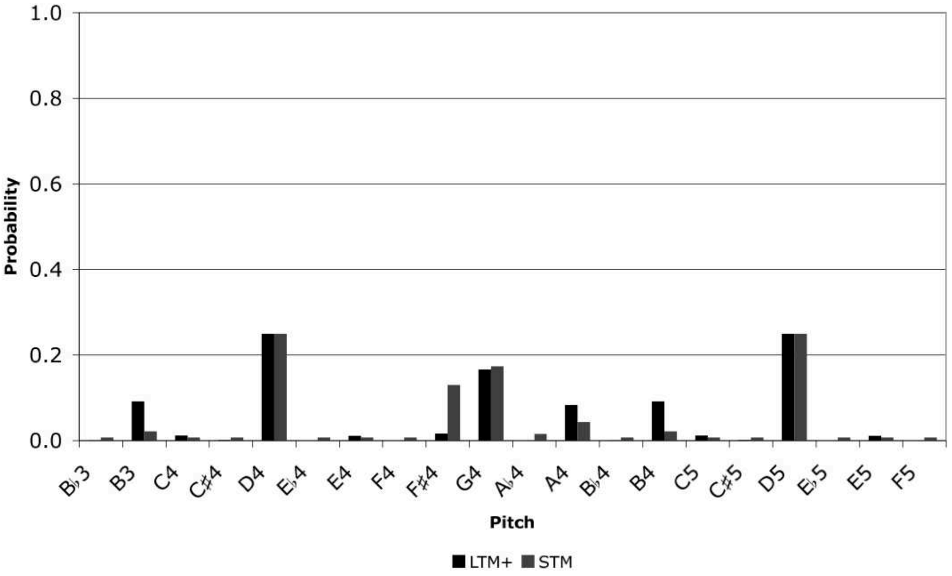


Figure 7.33: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint `Duration` ⊗ `ScaleDegree`, predicting the pitch of the note at the beginning of the fifth phrase of hymn tune *Würzburg* (Vaughan Williams 1933, hymn no. 127).

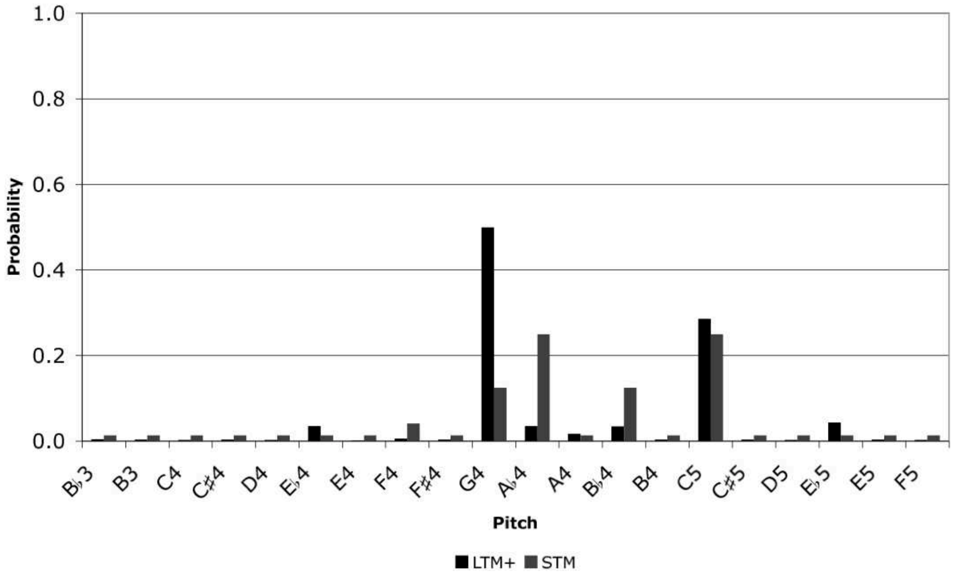


Figure 7.34: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint `IntFirstInPhrase` ⊗ `ScaleDegree`, predicting the pitch of the fifth note of the final phrase of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36).

`IntFirstInPhrase` on its own is that there is no way of knowing the degree of the scale on which a phrase begins. By linking `IntFirstInPhrase` with `ScaleDegree` the starting degree of scale is effectively known, meaning that separate statistics can be gathered and used for each different starting degree.

Figure 7.34 shows LTM+ and STM prediction probability distributions for viewpoint `IntFirstInPhrase` ⊗ `ScaleDegree`, after conversion to distributions over pitches, predicting the pitch of the fifth note of the final phrase of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36). The melody is in the key of E♭ major, and the first four notes of the final phrase are B♭4, G4, A♭4, B♭4. The LTM+ distribution shows G4 and C5 as by far the most likely predictions, while the STM reinforces the C5 prediction in particular. The composed continuation is A♭4, which is considered likely by the STM but not the LTM+. A♭4 and C5 are singled out by the STM because the previous phrase also starts on a B♭4, and contains B♭4s followed by both A♭4 and C5. All of the notes with high probabilities are a minor third or less from the previous note and appear in the scale of E♭ major.

7.10 Summary and Conclusion

In this chapter we examined version 0 (melodic) multiple viewpoint systems and speculated on why certain viewpoints had been selected from a music theoretic point of view. Musical regularities and other factors relevant to viewpoints performing well are briefly summarised below for a selection of the viewpoints analysed.

On its own `ScaleDegree` is a good viewpoint, as it effectively learns which degrees of the chromatic scale are present in the major scale (*i.e.*, those occurring most frequently in the corpus). Linking viewpoints such as `Tessitura` and `Interval` with `ScaleDegree` endows an ability (albeit imperfect) to differentiate between octaves, which improves performance. Viewpoint `InScale` is a good substitute for `ScaleDegree` in the STM.

There are strong metrical regularities relating to both note length and pitch, as evidenced by the performance of `Duration` $\otimes$ `Metre`, `Duration` $\otimes$ `TactusPositionInBar`, `DurRatio` $\otimes$ `TactusPositionInBar`, `ScaleDegree` $\otimes$ `Metre` and `Interval` $\otimes$ `TactusPositionInBar`. There are also regularities at phrase boundaries. Viewpoints which perform particularly well in this respect are `DurRatio` $\otimes$ `Phrase`, `ScaleDegree` $\otimes$ `Phrase`, `ScaleDegree` $\otimes$ `Piece`, `Interval` $\otimes$ `Phrase`, `Interval` $\otimes$ `LastInPhrase`, `Duration` $\otimes$ (`ScaleDegree` $\ominus$ `FirstIn-Phrase`) and `Interval` $\otimes$ `FirstInPhrase`. There is reason to believe that viewpoints `DurRatio` $\otimes$ `Interval` and `Interval` $\otimes$ `FirstInBar` may be able to model what is happening at phrase boundaries (although not exclusively), using `DurRatio` as a proxy for `Phrase` and `FirstInBar` as a proxy for `FirstInPhrase`.

Of the viewpoints analysed here to uncover the cause of their exceptional performance in the prediction of melody, the vast majority are new to this research. We have seen many instances where predictions have been in line with intuitive or music theoretic expectations, leading to the conclusion that the selected viewpoints are doing a good job of finding regularities of various kinds in the corpus and in individual pieces.

Chapter 8

Analysis of Selected Version 1 to 3 Viewpoints

8.1 Introduction

In Chapter 7, viewpoints selected for melodic modelling were analysed from a music theoretic perspective. In this chapter we investigate how viewpoints may perform the same roles in harmonic modelling, how their roles develop with respect to harmony and the extent to which viewpoints are chosen specifically for harmonic modelling. Examination of version 3 viewpoints will reveal the benefits of linking more than two primitive viewpoints together. Please note that scores of the harmonisations referred to in this chapter are included in Appendix D for convenient reference.

In §8.2 we look at version 1 BOTH+ multiple viewpoint systems and speculate on why particular viewpoints might have been selected. In §8.3 we inspect version 2 BOTH+ systems and try to discover why particular viewpoints were selected for particular tasks. In §8.4 we analyse version 3 BOTH+ systems and comment upon the way that certain inter-layer linked viewpoints have evolved in the viewpoint selection process from a music theoretic point of view. Finally, in §8.5, the chapter is summarised and a conclusion given.

8.2 Version 1

In this section we examine version 1 multiple viewpoint systems and speculate on why particular viewpoints might have been selected from a music theoretic point of view. The first three viewpoints to be selected for each multiple viewpoint system are discussed in detail on their first appearance.

Multiple viewpoint system	x-ent.
$\{(\mathtt{Duration})_{SATB}\}$	0.99
$+\ (\mathtt{DurRatio} \otimes \mathtt{Phrase})_{SATB}$	0.87
$+\ (\mathtt{Duration} \otimes \mathtt{PositionInBar})_{SATB}$	0.80
$-\ (\mathtt{Duration})_{SATB}$	0.77
$+\ (\mathtt{Duration} \otimes \mathtt{LastInPhrase})_{SATB}$	0.74
$+\ (\mathtt{DurRatio} \otimes (\mathtt{IOI} \ominus \mathtt{FirstInBar}))_{SATB}$	0.73
$+\ (\mathtt{DurRatio} \otimes \mathtt{TactusPositionInBar})_{SATB}$	0.73
$+\ (\mathtt{DurRatio} \otimes (\mathtt{ScaleDegree} \ominus \mathtt{LastInPhrase}))_{SATB}$	0.72

Table 8.1: Cross-entropies (*x-ent.*, bits/prediction) up to and including the sixth round of viewpoint addition/deletion during viewpoint selection of the best version 1 multiple viewpoint system for the prediction of `Duration` only (corpus ‘A’).

8.2.1 Prediction of `Duration` Alone

The first six viewpoints selected for the best multiple viewpoint system are presented in order of selection in Table 8.1, along with the deletion of $(\mathtt{Duration})_{SATB}$. Cross-entropies are shown at each stage. $\mathtt{DurRatio} \otimes \mathtt{Phrase}$ (see §7.2) and $\mathtt{Duration} \otimes \mathtt{LastInPhrase}$ (see §7.3) have already been examined in detail in relation to the modelling of melody, and will not be discussed further here. The last three viewpoints are only included for purposes of comparison with other versions.

$(\mathtt{Duration} \otimes \mathtt{PositionInBar})_{SATB}$ This is the second viewpoint to be selected for this system. We have already examined correlations between `Duration` and `Metre` (see §7.2) and `DurRatio` and `TactusPositionInBar` (see §7.3), where there is a definite metrical relationship. With $(\mathtt{Duration} \otimes \mathtt{PositionInBar})_{SATB}$ there is no predefined notion of metrical equivalence or even of where the tactus beats are; because of the preponderance of $\frac{4}{2}$ time in corpus ‘A’, however, this viewpoint is able to infer metrical structure. It is unlikely that $(\mathtt{Duration} \otimes \mathtt{PositionInBar})_{SATB}$ would be as effective with corpora which are more heterogenous with respect to time signature (although this is not a foregone conclusion, since useful statistics can be built up in the STM, as we shall see shortly).

Figure 8.1 shows LTM+ prediction probability distributions for viewpoint $(\mathtt{Duration} \otimes \mathtt{PositionInBar})_{SATB}$, predicting `Duration` on the first and second beats of the final bar of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33). We see that a minim is deemed considerably more likely than a crotchet on the first beat of the bar, whereas the opposite is slightly more likely on the second. On the other hand, the STM distribution of Figure 8.2 shows a crotchet to be by far the more probable on both beats, which tallies with the crotchet movement in the alto in the hymnal harmonisation.

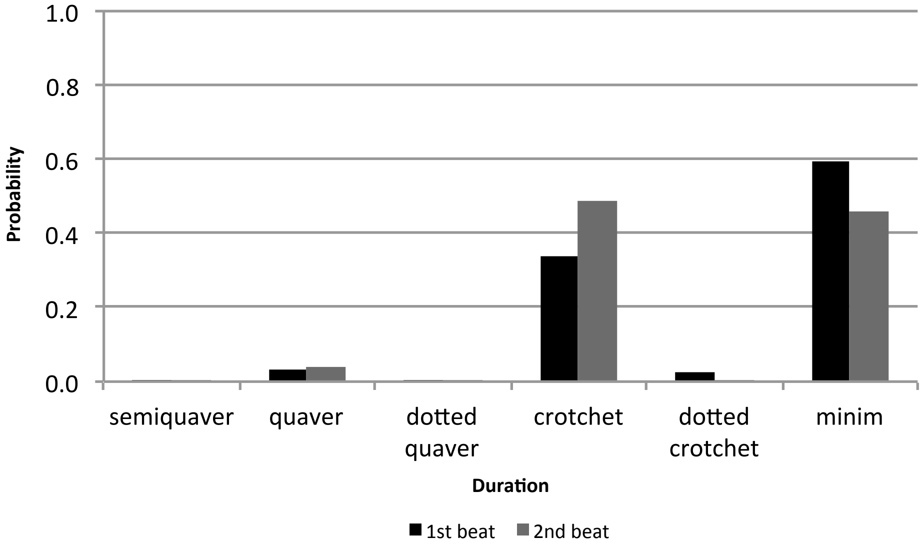


Figure 8.1: Bar chart showing LTM+ prediction probability distributions for viewpoint ($\texttt{Duration} \otimes \texttt{PositionInBar})_{SATB}$, predicting `Duration` on the first and second beats of the final bar of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33).

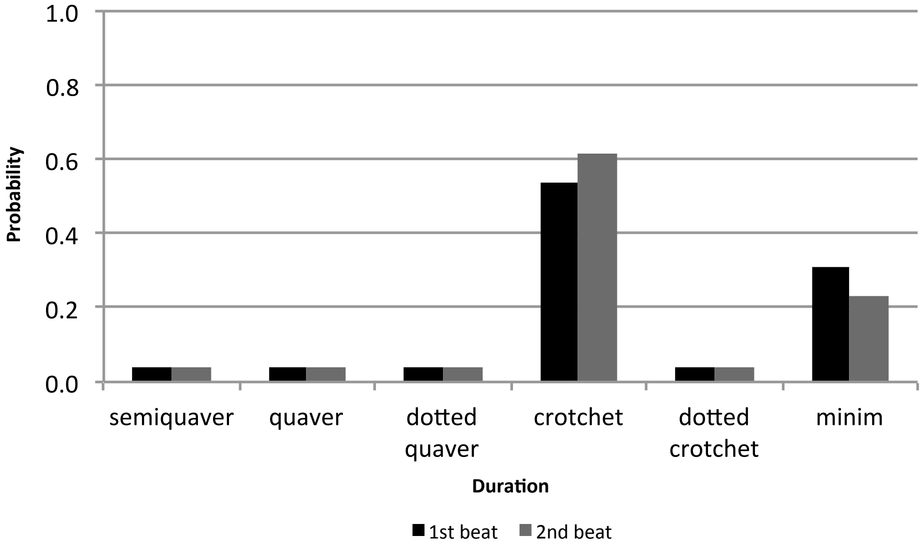


Figure 8.2: Bar chart showing STM prediction probability distributions for viewpoint ($\texttt{Duration} \otimes \texttt{PositionInBar})_{SATB}$, predicting `Duration` on the first and second beats of the final bar of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33).

Multiple viewpoint system	x-ent.
$\{(\texttt{Cont})_{SATB}\}$	0.91
$+$ $(\texttt{Cont} \otimes \texttt{Interval})_{SATB}$	0.80
$+$ $(\texttt{Cont} \otimes \texttt{Metre})_{SATB}$	0.74
$-$ $(\texttt{Cont})_{SATB}$	0.66
$+$ $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{SATB}$	0.64

Table 8.2: Cross-entropies (*x-ent.*, bits/prediction) up to and including the third round of viewpoint addition/deletion during viewpoint selection of the best version 1 multiple viewpoint system for the prediction of Cont only (seen Pitch domain, corpus 'A').

8.2.2 Prediction of Cont Alone

The first three viewpoints selected for the best multiple viewpoint system are presented in order of selection in Table 8.2, along with the deletion of $(\texttt{Cont})_{SATB}$. Cross-entropies are shown at each stage. In this instance, viewpoint selection was carried out using the seen Pitch domain; but the probability distribution charts below result from the use of the augmented domain.

$(\texttt{Cont} \otimes \texttt{Interval})_{SATB}$ If pairs of chords in different parts of the corpus have the same intervals in corresponding parts, it is fairly likely (although by no means certain) that the pairs of chords are the same, relative to the tonic. Statistics on Cont can then be usefully gathered for the purpose of prediction. It must be reiterated that, for any chord, Cont is predicted before Pitch; therefore Interval information is only available in the context. Let us assume that we are predicting Cont in the penultimate chord (after expansion) of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33). The immediately preceding SATB Interval values are -2, 0, -2 and 0, and the corresponding Cont values are all F. Using only this information (since an $\hbar$ of 1 is used here), we see from Figure 8.3 that the LTM+ assigns the highest probability to $T\ F\ T\ T$. This means that the soprano, tenor and bass notes are continued while there is a newly sounded alto note, corresponding with the harmony in the hymnal. The viewpoint model has effectively detected an appoggiatura in the alto, and treated it appropriately. Having said that, both the LTM+ and STM suggest a high likelihood for bass movement only, which is not appropriate given the appoggiatura. In the case of the STM (and probably also the LTM+), this is due to order 0 statistics. There is good agreement between the models as to which Cont combinations are particularly unlikely.

$(\texttt{Cont} \otimes \texttt{Metre})_{SATB}$ The choice of this viewpoint so early in the viewpoint selection process suggests a strong correlation between Cont and metrical structure, which, because of the relative abundance of such statistics, is capable of being deduced (to some extent) within the confines of a single piece. Figure 8.4 shows LTM+ and STM prediction probability distributions for viewpoint $(\texttt{Cont} \otimes \texttt{Metre})_{SATB}$, predicting Cont on the

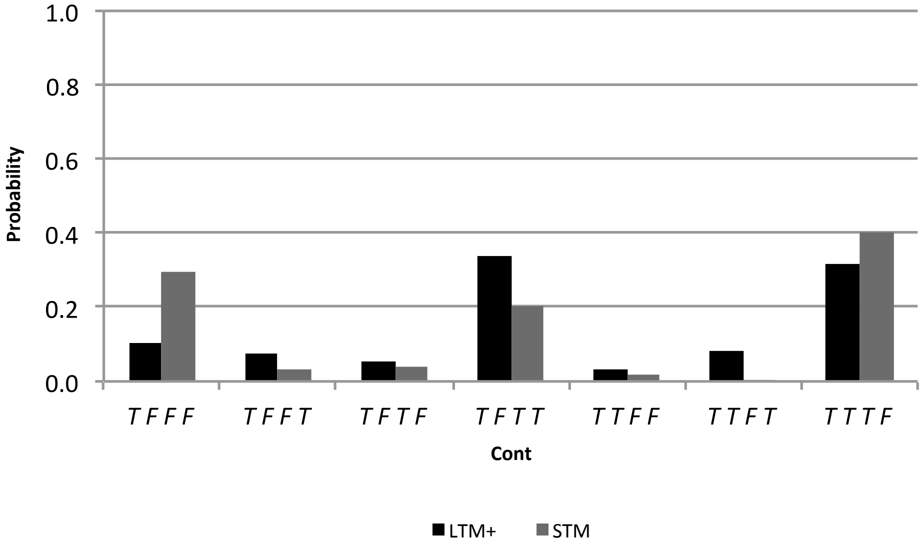


Figure 8.3: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint (`Cont` ⊗ `Interval`)$_{SATB}$, predicting `Cont` on the penultimate chord (after expansion) of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33).

penultimate chord (after expansion) of hymn tune *St. Edmund* (Vaughan Williams 1933, hymn no. 47). The `Cont` value corresponding with the hymnal harmony, $T\ F\ T\ T$, is assigned a probability of about 0.93 by the LTM+, supported by an STM probability of 0.50. This is a similar case to that described in the discussion of (`Cont`⊗`Interval`)$_{SATB}$ above, except that on the first half of the beat the alto note continues to sound and is concordant with all of the newly sounded notes. On this occasion, the alto movement on the second half of the beat is a passing note. It must be stated that in this particular case, (`Cont`)$_{SATB}$ is equally effective.

(`Cont` ⊗ `TactusPositionInBar`)$_{SATB}$ This viewpoint is similar in nature to (`Cont` ⊗ `Metre`)$_{SATB}$. In this case, however, there is no pre-ordained notion of metrical equivalence within and between time signatures: any such equivalence is inferred from the statistics gathered from the corpus. Predicting the same chord as above (see again Figure 8.4), we find the main difference to be a reduction from 0.33 to 0.25 in the probability assigned by the STM to `Cont` value $T\ T\ F\ T$.

8.2.3 Prediction of `Pitch` Alone

The first three viewpoints selected for the best multiple viewpoint system are presented in order of selection in Table 8.3, along with the deletion of (`Pitch`)$_{SATB}$. Cross-entropies are shown at each stage.

(`Cont`⊗`ScaleDegree`)$_{SATB}$ Since `Cont` is predicted before `Pitch`, `Cont` information is available in the context and the prediction. Let us assume that we are predicting `Pitch`

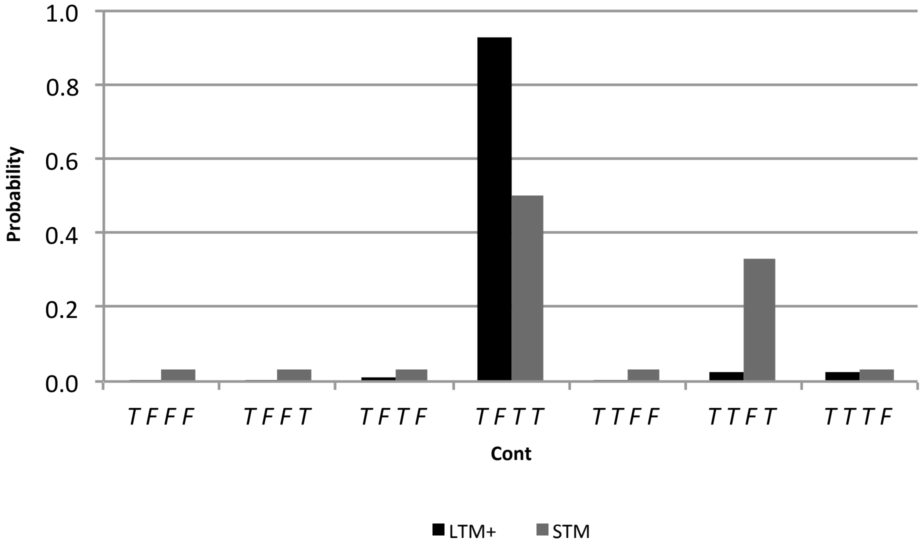


Figure 8.4: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint $(\texttt{Cont} \otimes \texttt{Metre})_{SATB}$, predicting `Cont` on the penultimate chord (after expansion) of hymn tune *St. Edmund* (Vaughan Williams 1933, hymn no. 47).

Multiple viewpoint system	x-ent.
$\{(\texttt{Pitch})_{SATB}\}$	5.07
$+\ (\texttt{Cont} \otimes \texttt{ScaleDegree})_{SATB}$	4.41
$+\ (\texttt{Cont} \otimes \texttt{Interval})_{SATB}$	4.09
$-\ (\texttt{Pitch})_{SATB}$	4.05
$+\ (\texttt{ScaleDegree} \otimes \texttt{LastInPhrase})_{SATB}$	3.99

Table 8.3: Cross-entropies (*x-ent.*, bits/prediction) up to and including the third round of viewpoint addition/deletion during viewpoint selection of the best version 1 multiple viewpoint system for the prediction of `Pitch` only (corpus ‘A’).

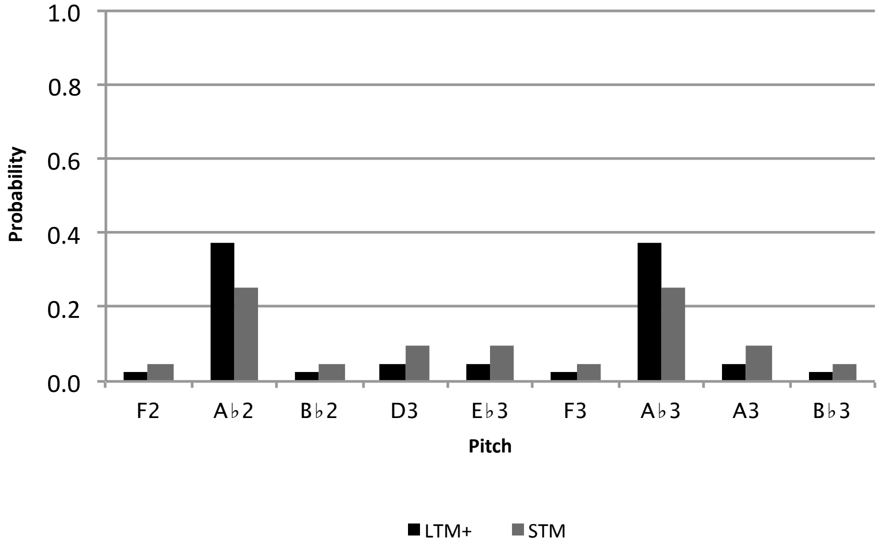


Figure 8.5: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint $(\texttt{Cont} \otimes \texttt{ScaleDegree})_{SATB}$, effectively predicting `Pitch` in the bass of the final chord (after expansion) of the tenth bar of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33).

in the final chord (after expansion) of the tenth bar of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33). Despite using an $\hbar$ of 2 here, let us concentrate on the chord immediately preceding the prediction. The SATB notes are F4, D4, B♭3 and B♭3 respectively, and the corresponding `Cont` values are all F. The SATB `Cont` values of the prediction are $T\ T\ T\ F$, which constrains the choice of chord to F4, D4 and B♭3 in the S, A and T respectively; we are therefore effectively only predicting the bass note. Figure 8.5 shows that both the LTM+ and STM assign the highest probabilities to A♭2 and A♭3. In reality, a step from B♭3 to A♭3 is far more likely than a leap of over an octave, and indeed an A♭3 appears in the hymnal. This is yet another case (although the first we have seen in the context of harmonic modelling) in which a viewpoint similar to `ScaleDegree`, but able to distinguish between octaves (see §7.2), would be highly beneficial.

$(\texttt{Cont} \otimes \texttt{Interval})_{SATB}$ We have examined this viewpoint in relation to the prediction of `Cont` above, but now its utility with regard to the prediction of `Pitch` must be investigated. In pursuance of this, it would be instructive to continue the example started in §8.2.2. Following an appoggiatura in the alto, the `Cont` configuration of the penultimate chord (after expansion) of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33) is $T\ F\ T\ T$, which affords the opportunity for resolution in the alto part. Resolution of an appoggiatura always proceeds by step, in this case resulting in a D4; but can this viewpoint model the resolution successfully? Figure 8.6 reveals that it is reasonably successful, inasmuch as D♭4 and D4 are considered highly probable: both

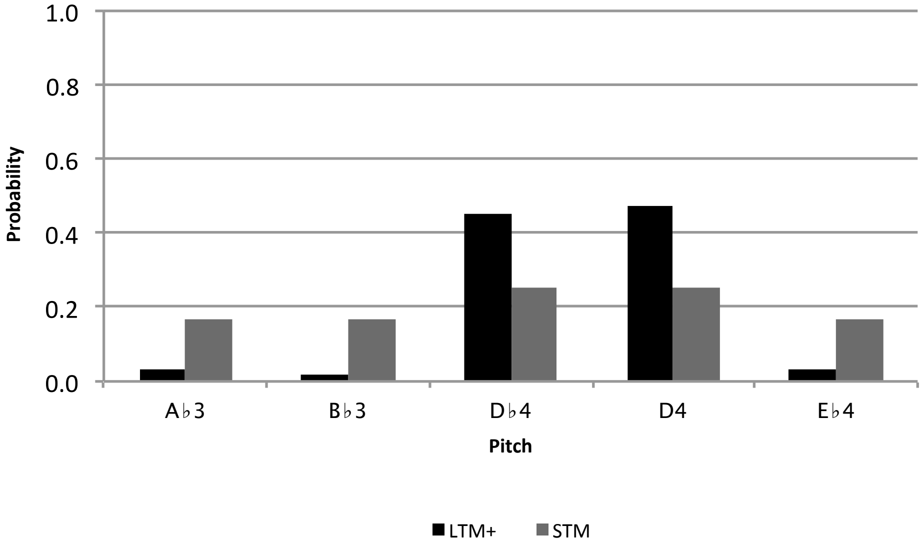


Figure 8.6: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint (`Cont` ⊗ `Interval`)$_{SATB}$, effectively predicting `Pitch` in the alto of the penultimate chord (after expansion) of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33).

are reached by step from the E♭4 appoggiatura. The problem is that this viewpoint is blind to key, with D♭4 being completely legitimate in A♭ major. The fact that stepwise predictions are so much more probable than all other predictions (in the LTM+, at least) is very encouraging.

(`ScaleDegree`⊗`LastInPhrase`)$_{SATB}$ Similar viewpoint `ScaleDegree`⊗`Phrase` proved highly effective in the prediction of melody (see §7.3). Here, we examine why the link with `LastInPhrase` is particularly useful for modelling harmony. Quite simply, the chords most commonly used at phrase endings are the tonic and dominant (usually in root position), resulting from the use of perfect, imperfect and plagal cadences in the home key; perfect cadences after a modulation to the dominant; and imperfect cadences after modulation to the subdominant. Less common are the dominant of the dominant; the tonic and dominant of the relative minor; and the subdominant.

There is a very straightforward perfect cadence at the conclusion of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37), with the soprano finishing on the tonic. The final chord used in the hymnal has an LTM+ prediction probability of 0.667, which is by far the highest out of a very large prediction set. Similarly, the chord at the end of the second phrase in the hymnal, the result of a straightforward imperfect cadence in the home key, is given a probability of 0.450; we can therefore begin to see why this viewpoint is able to reduce the overall cross-entropy. Unsurprisingly, this viewpoint can perform quite badly in connection with more unusual phrase endings. A case in point is a perfect cadence in the relative minor at the end of the twelfth bar of hymn tune *St.*

Edmund (Vaughan Williams 1933, hymn no. 47). The concluding E minor chord, as it appears in the hymnal, is assigned a probability of only 0.001 by the LTM+.

8.3 Version 2

In this section we examine viewpoints selected using version 2, as far as possible highlighting differences between prediction stages from a music theoretic point of view.

8.3.1 Prediction of `Duration` Alone

8.3.1.1 Prediction of Bass Given Soprano

The first six viewpoints selected for the best multiple viewpoint system are presented in order of selection in Table 8.4, along with the deletion of $(\texttt{Duration})_{SB}$. Cross-entropies are shown at each stage. Since the first three are the same viewpoints as were selected for version 1, with exactly the same cross-entropies occurring at each addition and deletion, they will not be further examined individually here. Suffice it to say that the cross-entropies are the same because primitive viewpoints applying to sequence positions are employed, rather than viewpoints directly associated with the soprano and bass (such as those derived from `Pitch`). It is reasonable to expect that if viewpoints relating to sequence position perform well in one version, they will also be effective in others. $\texttt{DurRatio} \otimes \texttt{TactusPositionInBar}$ was thoroughly investigated in §7.3, and so will also not be discussed in detail here.

It is not until the fifth round of viewpoint addition that a difference occurs: $(\texttt{DurRatio} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{SB}$ as opposed to $(\texttt{DurRatio} \otimes \texttt{TactusPositionInBar})_{SATB}$. The other of these viewpoints is added next in each case. At first sight, it seems that the less sparse `Pitch` derived data of the first stage of version 2 prediction allows `Pitch` derived viewpoints to be a little more effective in the prediction of `Duration`. This would be a misreading of the facts, however. In both cases, an $\hbar$ of 0 means that there is no possibility of a `Pitch` derived viewpoint in the context; in addition `Duration` is predicted before `Pitch`, which means that `Pitch` derived viewpoints can only come into play inasmuch as the given soprano note modifies the prediction. Since this note is the same in both cases, it is hard to see why there should be any difference between the versions at all. The difference is small enough to be of no real concern in relation to the analysis in Chapter 6; but it will be investigated as a matter of urgency to ensure complete consistency in future work.

$(\texttt{DurRatio} \otimes (\texttt{IOI} \ominus \texttt{FirstInBar}))_{SB}$ In spite of the fact that this viewpoint reduces the overall cross-entropy a little, the implementation of $\texttt{IOI} \ominus \texttt{FirstInBar}$ is known to be faulty (see §7.1). In view of this, there is little to be gained by studying its probability distributions. Properly implemented, it could be particularly good at capturing regularities within single pieces, as exemplified by hymn tune *Innocents* (Vaughan Williams

Multiple viewpoint system	x-ent.
$\{$(`Duration`)$_{SB}\}$	0.99
+ (`DurRatio` ⊗ `Phrase`)$_{SB}$	0.87
+ (`Duration` ⊗ `PositionInBar`)$_{SB}$	0.80
− (`Duration`)$_{SB}$	0.77
+ (`Duration` ⊗ `LastInPhrase`)$_{SB}$	0.74
+ (`DurRatio` ⊗ (`IOI` ⊖ `FirstInBar`))$_{SB}$	0.73
+ (`DurRatio` ⊗ `ScaleDegree` ⊖ `LastInPhrase`)$_{SB}$	0.73
+ (`DurRatio` ⊗ `TactusPositionInBar`)$_{SB}$	0.72

Table 8.4: Cross-entropies (*x-ent.*, bits/prediction) up to and including the sixth round of viewpoint addition/deletion during viewpoint selection of the best version 2 multiple viewpoint system for the prediction of `Duration` in the bass given soprano (corpus ‘A’).

1933, hymn no. 37). The length of the chord on the first beat of a bar is exactly twice the length of its preceding chord on seven out of eight occasions (after expansion). The STM prediction probability for a `DurRatio` value of 2 would therefore be very high by the end of the piece.

(`DurRatio` ⊗ (`ScaleDegree` ⊖ `LastInPhrase`))$_{SB}$ With no context ($\hbar = 0$) and with `Duration` being predicted before `Pitch`, this viewpoint is effectively (`DurRatio` ⊖ `LastInPhrase`)$_{SB}$ except for the contribution of the given soprano note and the fact that `DurRatio` retains its local meaning. This is similar to (`DurRatio` ⊗ `Phrase`)$_S$ (which performed well in melodic models), except that it predicts only at the ends of phrases (in addition to using the soprano note). Let us therefore compare it directly with (`DurRatio`⊗`Phrase`)$_{SB}$. Figure 8.7 shows such a comparison for STM prediction probability distributions, predicting `Duration` at the final chord of hymn tune *St. Edmund* (Vaughan Williams 1933, hymn no. 47). There is a huge difference between the distributions, with (`DurRatio`⊗`Phrase`)$_{SB}$ and (`DurRatio`⊗(`ScaleDegree`⊖`LastInPhrase`))$_{SB}$ assigning the highest probability to a crotchet and a minim respectively; the hymnal chord length is a minim. Being not completely blind to `ScaleDegree`, the latter viewpoint was able to find regularities beyond the comprehension of (`DurRatio`⊗`Phrase`)$_{SB}$, which, in this particular case at least, proved beneficial. For the sake of completeness, both LTM+ distributions showed a crotchet to be most likely.

8.3.1.2 Prediction of Alto/Tenor Given Soprano/Bass

The first six viewpoints selected for the best multiple viewpoint system (again using an $\hbar$ of 0) are presented in order of selection in Table 8.5, along with the deletion of (`Duration`)$_{SATB}$. Cross-entropies are shown at each stage. The first three were also selected for version 1 and the first prediction stage of version 2, and so will not be further investigated here. In addition, since (`DurRatio` ⊗ (`ScaleDegree` ⊖ `FirstInBar`))$_{SATB}$ and (`Duration` ⊗ (`ScaleDegree` ⊖ `FirstInBar`))$_{SATB}$ are very similar viewpoints, only

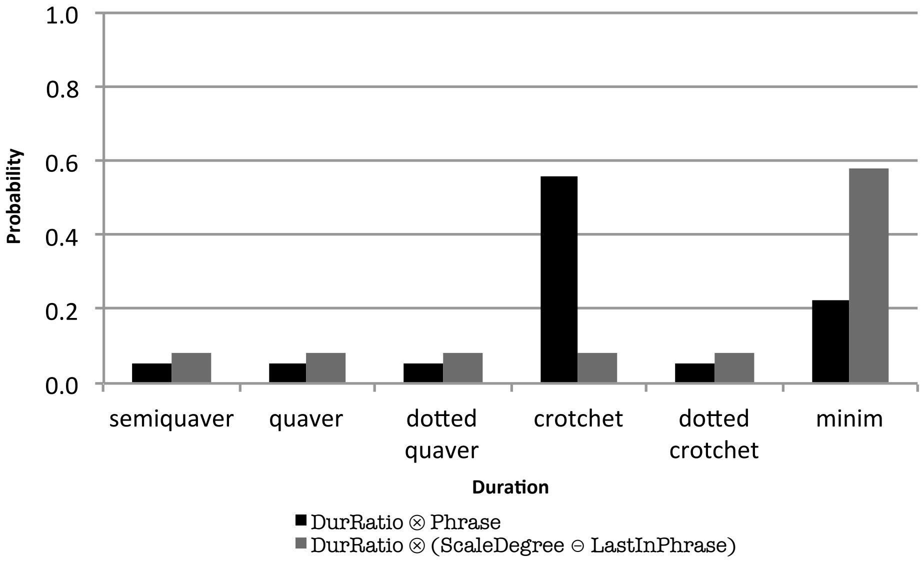


Figure 8.7: Bar chart showing STM prediction probability distributions for viewpoints $(\texttt{DurRatio} \otimes \texttt{Phrase})_{SB}$ and $(\texttt{DurRatio} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{SB}$, predicting `Duration` in the bass given soprano at the final chord of hymn tune *St. Edmund* (Vaughan Williams 1933, hymn no. 47).

the former will be examined in detail.

$(\texttt{DurRatio} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInBar}))_{SATB}$ Figure 8.8 shows LTM+ and STM prediction probability distributions for viewpoint $(\texttt{DurRatio} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInBar}))_{SATB}$, predicting `Duration` in the alto/tenor given soprano/bass on the first beat of the penultimate bar of hymn tune *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97). What is known about the chord being predicted at this stage is that the soprano and bass notes are minims, the soprano note is A4 (mediant) and the bass note F3 (tonic). The chord immediately preceding it is a semibreve. At first glance, it

Multiple viewpoint system	x-ent.
$\{(\texttt{Duration})_{SATB}\}$	0.84
$+$ $(\texttt{DurRatio} \otimes \texttt{Phrase})_{SATB}$	0.70
$+$ $(\texttt{Duration} \otimes \texttt{PositionInBar})_{SATB}$	0.62
$-$ $(\texttt{Duration})_{SATB}$	0.59
$+$ $(\texttt{Duration} \otimes \texttt{LastInPhrase})_{SATB}$	0.56
$+$ $(\texttt{DurRatio} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInBar}))_{SATB}$	0.55
$+$ $(\texttt{DurRatio} \otimes (\texttt{Interval} \ominus \texttt{FirstInBar})_{SATB}$	0.55
$+$ $(\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInBar}))_{SATB}$	0.54

Table 8.5: Cross-entropies (*x-ent.*, bits/prediction) up to and including the sixth round of viewpoint addition/deletion during viewpoint selection of the best version 2 multiple viewpoint system for the prediction of `Duration` in the alto/tenor given soprano/bass (seen `Pitch` domain, corpus ‘A’).

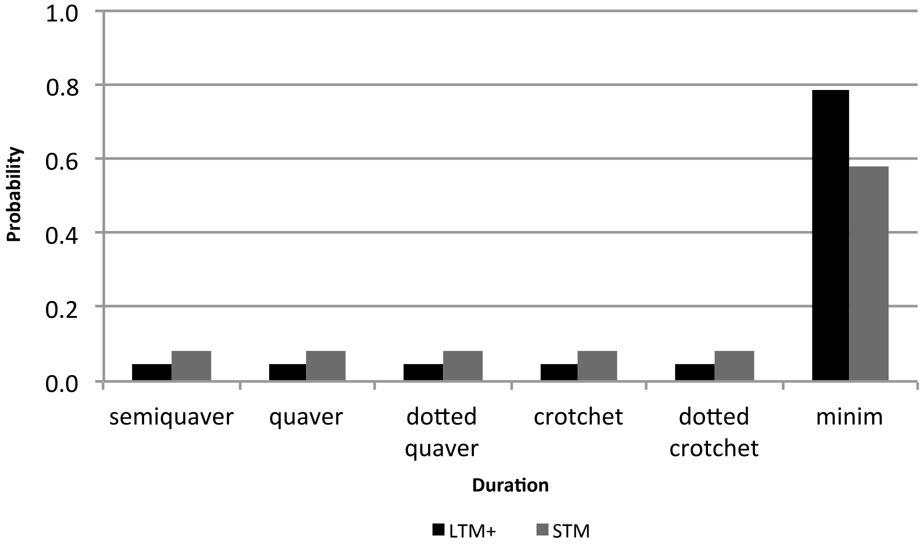


Figure 8.8: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint $(\texttt{DurRatio} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInBar}))_{SATB}$, predicting `Duration` in the alto/tenor given soprano/bass on the first beat of the penultimate bar of hymn tune *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97).

seems extraordinary that so much of the probability mass is concentrated on a minim and that the rest of the distribution (in each case) is uniform. Inspection of the corpus makes this believable, however. It is rarely the case that a `DurRatio` value of less than 0.5 occurs on the first beat of a bar; the domain is restricted to `Duration` values seen in the corpus; the maximum `Duration` value in the distribution is equal to the length of the bass note (at this prediction stage) being harmonised, in this case a minim; and only `DurRatio` values associated with a mediant soprano and a tonic bass at the first in bar are admitted to the statistics for this particular distribution. These four factors conspire to produce a distribution of this shape, resulting in the hymnal note length being predicted with great certainty. Since this viewpoint was not selected at all for the prediction of bass given soprano, we can speculate that the contribution of the already predicted bass note, in combination with the given soprano note, is significant at this prediction stage.

$(\texttt{DurRatio} \otimes (\texttt{Interval} \ominus \texttt{FirstInBar}))_{SATB}$ In this viewpoint, `Interval` is between successive first in bars, whereas `DurRatio` retains its local meaning. It turns out that this viewpoint is completely useless at predicting `Duration` on the first beat of the penultimate bar of hymn tune *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97), as above, because both the LTM+ and STM distributions are uniform (`Interval` values of 0 and 3 in the soprano and bass respectively, in conjunction with a `DurRatio` value of 0.5 or less, have apparently not been previously seen). As for the bar before (bar twelve), we find that the LTM+ is able to make a prediction with considerable

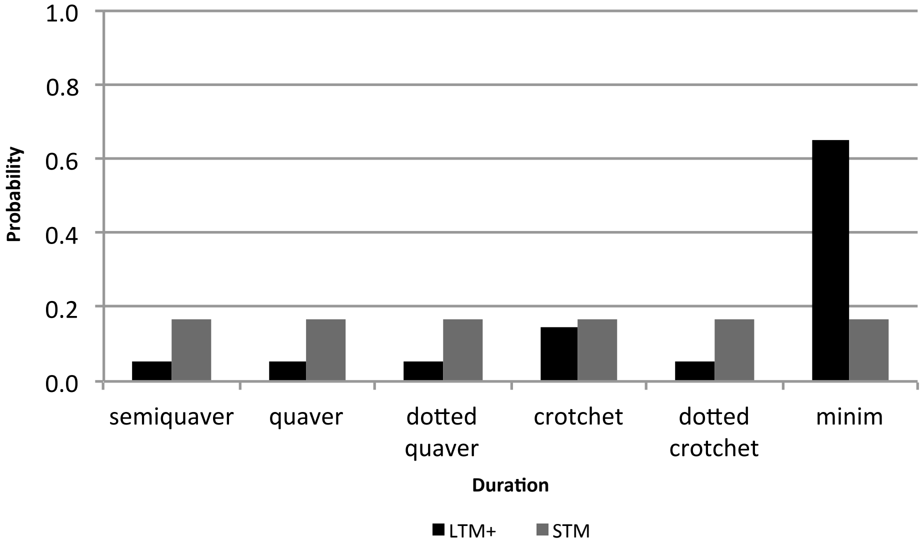


Figure 8.9: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint $(\texttt{DurRatio} \otimes (\texttt{Interval} \ominus \texttt{FirstInBar}))_{SATB}$, predicting `Duration` in the alto/tenor given soprano/bass on the first beat of the antepenultimate bar of hymn tune *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97).

certainty, as shown in Figure 8.9. In this case, the relevant soprano and bass intervals are 4 and 0 respectively, and the last chord of bar eleven is a minim. The chord being predicted is a minim in the hymnal. This viewpoint was chosen earlier in the selection process than it was for the system predicting bass given soprano, suggesting that it is more effective at the second stage of prediction. It appears that the addition of the bass interval is responsible for its improved performance.

8.3.2 Prediction of `Cont` Alone

8.3.2.1 Prediction of Bass Given Soprano

The three viewpoints selected for the best multiple viewpoint system (again using an $\hbar$ of 0) are presented in order of selection in Table 8.6, along with the deletion of $(\texttt{Cont})_{SB}$. Cross-entropies are shown at each stage. The first and last of these viewpoints have already been examined in detail in §8.2.2, and so will not be further discussed here.

$(\texttt{Cont} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInPhrase}))_{SB}$ All phrases begin with chords in which all notes are newly sounded (*i.e.*, they have `Cont` values of F). This being the case, a very high LTM+ probability is expected for F in the soprano and bass. At the beginning of the second phrase (bar 3) of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33), the LTM+ and STM probabilities for a `Cont` value of F F are 0.990 and 0.923 respectively. Although the LTM+ value is unsurprising, at first glance the STM value is astonishing given that only one previous first in phrase has occurred. The reason is that the domain comprises twelve compound values for a `Cont` of F (one for

Multiple viewpoint system	x-ent.
$\{(\texttt{Cont})_{SB}\}$	0.44
$+$ $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{SB}$	0.36
$-$ $(\texttt{Cont})_{SB}$	0.27
$+$ $(\texttt{Cont} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInPhrase}))_{SB}$	0.26
$+$ $(\texttt{Cont} \otimes \texttt{Metre})_{SB}$	0.26

Table 8.6: Cross-entropies (*x-ent.*, bits/prediction) up to and including the third and final round of viewpoint addition/deletion during viewpoint selection of the best version 2 multiple viewpoint system for the prediction of `Cont` in the bass given soprano (corpus 'A').

Multiple viewpoint system	x-ent.
$\{(\texttt{Cont})_{SATB}\}$	0.48
$+$ $(\texttt{Cont} \otimes \texttt{Metre})_{SATB}$	0.45
$-$ $(\texttt{Cont})_{SATB}$	0.42
$+$ $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{SATB}$	0.41
$+$ $(\texttt{Cont} \otimes \texttt{Interval})_{SATB}$	0.40
$+$ $(\texttt{Cont} \otimes \texttt{PositionInBar})_{SATB}$	0.40

Table 8.7: Cross-entropies (*x-ent.*, bits/prediction) up to and including the fourth and final round of viewpoint addition/deletion during viewpoint selection of the best version 2 multiple viewpoint system for the prediction of `Cont` in the alto/tenor given soprano/bass (corpus 'A').

each `ScaleDegree` value), but only one for a `Cont` of T (where a note continues). The distribution is uniform in this case, and so on conversion to a distribution over `Cont` the probability of the value F predominates. Arguably, this built-in predisposition towards the value F is what makes this a good viewpoint overall.

8.3.2.2 Prediction of Alto/Tenor Given Soprano/Bass

The four viewpoints selected for the best multiple viewpoint system (once again using an $\hbar$ of 0) are presented in order of selection in Table 8.7, along with the deletion of $(\texttt{Cont})_{SATB}$. Cross-entropies are shown at each stage. The first three of the added viewpoints have already been examined in §8.2.2, and so will not be further discussed in detail here.

$(\texttt{Cont} \otimes \texttt{PositionInBar})_{SATB}$ This viewpoint is defined in terms of the number of time units after the beginning of the bar (a semibreve is equal to 96 of these units), and therefore has no built-in knowledge of where the tactus beats are. When predicting the same chord as $(\texttt{Cont} \otimes \texttt{Metre})_{SATB}$ and $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{SATB}$ in §8.2.2, the LTM+ assigns `Cont` value T F T T (as in the hymnal) a probability of 0.333 and the STM 0.2. These are much lower probabilities than were assigned by the aforementioned

Multiple viewpoint system	x-ent.
$\{(\texttt{Pitch})_{SB}\}$	2.33
$+\ (\texttt{Interval} \otimes \texttt{InScale})_{SB}$	1.93
$+\ (\texttt{Cont} \otimes \texttt{ScaleDegree})_{SB}$	1.78
$+\ (\texttt{Interval} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_{SB}$	1.72

Table 8.8: Cross-entropies (*x-ent.*, bits/prediction) up to and including the third round of viewpoint addition/deletion during viewpoint selection of the best version 2 multiple viewpoint system for the prediction of Pitch in the bass given soprano (corpus ‘A’).

viewpoints, precisely because $(\texttt{Cont}\otimes\texttt{PositionInBar})_{SATB}$ is blind to tactus: the hymn in question has a time signature of $\frac{4}{4}$, which occurs far less frequently in the corpus than $\frac{4}{2}$. We would therefore expect this viewpoint to perform better on hymns with the latter time signature, which is borne out by the fact that a similar configuration at the same position in *Grafton* (Vaughan Williams 1933, hymn no. 33) is assigned a probability of 0.458 by the LTM+.

8.3.3 Prediction of Pitch Alone

8.3.3.1 Prediction of Bass Given Soprano

The first three viewpoints selected for the best multiple viewpoint system (using an $\hbar$ of 1) are presented in order of selection in Table 8.8. Cross-entropies are shown at each stage. The second of these viewpoints has already been examined in detail in §8.2.3, and so will not be further discussed here.

$(\texttt{Interval}\otimes\texttt{InScale})_{SB}$ This viewpoint has been thoroughly examined from the point of view of melodic modelling in §7.4. Intervals ending on notes belonging to the scale have relatively high probabilities, which makes this viewpoint particularly good at modelling how the soprano and bass lines move in relation to each other. Often, a step or repeated note in the soprano is accompanied by a leap in the bass, as in the vast majority of cadences. It is therefore particularly pertinent to the generation of a wellformed bass line.

Figure 8.10 shows LTM+ and STM prediction probability distributions for viewpoint $(\texttt{Interval} \otimes \texttt{InScale})_{SB}$, predicting Pitch in the bass given soprano in the last chord of the sixth bar of hymn tune *St. Edmund* (Vaughan Williams 1933, hymn no. 47). In the context, the soprano falls three semitones while the bass rises one; and in the prediction, the soprano falls a further two semitones. All three of these intervals end on a note of the scale of G major. We see that three pitches stand out in the distribution: G2, C3 and D3. In the hymnal the bass continues with G2 (completing a typical plagal cadence), which has a particularly high STM probability because bar 6 is identical to bar 2. D3 is a perfectly good alternative, completing an imperfect cadence. C3 is not

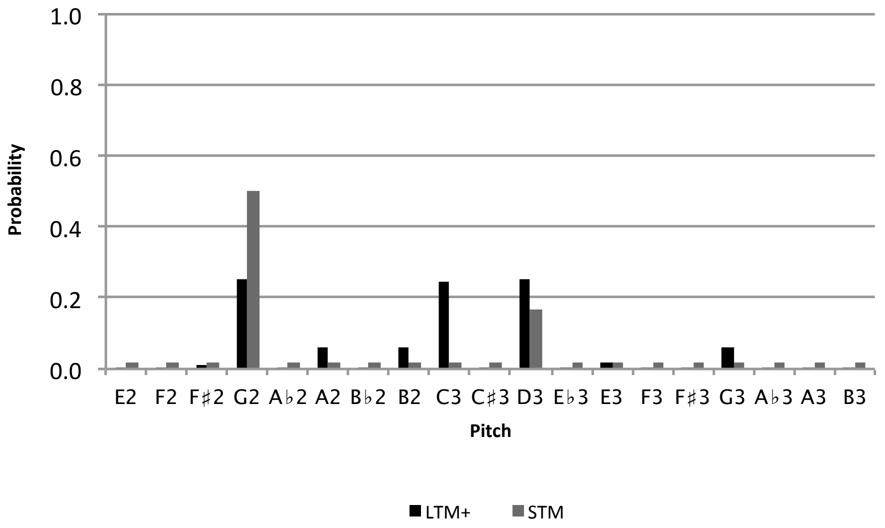


Figure 8.10: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint ($\texttt{Interval} \otimes \texttt{InScale})_{SB}$, predicting `Pitch` in the bass given soprano in the last chord of the sixth bar of hymn tune *St. Edmund* (Vaughan Williams 1933, hymn no. 47).

appropriate at this point; but the progression Ib–IV with a passing D3 in the soprano could occur elsewhere.

$(\texttt{Interval} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_{SB}$ Already investigated in relation to melodic modelling in §7.3, this viewpoint looks at `ScaleDegree` on tactus beats, which are structurally important. Moreover, the link with `Interval` enables it to distinguish between octaves. There is, then, a sense here of relative pitch between the soprano and bass which is missing from $(\texttt{Interval} \otimes \texttt{InScale})_{SB}$ above. On the other hand, there is no foreknowledge of which notes belong to the scale of the piece.

Figure 8.11 shows LTM+ and STM prediction probability distributions for viewpoint $(\texttt{Interval} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_{SB}$, predicting `Pitch` in the bass given soprano on the last beat of the eleventh bar of hymn tune *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97). The progression is A4 and F3 in the soprano and bass respectively, followed by G4 in the soprano. The bass continues with an E3 in the hymnal (I–Vb), corresponding to by far the highest probability in the STM distribution. In the corpus as a whole, C3 is the most likely prediction (I–V): this progression is often found at phrase endings, where it forms an imperfect cadence. B♭2 is also a plausible prediction, completing I–IIb or I–V^{7}d progressions.

8.3.3.2 Prediction of Alto/Tenor Given Soprano/Bass

The first three viewpoints selected for the best multiple viewpoint system (again using an $\hbar$ of 1) are presented in order of selection in Table 8.9. Cross-entropies are shown at

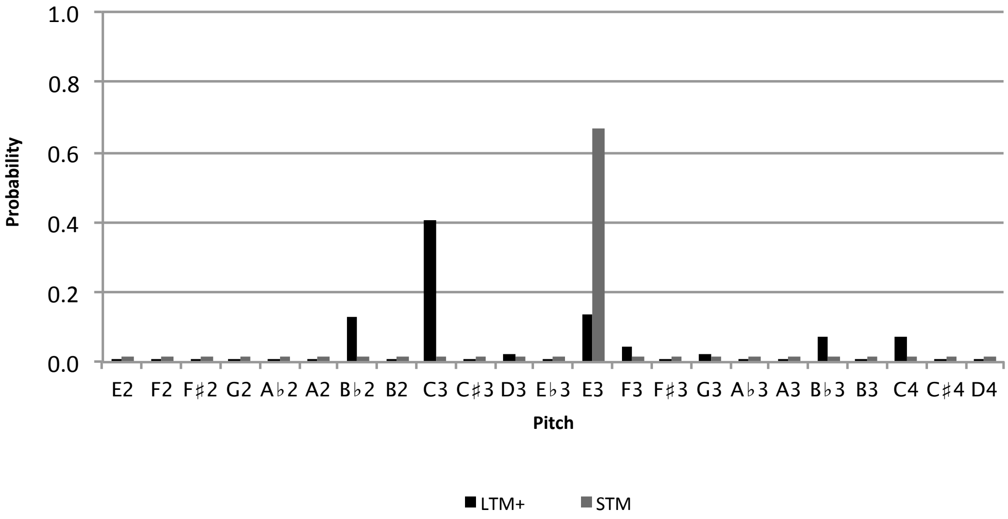


Figure 8.11: Bar chart showing LTM+ and STM prediction probability distributions for viewpoint $(\texttt{Interval} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_{SB}$, predicting `Pitch` in the bass given soprano on the last beat of the eleventh bar of hymn tune *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97).

Multiple viewpoint system	x-ent.
$\{(\texttt{Pitch})_{SATB}\}$	2.47
$+ (\texttt{Cont} \otimes \texttt{ScaleDegree})_{SATB}$	2.11
$+ (\texttt{Cont} \otimes \texttt{Interval})_{SATB}$	2.03
$+ (\texttt{ScaleDegree} \ominus \texttt{Tactus})_{SATB}$	2.00

Table 8.9: Cross-entropies (*x-ent.*, bits/prediction) up to and including the third round of viewpoint addition/deletion during viewpoint selection of the best version 2 multiple viewpoint system for the prediction of `Pitch` in the alto/tenor given soprano/bass (corpus ‘A’).

each stage. The first two of these viewpoints have already been examined in detail in §8.2.3, and so will not be further discussed here.

$(\texttt{ScaleDegree}\ominus\texttt{Tactus})_{SATB}$ This viewpoint is similar to $(\texttt{Interval}\otimes(\texttt{ScaleDegree}\ominus \texttt{Tactus}))_{SB}$ (see §8.3.3.1 above) except that it is unable to distinguish between octaves. One can speculate that this ability is not so important for the inner parts as, generally speaking, the alto is no more than an octave below the soprano and the tenor is no more than an octave below the alto. The final bar of *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97) begins with the chord A4, F4, C4, C3, continuing with G4 in the soprano and C3 in the bass. The tenor is known to have a `Cont` value of T, which means that the tenor stays on C4; we are therefore effectively only predicting the alto note. According to the distributions, the only possibilities are E♭4, E4 and F4. The alto note found in the hymnal, E4, is overwhelmingly the most likely prediction,

Multiple viewpoint system	x-ent.
$\{$(`Duration`)$_{ATB}\}$	0.99
$+$ (`Duration` $\otimes$ `Metre`)$_{ATB}$	0.90
$-$ (`Duration`)$_{ATB}$	0.85
$+$ (`LastInPhrase`)$_S$ $\otimes$ (`Duration` $\otimes$ `Metre`)$_{ATB}$	0.79
$+$ (`PositionInBar` $\otimes$ `LastInPhrase`)$_S$ $\otimes$ (`Duration` $\otimes$ `Metre`)$_{ATB}$	0.72
$-$ (`Duration` $\otimes$ `Metre`)$_{ATB}$	0.71
$-$ (`LastInPhrase`)$_S$ $\otimes$ (`Duration` $\otimes$ `Metre`)$_{ATB}$	0.74

Table 8.10: Cross-entropies (*x-ent.*, bits/prediction) are shown at each point in the evolution of the best performing version 3 viewpoint for the prediction of `Duration` in the alto/tenor/bass given soprano (corpus 'A'), assuming that no other viewpoints were added or deleted during viewpoint selection.

with LTM and STM probabilities of 0.933 and 0.667 respectively. This forms a cadential Ic–V progression, which is duly completed with a tonic chord in root position (after a passing note in the tenor). Another possibility would have been a suspended F4 in the alto part of the second chord; but apparently this rarely happens in the corpus.

8.4 Version 3

In this section we examine version 3 multiple viewpoint systems and speculate on the way that certain inter-layer linked viewpoints have evolved in the viewpoint selection process from a music theoretic point of view.

8.4.1 Prediction of `Duration` Alone

8.4.1.1 Prediction of Alto/Tenor/Bass Given Soprano

The best multiple viewpoint system uses an $\hbar$ of 0. Rather than tabulating viewpoints selected early in the viewpoint selection process as before, Table 8.10 presents additions and deletions relevant to the creation of the viewpoint which performs best in conjunction with (`Duration`)$_{ATB}$ (ensuring that the entire sequence can be predicted). Cross-entropies are shown at each stage. Please note that although the removal of (`LastInPhrase`)$_S$ $\otimes$ (`Duration` $\otimes$ `Metre`)$_{ATB}$ causes the cross-entropy to rise here, the deletion makes sense in the context of a full viewpoint selection.

Evolution of (`PositionInBar`$\otimes$`LastInPhrase`)$_S$ $\otimes$ (`Duration`$\otimes$`Metre`)$_{ATB}$ The first stage of the evolution of this viewpoint is the intra-layer linking of (`Duration`)$_{ATB}$ with `Metre` to form (`Duration`$\otimes$`Metre`)$_{ATB}$. The latter viewpoint has already been discussed in relation to melodic modelling in §7.2, and so will not be examined in detail here. Suffice it to say that there is a definite correlation between metrical position and note length; for example, there is a tendency for long notes to occur on the first beat of the bar (and the third beat of bars comprising four tactus beats). The picture is complicated somewhat in relation to harmony, however, because of its expansion (necessitated by passing notes, and so on).

Next, $(\texttt{LastInPhrase})_S \otimes (\texttt{Duration} \otimes \texttt{Metre})_{ATB}$ is created by adding LastInPhrase in the soprano. It is reasonable to suppose that knowing whether or not a chord is the last in its phrase in addition to its metrical position will result in more accurate probability distributions. For example, a long chord is even more likely to occur on the first or third beat if it is also the last in the phrase. Finally, PositionInBar is added to the soprano to form $(\texttt{PositionInBar} \otimes \texttt{LastInPhrase})_S \otimes (\texttt{Duration} \otimes \texttt{Metre})_{ATB}$. The combination of PositionInBar and Metre enables finer distinctions than are possible with Metre alone; for example, when there are four beats to the bar, the second and fourth beats (considered metrically equivalent) are now distinguishable by their PositionInBar values.

Figure 8.12 shows LTM+ prediction probability distributions for the three viewpoints, predicting Duration in the alto/tenor/bass given soprano on the second beat of the penultimate bar of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37). Since this chord is not at the end of a phrase, it is unsurprising that there is very little difference in the probabilities for $(\texttt{Duration} \otimes \texttt{Metre})_{ATB}$ and $(\texttt{LastInPhrase})_S \otimes (\texttt{Duration} \otimes \texttt{Metre})_{ATB}$. The addition of PositionInBar causes a large transfer of probability mass from crotchet to quaver. It would appear, then, that sub-tactus durations are more likely on the second beat than on the metrically equivalent fourth. A crotchet is still more probable overall, however, as appears in the hymnal at this point.

STM prediction probability distributions for the same three viewpoints are shown in Figure 8.13. The $(\texttt{Duration} \otimes \texttt{Metre})_{ATB}$ and $(\texttt{LastInPhrase})_S \otimes (\texttt{Duration} \otimes \texttt{Metre})_{ATB}$ quaver probabilities are fairly similar to before; but this time the likelihood of a quaver plummets on the addition of PositionInBar. The reason for this is that quavers occur on some fourth beats in *Innocents*, but not on second beats. These beats can be differentiated by PositionInBar.

8.4.1.2 Prediction of Bass Given Soprano

The best multiple viewpoint system is exactly the same as that used in §8.4.1.1, again using an $\hbar$ of 0. The best performing viewpoint, $(\texttt{PositionInBar} \otimes \texttt{LastInPhrase})_S \otimes (\texttt{Duration} \otimes \texttt{Metre})_B$, has already been discussed; therefore Table 8.11 presents additions and deletions relevant to the creation of another viewpoint which performs well in conjunction with $(\texttt{Duration})_B$. Cross-entropies are shown at each stage.

Evolution of $(\texttt{TactusPositionInBar} \otimes \texttt{FirstInPhrase})_S \otimes (\texttt{DurRatio} \otimes \texttt{LastInPhrase})_B$ We begin with the intra-layer linking of DurRatio and LastInPhrase to form $(\texttt{DurRatio} \otimes \texttt{LastInPhrase})_B$. Similar viewpoint DurRatio ⊗ Phrase has already been examined in relation to melodic modelling in §7.2. Since chords at phrase endings tend to

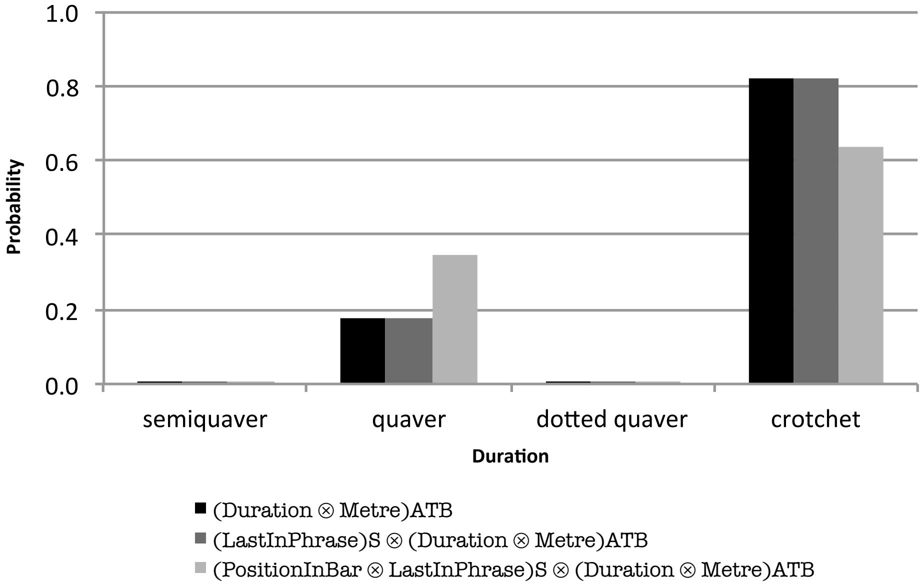


Figure 8.12: Bar chart showing LTM+ prediction probability distributions for viewpoints $(\texttt{Duration} \otimes \texttt{Metre})_{ATB}$, $(\texttt{LastInPhrase})_S \otimes (\texttt{Duration} \otimes \texttt{Metre})_{ATB}$ and $(\texttt{PositionInBar} \otimes \texttt{LastInPhrase})_S \otimes (\texttt{Duration} \otimes \texttt{Metre})_{ATB}$, predicting `Duration` in the alto/tenor/bass given soprano on the second beat of the penultimate bar of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37).

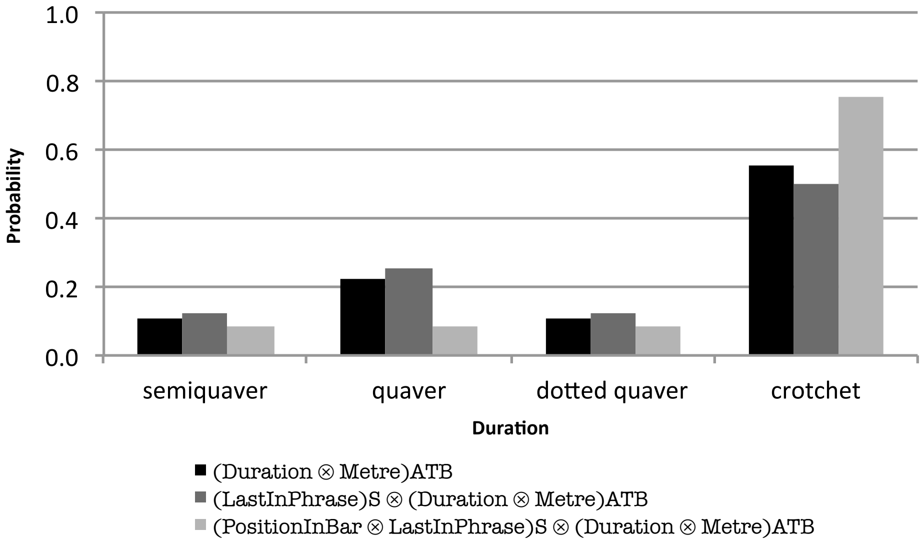


Figure 8.13: Bar chart showing STM prediction probability distributions for viewpoints $(\texttt{Duration} \otimes \texttt{Metre})_{ATB}$, $(\texttt{LastInPhrase})_S \otimes (\texttt{Duration} \otimes \texttt{Metre})_{ATB}$ and $(\texttt{PositionInBar} \otimes \texttt{LastInPhrase})_S \otimes (\texttt{Duration} \otimes \texttt{Metre})_{ATB}$, predicting `Duration` in the alto/tenor/bass given soprano on the second beat of the penultimate bar of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37).

Multiple viewpoint system	x-ent.
$\{(\texttt{Duration})_B\}$	0.99
$+$ $(\texttt{DurRatio} \otimes \texttt{LastInPhrase})_B$	0.87
$+$ $(\texttt{TactusPositionInBar})_S \otimes (\texttt{DurRatio} \otimes \texttt{LastInPhrase})_B$	0.82
$+$ $(\texttt{TactusPositionInBar} \otimes \texttt{FirstInPhrase})_S \otimes (\texttt{DurRatio} \otimes \texttt{LastInPhrase})_B$	0.80
$-$ $(\texttt{DurRatio} \otimes \texttt{LastInPhrase})_B$	0.79

Table 8.11: Cross-entropies (*x-ent.*, bits/prediction) are shown at each point in the evolution of one of the better performing version 3 viewpoints for the prediction of `Duration` in the bass given soprano (corpus 'A'), assuming that no other viewpoints were added or deleted during viewpoint selection.

be long, it follows that `DurRatio` tends to have values greater than 1 at these positions. Next, `TactusPositionInBar` is added to the soprano, giving $(\texttt{TactusPositionInBar})_S \otimes (\texttt{DurRatio} \otimes \texttt{LastInPhrase})_B$. This, for example, modifies the distributions such that there is a reduction in the likelihood of long chords occurring on weak tactus beats. Finally, `FirstInPhrase` is added to the soprano, creating $(\texttt{TactusPositionInBar} \otimes \texttt{FirstInPhrase})_S \otimes (\texttt{DurRatio} \otimes \texttt{LastInPhrase})_B$. `DurRatio` tends to have values less than 1 at first in phrase.

Figure 8.14 shows LTM+ prediction probability distributions for the three viewpoints, predicting `Duration` in the bass given soprano at the first chord of the last bar of hymn tune *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97). $\texttt{DurRatio}\otimes \texttt{LastInPhrase})_B$ predicts a crotchet with very high probability. Since the previous chord is of crotchet length this corresponds with a `DurRatio` value of 1, which is not unreasonable given that this chord is not at the end of a phrase and that much of the corpus is substantially isochronous. The addition of `TactusPositionInBar` results in a transfer of probability mass from crotchet to minim. This is because longer chords are more likely on the first and third beats of the bar. The most noticeable effect of the addition of `FirstInPhrase` is the reduction in probability of a quaver. Knowing that this chord does not begin a phrase, this suggests a tendency for the first chord in a phrase to be shorter than the last of the previous, as expected. Note that this chord is actually a minim; therefore these viewpoints do not predict well in this instance.

STM prediction probability distributions for the same three viewpoints are shown in Figure 8.15. $(\texttt{DurRatio} \otimes \texttt{LastInPhrase})_B$ does not predict a crotchet with such a high probability as before. There is a much bigger difference in the probability of a crotchet on the addition of `TactusPositionInBar`, however. In this case, a crotchet becomes less likely than a quaver (a `DurRatio` value of 0.5 occurs quite frequently on the first beat of the bar in the harmonisation of *Das ist meine Freude*). The picture changes dramatically on the addition of `FirstInPhrase`: a quaver now becomes very unlikely, while there is a surge in the probabilities of a crotchet and minim. This is simply because the aforementioned `DurRatio` value of 0.5 occurs only at the first in phrase.

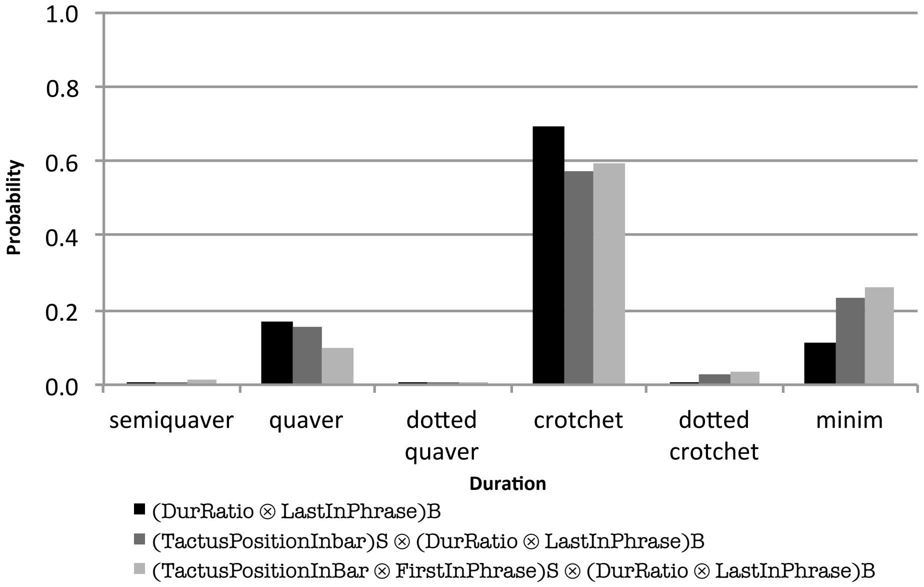


Figure 8.14: Bar chart showing LTM+ prediction probability distributions for viewpoints $(\texttt{DurRatio} \otimes \texttt{LastInPhrase})_B$, $(\texttt{TactusPositionInBar})_S \otimes (\texttt{DurRatio} \otimes \texttt{LastInPhrase})_B$ and $(\texttt{TactusPositionInBar} \otimes \texttt{FirstInPhrase})_S \otimes (\texttt{DurRatio} \otimes \texttt{LastInPhrase})_B$, predicting `Duration` in the bass given soprano at the first chord of the last bar of hymn tune *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97).

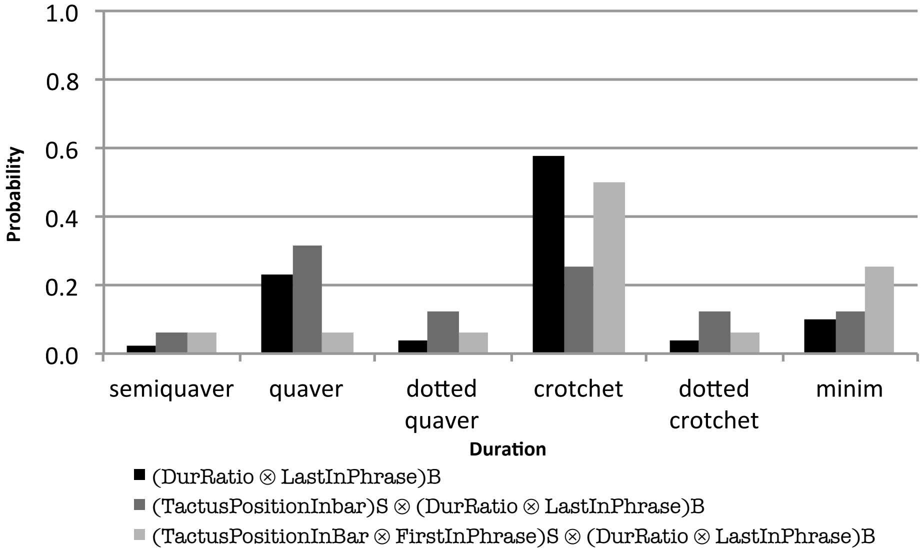


Figure 8.15: Bar chart showing STM prediction probability distributions for viewpoints $(\texttt{DurRatio} \otimes \texttt{LastInPhrase})_B$, $(\texttt{TactusPositionInBar})_S \otimes (\texttt{DurRatio} \otimes \texttt{LastInPhrase})_B$ and $(\texttt{TactusPositionInBar} \otimes \texttt{FirstInPhrase})_S \otimes (\texttt{DurRatio} \otimes \texttt{LastInPhrase})_B$, predicting `Duration` in the bass given soprano at the first chord of the last bar of hymn tune *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97).

Multiple viewpoint system	x-ent.
$\{(\texttt{Duration})_{AT}\}$	0.84
$+\ (\texttt{DurRatio})_{AT} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_{B}$	0.76
$+\ (\texttt{DurRatio} \otimes \texttt{FirstInBar})_{AT} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_{B}$	0.72
$+\ (\texttt{DurRatio} \otimes \texttt{FirstInBar})_{AT} \otimes (\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{B}$	0.67
$+\ (\texttt{Cont})_{S} \otimes (\texttt{DurRatio} \otimes \texttt{FirstInBar})_{AT} \otimes (\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{B}$	0.66
$-\ (\texttt{DurRatio})_{AT} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_{B}$	0.67

Table 8.12: Cross-entropies (*x-ent.*, bits/prediction) are shown at each point in the evolution of one of the better performing version 3 viewpoints for the prediction of `Duration` in the alto/tenor given soprano/bass (corpus 'A'), assuming that no other viewpoints were added or deleted during viewpoint selection.

8.4.1.3 Prediction of Alto/Tenor Given Soprano/Bass

The best multiple viewpoint system again uses an $\hbar$ of 0. The best performing viewpoint is $(\texttt{DurRatio}\otimes\texttt{TactusPositionInBar})_{AT}\otimes(\texttt{Phrase})_{B}$. This is essentially the same as $(\texttt{TactusPositionInBar} \otimes \texttt{FirstInPhrase})_{S} \otimes (\texttt{DurRatio} \otimes \texttt{LastInPhrase})_{B}$ (since `Phrase` is a combination of `FirstInPhrase` and `LastInPhrase`), which was investigated in §8.4.1.2. Similarly, $(\texttt{Duration} \otimes \texttt{PositionInBar})_{AT}$ is equivalent to $(\texttt{Duration} \otimes \texttt{PositionInBar})_{SATB}$, which was analysed in §8.2.1. This being the case, these viewpoints are ignored here, and Table 8.12 instead presents additions and deletions relevant to the creation of another viewpoint which performs well in conjunction with $(\texttt{Duration})_{AT}$. Cross-entropies are shown at each stage. Please note that although the removal of $(\texttt{DurRatio})_{AT} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_{B}$ causes the cross-entropy to rise here, the deletion makes sense in the context of a full viewpoint selection.

Evolution of $(\texttt{Cont})_{S} \otimes (\texttt{DurRatio} \otimes \texttt{FirstInBar})_{AT} \otimes (\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{B}$ A chord at the end of a phrase is quite likely to be long, but `ScaleDegree` in the bass may indicate where this is more or less probable; for example, a tonic in the bass is particularly likely to be long, bearing in mind that pieces generally end in this way. We can see, then, how $(\texttt{DurRatio})_{AT}\otimes(\texttt{ScaleDegree}\ominus\texttt{LastInPhrase})_{B}$ is an improvement on $(\texttt{DurRatio})_{AT}$ for prediction at phrase endings. Since long notes are also generally quite common on the first beat of the bar, the addition of `FirstInBar` further enhances performance. The fact that `DurRatio` deals only in relative note lengths can be a drawback; for example, if the preceding chord is of unusually short duration, the length being predicted at the end of a phrase would be a large multiple of this, resulting in a low prediction probability. The introduction of a known value of `Duration` in the bass overcomes this problem by effectively making use of 1^{st}-order `Duration` statistics, which are finer-grained than 0^{th}-order `DurRatio` statistics. A `Cont` value of T in the soprano at the last chord of a phrase indicates that a shorter than usual duration is more likely; see the conclusion of the third phrase of *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36). Although `FirstInPiece` is intra-layer linked with `Cont` later in the viewpoint selection process, this does not really create a new viewpoint since the first chord of a piece can never coincide with the last chord of a phrase. Two effectively identical viewpoints have found their way into the multiple viewpoint system, very probably because the bias used during viewpoint selection is not optimal.

Figure 8.16 shows LTM+ prediction probability distributions for viewpoints $(\texttt{DurRatio})_{AT} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_{B}$, $(\texttt{DurRatio} \otimes \texttt{FirstInBar})_{AT} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_{B}$ and $(\texttt{DurRatio} \otimes \texttt{FirstInBar})_{AT} \otimes (\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{B}$, predicting `Duration` in the alto/tenor given soprano/bass in the final chord of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36). Here, we investigate in detail what happens when the preceding chord is very short (in this case a quaver). In the $(\texttt{DurRatio})_{AT} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_{B}$ distribution, a crotchet has by far the highest probability. This seems reasonable, considering that it is quite common in the corpus for phrases to end with a `DurRatio` value of 2 (*e.g.*, a minim followed by a semibreve in a piece with a $\frac{4}{2}$ time signature). A `DurRatio` value of 1 often occurs at phrase endings in pieces starting with an anacrusis on the last beat of the bar, hence the relatively high probability of a quaver. A minim, at four times the length of the preceding note, is deemed to be less likely.

A chord at the beginning of a bar is quite likely to be long (more so than anywhere else, on average); since prediction is on the third beat, it therefore comes as no surprise that there is a transfer of probability mass from crotchet and dotted crotchet to quaver (equivalent to a `DurRatio` value of 1) on the addition of `FirstInBar`. On the other hand, it does seem rather surprising that the likelihood of a minim is very slightly higher. The distribution is completely transformed by the introduction of `Duration` in the bass (where a minim has already been predicted), resulting in an extremely high probability for a minim, as expected. In fact, it is obvious from the chart that no other predictions in the distribution had been seen in the corpus. Finally, the addition of `Cont` makes absolutely no difference in this instance, and so the $(\texttt{Cont})_{S} \otimes (\texttt{DurRatio} \otimes \texttt{FirstInBar})_{AT} \otimes (\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{B}$ distribution is not shown.

The STM+ prediction probability distributions for these viewpoints are easy to describe, and so a bar chart is not presented. $(\texttt{DurRatio})_{AT} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_{B}$ predicts a quaver with a probability of 0.5, and the remaining probability mass is shared equally between the other predictions. This is due to the fact that E♭3 occurs only once before in the bass at the end of a phrase, in conjunction with a `DurRatio` value of 1. Since all phrases end on the third beat of the bar, even if there were multiple occurrences of E♭3 there would be no possibility of `FirstInBar` discriminating between them; therefore the distribution is unchanged on the addition of `FirstInBar`. The distribution is radically altered by the introduction of `Duration` in the bass, but in a different way from the LTM+: none of the predictions have been seen before, resulting in a completely uniform distribution. Finally, since the addition of `Cont` made

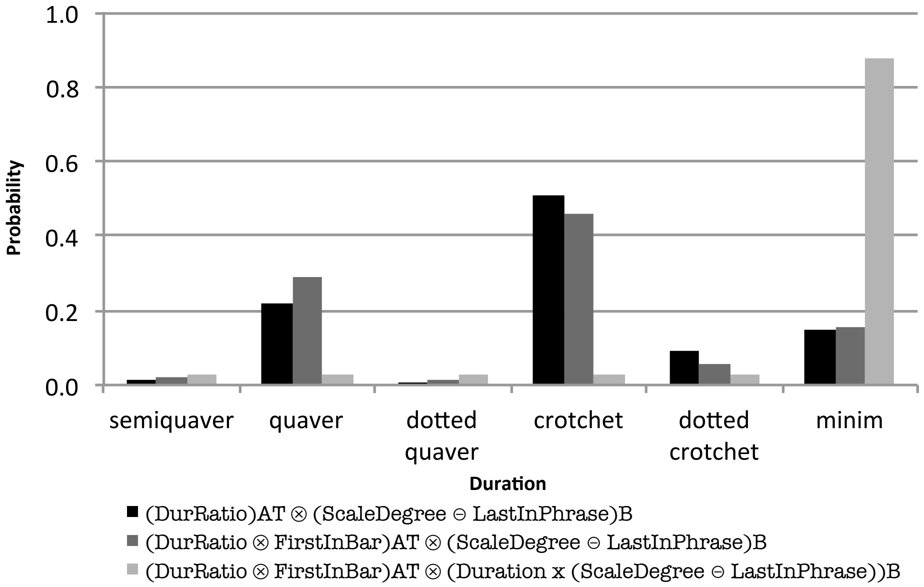


Figure 8.16: Bar chart showing LTM+ prediction probability distributions for viewpoints $(\texttt{DurRatio})_{AT} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_B$, $(\texttt{DurRatio} \otimes \texttt{FirstInBar})_{AT} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_B$ and $(\texttt{DurRatio} \otimes \texttt{FirstInBar})_{AT} \otimes (\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_B$, predicting `Duration` in the alto/tenor given soprano/bass in the final chord of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36).

no difference to the LTM+ distribution, it is guaranteed to make no difference to the STM one. $(\texttt{Cont})_S \otimes (\texttt{DurRatio} \otimes \texttt{FirstInBar})_{AT} \otimes (\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_B$ is clearly not a good STM predictor in this case, at least.

8.4.2 Prediction of `Cont` Alone

8.4.2.1 Prediction of Alto/Tenor/Bass Given Soprano

The best multiple viewpoint system uses an $\hbar$ of 2. Table 8.13 presents additions and deletions relevant to the creation of the viewpoint which performs best in conjunction with $(\texttt{Cont})_{ATB}$. Cross-entropies are shown at each stage.

Evolution of $(\texttt{Cont} \otimes \texttt{Interval})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$ We begin with the linking of `TactusPositionInBar` with `Cont` in the alto, tenor and bass. Since notes tied across bar lines are almost unheard of in this corpus, an obvious regularity picked up by this viewpoint is that notes on the first beat of the bar have an extremely high likelihood of having a `Cont` value of F. Similarly, a `Cont` value of T is much more likely to occur at a non-tactus position than on a tactus beat. With the addition of `Cont` in the soprano, it is possible to further refine the prediction probabilities by matching further known `Cont` values in the context. Finally, linking `Interval` with `Cont` in the soprano is useful in distinguishing notes that are likely to be unessential from those likely to form part of a new chord; for example, when the soprano pitch moves by leap

Multiple viewpoint system	x-ent.
$\{(\texttt{Cont})_{ATB}\}$	1.18
$+\ (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$	0.97
$-\ (\texttt{Cont})_{ATB}$	0.81
$+\ (\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$	0.77
$-\ (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$	0.76
$+\ (\texttt{Cont} \otimes \texttt{Interval})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$	0.69

Table 8.13: Cross-entropies (*x-ent.*, bits/prediction) are shown at each point in the evolution of the best performing version 3 viewpoint for the prediction of Cont in the alto/tenor/bass given soprano (corpus 'A'), assuming that no other viewpoints were added or deleted during viewpoint selection.

to a non-tactus position, a complete change of chord becomes much more likely.

Figure 8.17 shows LTM+ prediction probability distributions for viewpoints $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$, $(\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$ and $(\texttt{Cont} \otimes \texttt{Interval})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$, predicting Cont in the alto, tenor and bass given soprano in the penultimate chord (after expansion) of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33). The soprano note is known to have a Cont value of T, which constrains all of the prediction probability distributions. The $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$ distribution indicates that, in these circumstances, it is generally more likely for a single note to be resounded (almost certainly with a pitch change) than for more than one note to change. A change in the alto note is most probable, followed by bass and then tenor. A change in tenor and bass together is about as likely as the latter case. Extra-tactus movement in the alto occurs in the two-chord N-gram context, and one could reasonably expect a continuation of this movement. The addition of Cont to the soprano has the effect of transferring probability mass from tenor/bass movement to bass alone, making this as likely as alto movement. The reason for this is not clear, but it reinforces the dominance of movement in a single part. Finally, the addition of Interval to the soprano effectively evens out the probabilities for movement in a single part, with movement in the tenor being most likely by a tiny margin. In fact, an appoggiatura is resolved in the alto in the hymnal harmonisation. The STM distributions (not shown) are completely different, inasmuch as $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$ assigns tenor movement the highest probability, and the other viewpoints deem bass movement to be most likely; otherwise, the distributions are fairly uniform (not surprising, as the data is sparse).

8.4.2.2 Prediction of Bass Given Soprano

The best multiple viewpoint system uses an $\hbar$ of 1. Table 8.14 presents additions and deletions relevant to the creation of the viewpoint which performs best in conjunction with $(\texttt{Cont})_B$. Cross-entropies are shown at each stage.

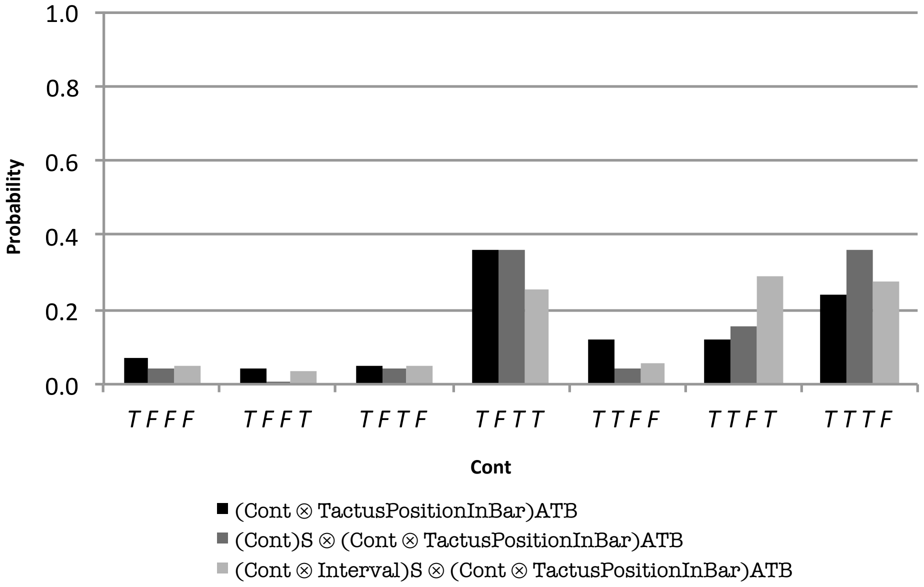


Figure 8.17: Bar chart showing LTM+ prediction probability distributions for viewpoints $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$, $(\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$ and $(\texttt{Cont} \otimes \texttt{Interval})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$, predicting Cont in the alto/tenor/bass given soprano in the penultimate chord (after expansion) of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33).

Multiple viewpoint system	x-ent.
$\{(\texttt{Cont})_B\}$	0.56
$+\ (\texttt{Cont} \otimes \texttt{Metre})_B$	0.42
$-\ (\texttt{Cont})_B$	0.31
$+\ (\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{Metre})_B$	0.30
$+\ (\texttt{DurRatio} \otimes \texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{Metre})_B$	0.28
$-\ (\texttt{Cont} \otimes \texttt{Metre})_B$	0.27

Table 8.14: Cross-entropies (*x-ent.*, bits/prediction) are shown at each point in the evolution of the best performing version 3 viewpoint for the prediction of Cont in the bass given soprano (corpus ‘A’), assuming that no other viewpoints were added or deleted during viewpoint selection.

Evolution of $(\texttt{DurRatio} \otimes \texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{Metre})_B$ The first stage of the evolution of this viewpoint is the linking of `Cont` and `Metre` in the bass. Since the latter viewpoint assumes that certain positions in the bar (in terms of the tactus) are equivalent, it has access to data which is less sparse than is available to `TactusPositionInBar`. Particularly relevant to the prediction of `Cont` is the fact that all non-tactus positions are regarded as equivalent, thereby increasing the amount of data which can be used to predict at positions where there are likely to be unessential notes (or their resolutions). The addition of `Cont` to the soprano is likely to have much the same effect as in $(\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{ATB}$ (see §8.4.2.1 above). Finally, the addition of `DurRatio` to the soprano makes the position of non-tactus chords more definite, as well as effectively extending the range of this very short ($\hbar = 1$) context.

Figure 8.18 shows LTM+ and STM prediction probability distributions for viewpoints $(\texttt{Cont} \otimes \texttt{Metre})_B$, $(\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{Metre})_B$ and $(\texttt{DurRatio} \otimes \texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{Metre})_B$, predicting `Cont` in the bass given soprano in the third chord (after expansion) of the tenth bar of hymn tune *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97). Continuation of the bass note is deemed more likely according to $(\texttt{Cont} \otimes \texttt{Metre})_B$ (in other words, movement is more likely in the inner parts). Since movement could occur in any of the lower parts here, the likelihoods make sense by virtue of there being two inner parts to one bass part. The LTM+ and STM probabilities are remarkably similar. The addition of `Cont` and then `DurRatio` to the soprano of this viewpoint makes little difference to the LTM+, there being a slight shift in probability mass towards movement in the bass. Their addition in the STM makes a huge difference in this case, however. $(\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{Metre})_B$ sees movement in the bass to be as likely as movement in either or both of the inner parts, while $(\texttt{DurRatio} \otimes \texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{Metre})_B$ deems movement in the bass to be extremely likely. Such large shifts in the probability mass are not unreasonable bearing in mind the sparsity of the data. In fact, the hymnal harmonisation shows a continuation in the bass note at this point, as shown to be more likely by the LTM+.

8.4.2.3 Prediction of Alto/Tenor Given Soprano/Bass

The best multiple viewpoint system uses an $\hbar$ of 2. Table 8.15 presents additions and deletions relevant to the creation of the viewpoint which performs best in conjunction with $(\texttt{Cont})_{AT}$. Cross-entropies are shown at each stage. Please note that although the removal of $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Cont})_B$ causes the cross-entropy to rise here, the deletion makes sense in the context of a full viewpoint selection.

Evolution of $(\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Cont})_B$ We begin with the linking of `TactusPositionInBar` with `Cont` in the alto and tenor, which is similar to the way in which evolution began in §8.4.2.1. By adding `Cont` to the bass and then to the soprano, the prediction probabilities are refined by matching further known `Cont`

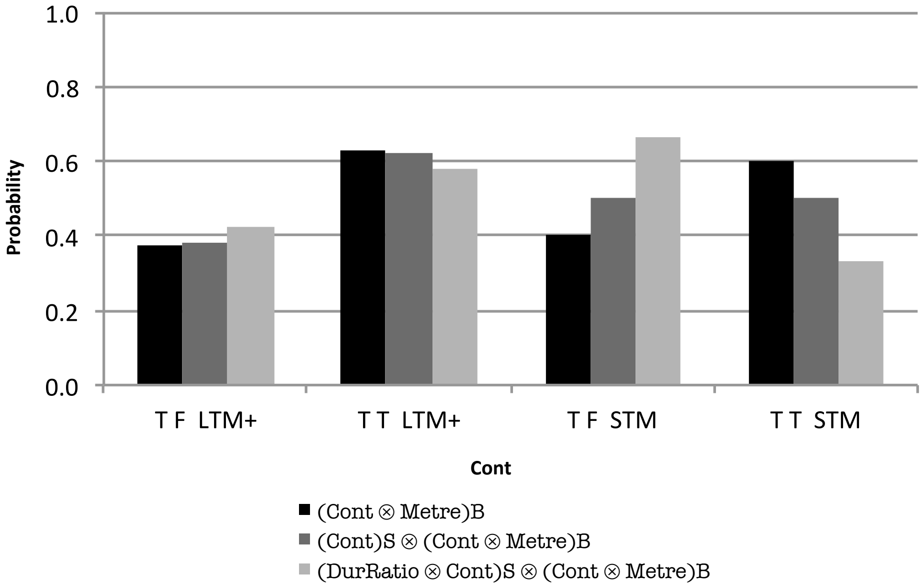


Figure 8.18: Bar chart showing LTM+ and STM prediction probability distributions for viewpoints $(\texttt{Cont} \otimes \texttt{Metre})_B$, $(\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{Metre})_B$ and $(\texttt{DurRatio} \otimes \texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{Metre})_B$, predicting Cont in the bass given soprano in the third chord (after expansion) of the tenth bar of hymn tune *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97).

Multiple viewpoint system	x-ent.
$\{(\texttt{Cont})_{AT}\}$	0.89
$+\ (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT}$	0.70
$-\ (\texttt{Cont})_B$	0.50
$+\ (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Cont})_B$	0.49
$-\ (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT}$	0.48
$+\ (\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Cont})_B$	0.46
$-\ (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Cont})_B$	0.47

Table 8.15: Cross-entropies (*x-ent.*, bits/prediction) are shown at each point in the evolution of the best performing version 3 viewpoint for the prediction of Cont in the alto/tenor given soprano/bass (corpus 'A'), assuming that no other viewpoints were added or deleted during viewpoint selection.

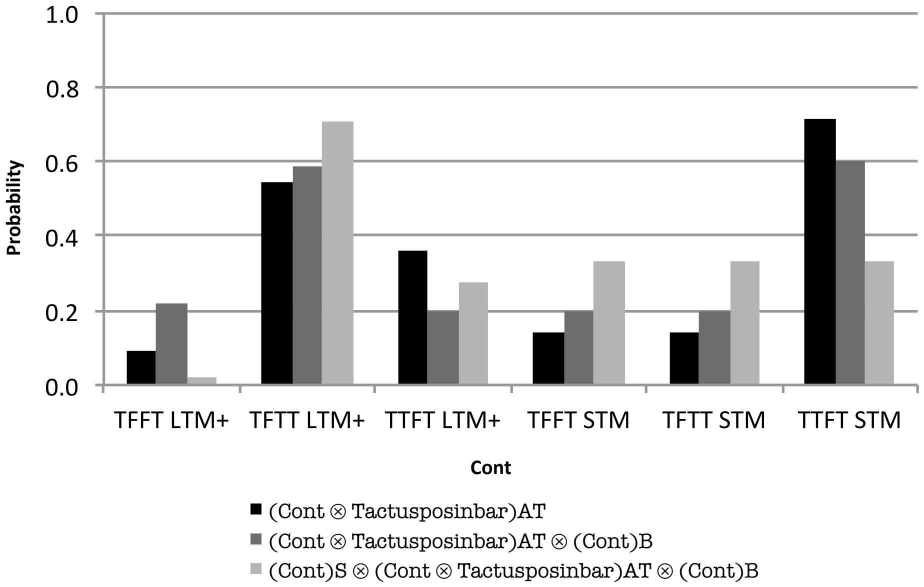


Figure 8.19: Bar chart showing LTM+ and STM prediction probability distributions for viewpoints $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT}$, $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Cont})_B$ and $(\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Cont})_B$, predicting `Cont` in the alto/tenor given soprano/bass in the penultimate chord (after expansion) of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33).

values in the context.

Figure 8.19 shows LTM+ and STM prediction probability distributions for viewpoints $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT}$, $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Cont})_B$ and $(\texttt{Cont})_S \otimes (\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT} \otimes (\texttt{Cont})_B$, predicting `Cont` in the alto/tenor given soprano/bass. For direct comparison with similar viewpoints in §8.4.2.1, prediction takes place in the penultimate chord (after expansion) of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33). The soprano and bass notes are known to have a `Cont` value of T, which constrains all of the prediction probability distributions. The LTM+ $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{AT}$ distribution has a similar shape to the relevant part of the ATB prediction distribution, but with generally higher probabilities due to the smaller domain. Whereas this distribution shows a change in alto note to be most likely, matching our expectation of a continuation of extra-tactus movement in the alto, it is a change in tenor note which is deemed most probable by the STM distribution. The addition of `Cont` to the bass has the effect in the LTM+ of transferring a good deal of probability mass from tenor movement to alto/tenor movement, and making alto movement slightly more likely. Similarly, in the STM, there is transfer of probability mass away from tenor movement. Finally, the addition of `Cont` to the soprano causes the STM distribution to become completely uniform in this case. In the LTM+, however, movement in the alto (as occurs in the hymnal harmonisation) becomes even more probable.

Multiple viewpoint system	x-ent.
$\{(\texttt{Pitch})_{ATB}\}$	4.96
$+$ $(\texttt{ScaleDegree})_{SATB}$	4.05
$-$ $(\texttt{Pitch})_{ATB}$	4.21
$+$ $(\texttt{ScaleDegree})_{S} \otimes (\texttt{Cont} \otimes \texttt{ScaleDegree})_{ATB}$	4.11
$+$ $(\texttt{ScaleDegree} \otimes \texttt{Phrase})_{S} \otimes (\texttt{Cont} \otimes \texttt{ScaleDegree})_{ATB}$	4.04
$-$ $(\texttt{ScaleDegree})_{SATB}$	4.09
$-$ $(\texttt{ScaleDegree})_{S} \otimes (\texttt{Cont} \otimes \texttt{ScaleDegree})_{ATB}$	4.22

Table 8.16: Cross-entropies (*x-ent.*, bits/prediction) are shown at each point in the evolution of the best performing version 3 viewpoint for the prediction of Pitch in the alto/tenor/bass given soprano (corpus 'A'), assuming that no other viewpoints were added or deleted during viewpoint selection.

8.4.3 Prediction of Pitch Alone

8.4.3.1 Prediction of Alto/Tenor/Bass Given Soprano

The best multiple viewpoint system uses an $\hbar$ of 3. Table 8.16 presents additions and deletions relevant to the creation of the viewpoint which performs best in conjunction with $(\texttt{Pitch})_{ATB}$. Cross-entropies are shown at each stage. Please note that although the removal of viewpoints causes the cross-entropy to rise here, their deletion makes sense in the context of a full viewpoint selection.

Evolution of $(\texttt{ScaleDegree} \otimes \texttt{Phrase})_{S} \otimes (\texttt{Cont} \otimes \texttt{ScaleDegree})_{ATB}$ The evolution begins with $(\texttt{ScaleDegree})_{SATB}$. ScaleDegree is known to be a good primitive viewpoint, appearing often (linked with other primitive viewpoints) in many multiple viewpoint systems for the prediction of both melody and harmony (see Tables 6.10 and 6.11). Since Cont is predicted before Pitch, linking Cont with ScaleDegree in the alto, tenor and bass means that known Cont values in the prediction (as well as in the context) can greatly assist with the prediction of Pitch. Finally, the addition of Phrase to the soprano means that the viewpoint is able to exploit harmonic regularities associated with phrase boundaries. An analysis of a sample of corpus 'A' revealed that by far the most common chord at the beginning of a phrase is the tonic in root position, while the dominant and submediant are fairly common. An examination of harmonic regularities at phrase endings is presented in the discussion of viewpoint $(\texttt{ScaleDegree} \otimes \texttt{LastInPhrase})_{SATB}$ in §8.2.3.

Figure 8.20 shows LTM+ and STM prediction probability distributions for viewpoints $(\texttt{ScaleDegree})_{SATB}$, $(\texttt{ScaleDegree})_{S} \otimes (\texttt{Cont} \otimes \texttt{ScaleDegree})_{ATB}$ and $(\texttt{ScaleDegree} \otimes \texttt{Phrase})_{S} \otimes (\texttt{Cont} \otimes \texttt{ScaleDegree})_{ATB}$, predicting Pitch in the alto, tenor and bass given soprano in the last chord (after expansion) of the third phrase of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36). Note that the domain is constrained by the already predicted Cont values such that effectively Pitch is only

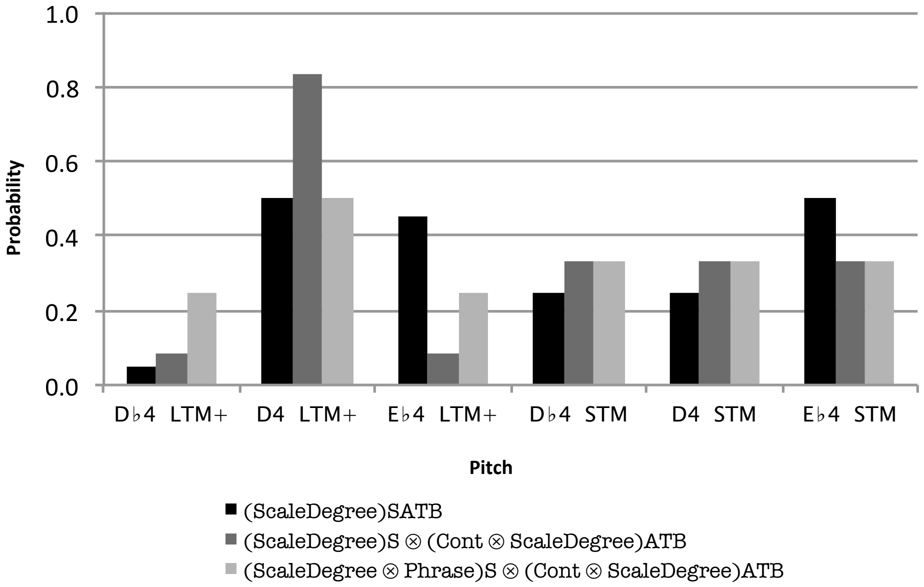


Figure 8.20: Bar chart showing LTM+ and STM prediction probability distributions for viewpoints $(\texttt{ScaleDegree})_{SATB}$, $(\texttt{ScaleDegree})_S \otimes (\texttt{Cont} \otimes \texttt{ScaleDegree})_{ATB}$ and $(\texttt{ScaleDegree} \otimes \texttt{Phrase})_S \otimes (\texttt{Cont} \otimes \texttt{ScaleDegree})_{ATB}$, predicting `Pitch` in the alto/tenor/bass (effectively only alto) given soprano in the last chord (after expansion) of the third phrase of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36).

being predicted in the alto. As expected, $(\texttt{ScaleDegree})_{SATB}$ in the LTM+ predicts notes D4 and E♭4 with high probability as they belong to the key of the piece, E♭ major. The STM has only previously encountered a tonic note in the alto in conjunction with supertonic in the soprano and dominant in the tenor and bass (immediately prior to the prediction position), and so E♭4 is deemed most likely in this case. The addition of `Cont` in the alto, tenor and bass changes things dramatically in the LTM+: now D4 is by far the most likely prediction, which resolves an appoggiatura. With soprano, tenor and bass notes continuing to sound, a repetition of the alto E♭4 makes no musical sense, and such a thing is highly unlikely to have been seen in the corpus. At this stage, the STM reverts to a uniform distribution. Finally, the addition of `Phrase` to the soprano substantially reduces the LTM+ probability of a D4 (as occurs in the hymnal harmonisation), although it is still much more probable than D♭4 or E♭4. Since prediction here is at the end of a phrase, the data is much more sparse: in fact, it is clear from the distribution that the configuration involving a leading note in the alto occurs only once in the corpus.

8.4.3.2 Prediction of Bass Given Soprano

The best multiple viewpoint system again uses an $\hbar$ of 3. Table 8.17 presents additions and deletions relevant to the creation of the viewpoint which performs best in conjunction with $(\texttt{Pitch})_B$. Cross-entropies are shown at each stage.

Multiple viewpoint system	x-ent.
$\{(\texttt{Pitch})_B\}$	2.92
$+$ $(\texttt{Phrase})_S \otimes (\texttt{Interval})_B$	2.74
$+$ $(\texttt{Phrase})_S \otimes (\texttt{Interval} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_B$	2.49
$+$ $((\texttt{ScaleDegree} \ominus \texttt{Tactus}) \otimes \texttt{Phrase})_S \otimes (\texttt{Interval} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_B$	2.18
$-$ $(\texttt{Phrase})_S \otimes (\texttt{Interval})_B$	2.15

Table 8.17: Cross-entropies (*x-ent.*, bits/prediction) are shown at each point in the evolution of the best performing version 3 viewpoint for the prediction of Pitch in the bass given soprano (corpus 'A'), assuming that no other viewpoints were added or deleted during viewpoint selection.

Evolution of $((\texttt{ScaleDegree} \ominus \texttt{Tactus}) \otimes \texttt{Phrase})_S \otimes (\texttt{Interval} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_B$ We begin with the linking of Phrase in the soprano with Interval in the bass. Since perfect cadences are extremely common in the corpus, it is reasonable to expect that a fall of a perfect fifth and a rise of a perfect fourth will have high probabilities at the end of a phrase. There is much more uniformity in the occurrence of intervals between phrases, although there is a tendency for more rising octaves to appear at this point than is usual. This happens when a phrase ends on a very low bass note, as a better alternative to repetition. The addition of $\texttt{ScaleDegree} \ominus \texttt{Tactus}$ to the bass has the effect of taking ScaleDegree into account at metrically important positions. This provides the bonus of increasing the range of the context when extra-tactus movement occurs. A major drawback to viewpoint Interval is that it is blind to key; but ScaleDegree effectively learns from the corpus which notes belong to the key of a piece, thereby improving on Interval alone. Another way of looking at this is that Interval provides an octave distinguishing capability to ScaleDegree. Finally, the addition of $\texttt{ScaleDegree} \ominus \texttt{Tactus}$ to the soprano gives a better idea of the unfolding chord progression in the context than is possible by looking only at the bass.

Figure 8.21 shows LTM+ probability distributions for $(\texttt{Phrase})_S \otimes (\texttt{Interval})_B$, $(\texttt{Phrase})_S \otimes (\texttt{Interval} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_B$ and $((\texttt{ScaleDegree} \ominus \texttt{Tactus}) \otimes \texttt{Phrase})_S \otimes (\texttt{Interval} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_B$, predicting Pitch in the bass given soprano in the last chord of the first phrase of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37). Predictions with all three probabilities less than 0.014 are not shown. $(\texttt{Phrase})_S \otimes (\texttt{Interval})_B$ predicts C3 and C4 with high probabilities, as anticipated, since the previous note is G3 and perfect cadences are very common. The fact is, however, that a C major chord (or any other chord with C in the bass) at this point makes no musical sense, bearing in mind that the piece is in the key of D major. It takes the addition of $\texttt{ScaleDegree} \ominus \texttt{Tactus}$ to the bass to shift the viewpoint from this erroneous position; we now see that C3 and C4 have an almost zero probability. What we have instead are high likelihoods for the tonic and mediant of D major (albeit that a first inversion chord would be unusual at this point), a chord which completes a

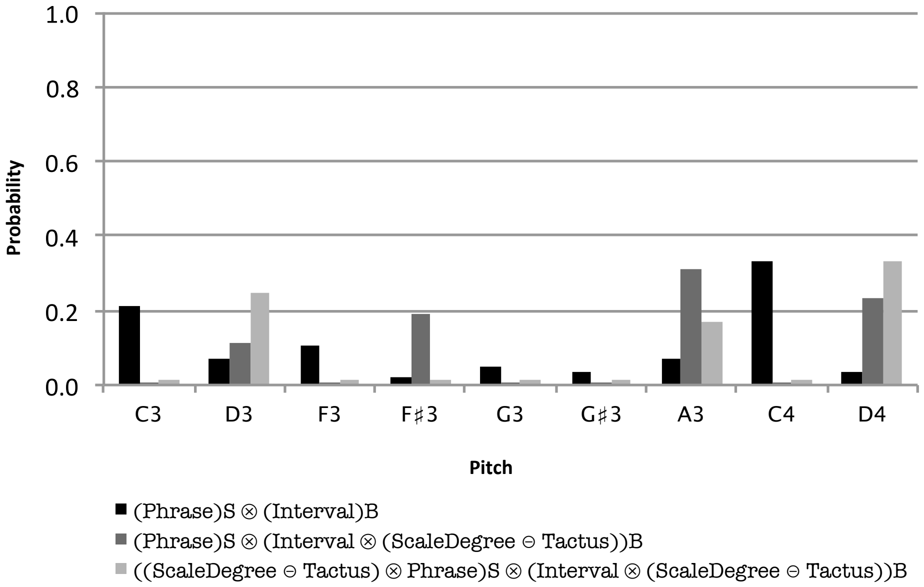


Figure 8.21: Bar chart showing LTM+ prediction probability distributions for viewpoints $(\texttt{Phrase})_S \otimes (\texttt{Interval})_B$, $(\texttt{Phrase})_S \otimes (\texttt{Interval} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_B$ and $((\texttt{ScaleDegree} \ominus \texttt{Tactus}) \otimes \texttt{Phrase})_S \otimes (\texttt{Interval} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_B$, predicting `Pitch` in the bass given soprano in the last chord of the first phrase of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37). Predictions with all three probabilities less than 0.014 are not shown.

perfectly satisfactory plagal cadence. A3 has the highest probability of all, and is also a constituent of a D major chord; but since a second inversion chord would never occur at the very end of a phrase, it almost certainly represents the root of the dominant chord, which completes an imperfect cadence. Finally, the addition of $\texttt{ScaleDegree}\ominus\texttt{Tactus}$ to the soprano provides information which is able to further refine the distribution. D3 (the continuation in the hymnal) and D4 now become highly likely, while the probability of a first inversion chord (represented by F♯3) drops to almost zero. There is also a sharp fall in the likelihood of an A3 (imperfect cadence). The STM distributions are all uniform, and so are of no interest here, due to the fact that no statistics have yet been gathered at phrase endings (we are predicting at the first phrase ending of the piece).

8.4.3.3 Prediction of Alto/Tenor Given Soprano/Bass

The best multiple viewpoint system here uses an $\hbar$ of 1. Table 8.18 presents additions and deletions relevant to the creation of a viewpoint which performs well in conjunction with $(\texttt{Pitch})_{AT}$ (note that $(\texttt{ScaleDegree})_S \otimes (\texttt{Cont} \otimes \texttt{ScaleDegree})_{AT} \otimes (\texttt{ScaleDegree})_B$ performs best, but similar viewpoint $(\texttt{ScaleDegree})_S \otimes (\texttt{Cont} \otimes \texttt{ScaleDegree})_{ATB}$ has already been discussed in §8.4.3.1). Cross-entropies are shown at each stage. Please note that although the removal of $(\texttt{Pitch})_{AT}$ causes the cross-entropy to rise here, its deletion makes sense in the context of a full viewpoint selection.

Multiple viewpoint system	x-ent.
$\{(\texttt{Pitch})_{AT}\}$	2.62
$+$ $(\texttt{ScaleDegree})_{ATB}$	2.09
$-$ $(\texttt{Pitch})_{AT}$	2.19
$+$ $(\texttt{Interval} \otimes \texttt{ScaleDegree})_{AT} \otimes (\texttt{ScaleDegree})_{B}$	2.03

Table 8.18: Cross-entropies (*x-ent.*, bits/prediction) are shown at each point in the evolution of one of the better performing version 3 viewpoint for the prediction of Pitch in the alto/tenor given soprano/bass (corpus 'A'), assuming that no other viewpoints were added or deleted during viewpoint selection.

Evolution of $(\texttt{Interval} \otimes \texttt{ScaleDegree})_{AT} \otimes (\texttt{ScaleDegree})_{B}$ Let us start this discussion with $(\texttt{Pitch})_{AT}$ in spite of its non-appearance in the evolving viewpoint. The fact that this viewpoint is blind to key means that progressions which are inappropriate to the key of the piece can be predicted with high probability, and *vice versa.* $(\texttt{ScaleDegree})_{ATB}$ rectifies this major flaw by relating pitches to the tonic. The addition of a bass note to the context further improves performance; the manner in which the bass line moves, very often utilising the root or third of a chord, is more indicative of a progression than that of the soprano. ScaleDegree still has room for improvement, however, in that it is unable to distinguish between octaves. The addition of Interval to the alto and tenor is a great help in that regard, as we now know whether notes rise or fall. Notice that Interval is not added to the bass; it would appear that a better knowledge of the bass line contour is of little importance when predicting the alto and tenor.

Figure 8.22 shows LTM+ probability distributions for viewpoints $(\texttt{Pitch})_{AT}$, $(\texttt{ScaleDegree})_{ATB}$ and $(\texttt{Interval} \otimes \texttt{ScaleDegree})_{AT} \otimes (\texttt{ScaleDegree})_{B}$, predicting Pitch in the alto/tenor given soprano/bass on the third beat of the penultimate bar of hymn tune *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97). Predictions with all three probabilities less than 0.043 are not shown. Three predictions are particularly likely according to $(\texttt{Pitch})_{AT}$. One of them, D4/G3, completes a second inversion chord, which is too early to form part of a cadence. It is also on a strong beat, which makes it inappropriate as a passing $^6_4$ chord. There is, of course, no possible way for Pitch to gauge metrical strength or proximity to a phrase ending. Both F4/B♭3 and F4/D4 complete a IVb chord, and are acceptable (although doubling of the major third, as occurs with F4/D4, is often avoided). F4/B♭3 is the continuation found in the hymnal. It is interesting to note that F4/F4 is given an almost zero probability, which resonates with a well-known rule of harmony: the alto/tenor combination in the previous chord was C4/C4, and parallel unisons are avoided in harmonic progressions (like parallel octaves). Using $(\texttt{ScaleDegree})_{ATB}$ instead of $(\texttt{Pitch})_{AT}$ makes an enormous difference to the D4/G3 (second inversion) prediction, which now has a likelihood close to zero despite ScaleDegree being no more able to recognise metrical or sequential position than Pitch. It is clear that this progression, as described by ScaleDegree, has rarely occurred. As a

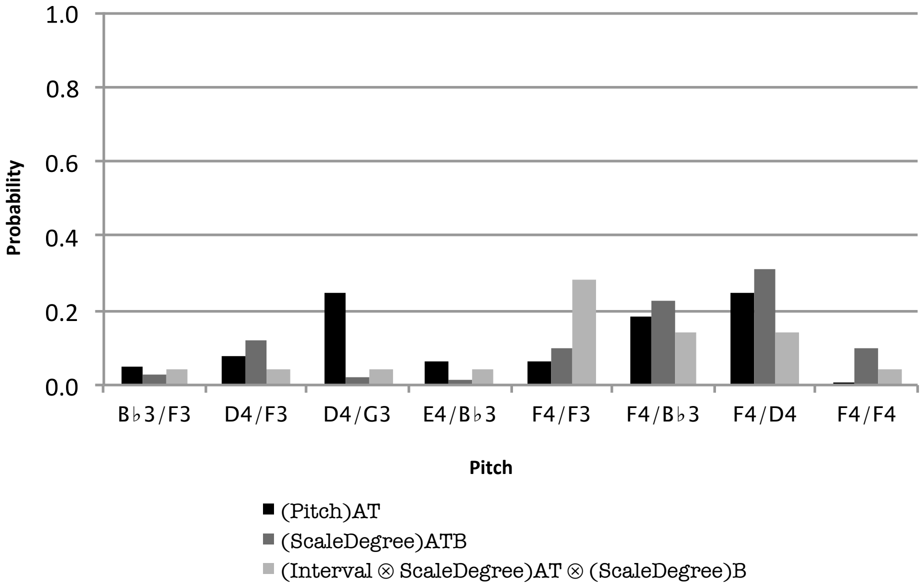


Figure 8.22: Bar chart showing LTM+ prediction probability distributions for viewpoints $(\texttt{Pitch})_{AT}$, $(\texttt{ScaleDegree})_{ATB}$ and $(\texttt{Interval} \otimes \texttt{ScaleDegree})_{AT} \otimes (\texttt{ScaleDegree})_{B}$, predicting Pitch in the alto/tenor given soprano/bass on the third beat of the penultimate bar of hymn tune *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97). Predictions with all three probabilities less than 0.043 are not shown.

consequence, most of the IVb chord combinations have increased probabilities, including F4/F4. This rule-breaking progression is given credence because of the inability of ScaleDegree to distinguish between octaves. Finally, the addition of Interval to the alto and tenor results in F4/F3 becoming most likely, while F4/D4 suffers a particularly large fall in probability. The latter can be considered a less good solution since the tenor D4 overlaps with the preceding alto C4. In the STM (not shown) the only significant likelihood with respect to $(\texttt{Pitch})_{AT}$ and $(\texttt{ScaleDegree})_{ATB}$ is that of D4/G3, which appears earlier in the piece as part of a Ic–V–I cadence. The distribution becomes completely uniform on the addition of Interval.

8.5 Summary and Conclusion

In this chapter we examined version 1, 2 and 3 multiple viewpoint systems and speculated on why certain viewpoints had been selected from a music theoretic point of view. Musical regularities and other factors relevant to viewpoints performing well are summarised below for a selection of the viewpoints analysed.

Let us begin by reviewing version 1 viewpoints, where we find that metrical importance also correlates with Cont (in addition to Duration and Pitch). Metrical structure, sometimes inferred from the corpus, is key to the success of $(\texttt{Duration} \otimes \texttt{PositionInBar})_{SATB}$, $(\texttt{Cont} \otimes \texttt{Metre})_{SATB}$ and $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{SATB}$. If pairs of chords in different parts of the corpus have the same intervals in corresponding parts, there is a fair chance that the pairs of chords are the same relative to the tonic. $(\texttt{Cont} \otimes \texttt{Interval})_{SATB}$ can therefore usefully gather statistics on `Cont` for the purpose of its prediction, following which `Pitch` can also be predicted. There is room for improvement in this regard, since `Interval` is blind to key. This deficiency is rectified by viewpoint $(\texttt{Cont} \otimes \texttt{ScaleDegree})_{SATB}$, which in addition is sure to find pairs of chords that are the same except for octave. Finally, $(\texttt{ScaleDegree} \otimes \texttt{LastInPhrase})_{SATB}$ is preferred here to $\texttt{ScaleDegree} \otimes \texttt{Phrase}$, which performs well with respect to melody. This suggests that it may be easier to determine harmonic regularities at phrase endings (*i.e.*, at cadences).

We now move on to version 2 viewpoints. All phrases begin with chords in which all notes are newly sounded; therefore a `Cont` prediction of $\langle F,\ F\rangle$ from $(\texttt{Cont} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInPhrase}))_{SB}$ is overwhelmingly likely. The threading is more important than the viewpoint threaded. Other viewpoints having a predictive edge at phrase and bar levels include $(\texttt{DurRatio} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{SB}$, $(\texttt{DurRatio} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInBar}))_{SATB}$ and $(\texttt{DurRatio} \otimes (\texttt{Interval} \ominus \texttt{First-InBar}))_{SATB}$. Meanwhile, $(\texttt{Cont} \otimes \texttt{PositionInBar})_{SATB}$ is able to infer metrical structure and its correlation with `Cont`. Viewpoint $(\texttt{Interval} \otimes \texttt{InScale})_{SB}$ is able to model how the soprano and bass lines move in relation to each other; its use of `InScale` means that the STM is able to play a large part in the prediction process. $(\texttt{Interval} \otimes (\texttt{ScaleDegree} \ominus \texttt{Tactus}))_{SB}$ and $(\texttt{ScaleDegree} \ominus \texttt{Tactus})_{SATB}$ are similar in nature (the latter predicting the inner parts).

Finally, we consider version 3 viewpoints with a strong prediction performance. At one end of the scale, there are viewpoints which are composed completely of pairwise links (intra- and inter-layer, able to predict at least one attribute) already seen amongst the better performing version 0, 1 and 2 viewpoints examined in Chapters 7 and 8, such as $(\texttt{PositionInBar} \otimes \texttt{LastInPhrase})_{S} \otimes (\texttt{Duration} \otimes \texttt{Metre})_{ATB}$. At the other are complex viewpoints containing many pairwise links not previously thought of as performing particularly well, such as $(\texttt{Cont})_{S} \otimes (\texttt{DurRatio} \otimes \texttt{FirstInBar})_{AT} \otimes (\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase}))_{B}$. The evolution of such viewpoints has enabled finer distinctions to be made than were possible by version 1 and 2 viewpoints.

Of the viewpoints analysed here to uncover the cause of their exceptional performance in the prediction of harmony, the vast majority are new to this research. Viewpoints common to melodic and harmonic modelling which can only predict `Duration` fulfil the same roles in each case (albeit that expansion of the harmony changes the statistics). Such viewpoints which predict `Pitch` clearly change their emphasis from melody to harmony (not least because the melody is given). There are a large number of viewpoints which are specific to harmony, not least because `Cont` is not relevant to melodic modelling, and version 3 allows more than two primitive viewpoints to be linked. We have again seen many instances of predictions agreeing with intuitive or music theoretic expectations, leading to the conclusion that the selected viewpoints are performing well at the task of finding correlations of various kinds in the corpus and in individual pieces.

Chapter 9

Generation of Melody and Harmony

9.1 Introduction

Melodies and harmonies are generated event by event by randomly sampling prediction probability distributions. For the generation of harmony, a given melody acts as a template; but melodic generation is less straightforward in this respect. In this case parameters directly or indirectly defining key, time signature, starting beat, number of phrases and the number of events per phrase (all phrases are the same length for simplicity) are specified at runtime to provide a template. Generated events are used as context when generating subsequent events. The Common Lisp implementation writes completed melodies or harmonisations to output files, and a Java program converts these to MIDI files.

Random sampling and the concept of the *probability threshold*, which modifies this procedure, are explained in §9.2. Probability thresholds for the best version 0 to 3 models are optimised in §9.3. Some automatically generated melodies are presented and analysed in §9.4, while in §9.5 automatically harmonised hymn tunes of varying quality are analysed and compared. Finally, conclusions are drawn in §9.6.

9.2 Random Sampling and Probability Thresholds

Generation is achieved simply by random sampling of overall prediction probability distributions. Each prediction probability has its place in the total probability mass; for example, attribute value X having a probability of 0.4 could be positioned in the range 0.5 to 0.9. A random number from 0 to 1 is generated, and if this number happens to fall between 0.5 and 0.9 then X is generated. All required attributes are generated in this way, in the same fixed order as occurs during prediction of existing data (*i.e.*, `Duration`, `Cont` and then `Pitch`, as applicable).

It was quickly very obvious, judging by the subjective quality of generated harmonisations, that a modification to the generation procedure would be required to produce something coherent. The problem was that random sampling sometimes generated a chord of very low probability. This was bad in itself because it was likely to be inappropriate in its context; but also bad because it then formed part of the next chord's context, which had probably rarely or never been seen in the corpus. This led to the generation of more low probability chords, resulting in harmonisations of much higher cross-entropy than those typically found in the corpus (quantitative evidence supporting the subjective assessment). The solution was to disallow the use of predictions below a chosen value, the *probability threshold*, defined as a fraction of the highest prediction probability in a given distribution. This definition ensures that there is always at least one usable prediction in the distribution, however high the fraction (*probability threshold parameter*). Bearing in mind that an expert musician faced with the task of harmonising a melody would consider only a limited number of the more likely options for each chord position, the removal of low probability predictions was considered to be a reasonable solution to the problem.[1] Separate thresholds have been implemented for `Duration`, `Cont` and `Pitch`, and these thresholds may be different for different stages of generation (in versions 2 and 3). Although motivated by the desire to improve generated harmony, the introduction of probability thresholds has also proven beneficial in the generation of melody. It is hoped that as the models improve, the thresholds can be reduced.

9.3 Optimisation of Probability Thresholds

9.3.1 Version 0

The melodic work presented here was done prior to the software being updated to allow a combination of separately selected multiple viewpoint systems to be used to generate the basic attributes. Here, then, we examine the best version 0 model employing a single system to generate `Duration` and `Pitch`. Since in practice `Duration` and `Pitch` predicting subsets of this system are used, it is still possible to calculate `Duration` and `Pitch` cross-entropies separately.

Melodic model probability thresholds are optimised such that the cross-entropy of each subtask (the prediction of `Duration` and `Pitch`), averaged across the generation of twenty-one melodies, approximately matches the corresponding prediction cross-entropy obtained by ten-fold cross-validation of corpus 'A+B'. The twenty-one melodies (with a time signature of $\frac{4}{2}$) are divided into three sets of seven, in which all key signatures from 3 flats to 3 sharps are represented. One set starts on the first beat, another on the fourth and the other on the third. The resulting optimised `Duration` and `Pitch` probability threshold parameters are 0.02 and 0.115 respectively. By predicting the test data set

[1] Unfortunately, low probability predictions known to be acceptable are disallowed along with those known to be unacceptable; see §9.3.

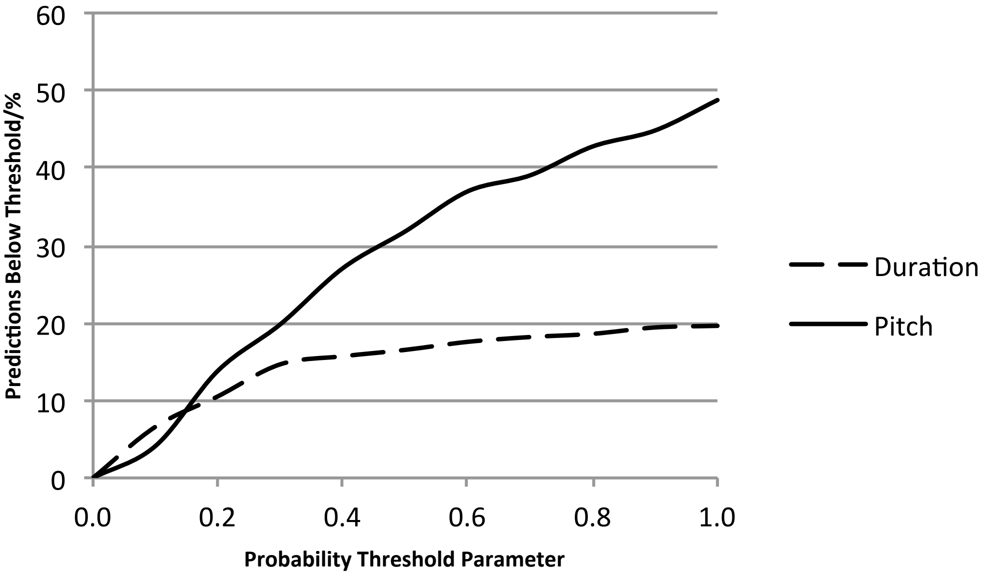


Figure 9.1: Plot of probability threshold parameter against percentage of predictions in test data set 'A+B' below the threshold for the best version 0 (melodic) model using corpus 'A+B'.

'A+B' melodies with these parameters in place, we find the percentage of `Duration` and `Pitch` predictions in the data set falling below the threshold to be 1.24 and 4.96 respectively. Overall, 3.10% predictions in the data set are below the threshold, which is not too high a price to pay for improving the quality of the generated melodies. Clearly, though, some completely appropriate note choices are being disallowed along with the inappropriate ones.

By using a range of probability threshold parameters during data set prediction, it is possible to see how the percentage of predictions below the threshold varies with the parameter. Such a plot for the version 0 model is shown in Figure 9.1.[2] There is a fairly shallow increase in `Pitch` predictions below the threshold up to a parameter of 0.1, with the curve steepening sharply after that, reaching nearly 50% at a parameter of 1. Another way of looking at this is that just over 50% `Pitch` predictions are of the highest probability in their respective distributions. The `Duration` curve initially rises more steeply than the `Pitch` one, but becomes very shallow from a parameter of 0.3 onwards. About 80% `Duration` predictions are of the highest probability in their respective distributions.

9.3.2 Versions 1 to 3

The best performing version 1 to 3 models, using a combination of separately selected multiple viewpoint systems to generate the basic attributes, are investigated here. Har-

[2] Data for the plot was gathered at probability threshold parameter intervals of 0.1, so the curve shapes are approximate.

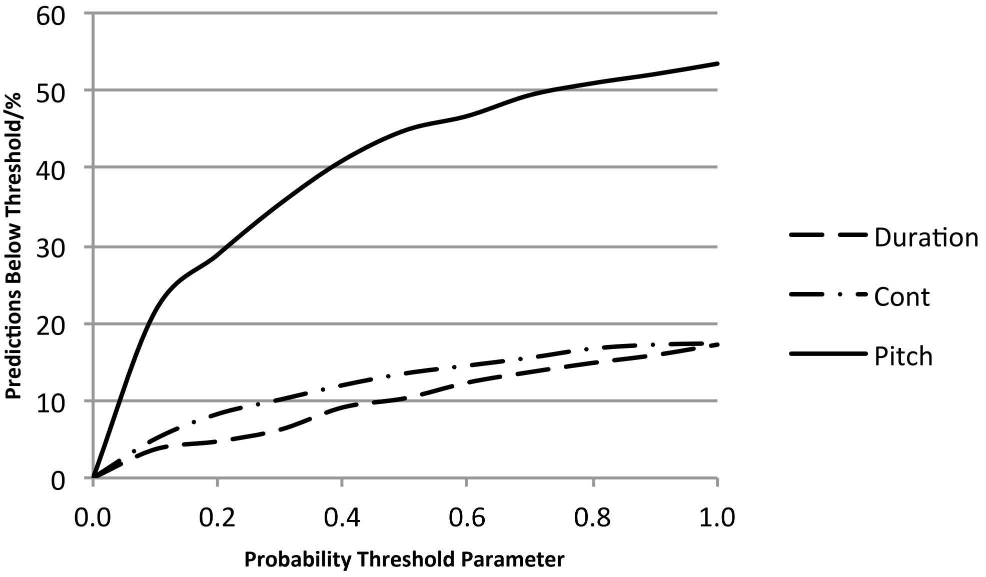


Figure 9.2: Plot of probability threshold parameter against percentage of predictions in test data set 'A+B' below the threshold for the best version 1 model (prediction of alto/tenor/bass given soprano) using the augmented `Pitch` domain and corpus 'A+B'.

monic model probability thresholds are generally optimised such that the cross-entropy of each subtask, averaged across twenty harmony generation runs using the melodies from test data set 'A+B', approximately matches the corresponding prediction cross-entropy obtained by ten-fold cross-validation of corpus 'A+B'. For version 1, this results in `Duration`, `Cont` and `Pitch` probability threshold parameters of 0.06, 0.53 and 0.045 respectively. By predicting the original test data set 'A+B' harmonisations with these parameters in place, we find the percentage of `Duration`, `Cont` and `Pitch` predictions in the data set falling below the threshold to be 2.53, 14.17 and 13.83 respectively. Overall, then, 10.18% predictions in the data set are below the threshold, which means that in seeking to avoid the generation of inappropriate and downright bad chords by employing probability thresholds, a large number of perfectly good, but low probability, chords are being rejected.

Figure 9.2 shows a plot of probability threshold parameter against percentage of predictions below the threshold for the best version 1 model. There is a very rapid increase in `Pitch` predictions below the threshold up to a parameter of 0.1, with the curve flattening out somewhat after that, but still reaching in excess of 50% at a parameter of 1. Interestingly, this means that well over 40% `Pitch` predictions are of the highest probability in their respective distributions. The `Duration` and `Cont` curves are much flatter, with less than 20% predictions below the threshold at a parameter of 1. The corollary of this is that more than 80% `Duration` and `Cont` predictions are of the highest probability in their respective distributions.

The fact that unexpanded melodies are offered for harmonisation presents a problem

	version 2		version 3	
subtask	PT param	% below PT	PT param	% below PT
`Duration` B given S	0.03	0.67	0.016	0.67
`Cont` B given S	0.96	12.98	0.73	7.08
`Pitch` B given S	0.046	3.37	0.045	2.87
`Duration` AT given SB	0.006	0.00	0.01	0.17
`Cont` AT given SB	0.6	8.26	0.74	10.12
`Pitch` AT given SB	0.111	10.96	0.1	9.11

Table 9.1: Probability threshold parameter (*PT param*) and the corresponding percentage of predicted events below the threshold (*% below PT*) are tabulated for each subtask. Versions 2 and 3 are compared.

peculiar to harmonisation in stages, which is that each stage affords the opportunity for expansion (*e.g.*, to insert passing notes). This means that the number of generated events is different for each stage; for example, in one set of version 2 runs, the average number of events was 185 higher for the alto/tenor stage compared with the bass generation stage. Consequently, the stage cross-entropies cannot, strictly speaking, be simply added to determine the cross-entropy on a per chord basis. The question then is, should the bass generation cross-entropies be reduced (by multiplying them by the ratio of generated events in each stage) prior to comparison with the prediction cross-entropies obtained by ten-fold cross-validation of the corpus? After some preliminary trials it was decided that, since it was not completely clear that this was the correct approach, no adjustment would be made in this research. For the future, consideration will be given to updating the software such that generated harmonisations can quickly and easily be predicted (like those in a test data set) to give a completely fair comparison. See Table 9.1 for version 2 probability threshold parameters and the corresponding percentage of predicted events below the threshold. Overall, 6.04% predictions in the data set are below the threshold, which is a marked improvement over version 1 (10.18%).

Figure 9.3 shows a plot of probability threshold parameter against percentage of predictions below the threshold for the best version 2 model for the prediction of bass given soprano. The first thing to notice is that the `Pitch` curve is much flatter, rising to only about 35% at a parameter of 1. A very similar story is told by Figure 9.4, which shows a version 2 plot for the prediction of alto/tenor given soprano/bass. This must mean that version 2 is better at separating acceptable predictions from unacceptable ones in terms of their probabilities, which suggests that it is better for generating harmony. This also appears to be the case, overall, for `Duration` and `Cont`.

Although version 3.2+ is best overall, it cannot at present be used for generation because the version 2 subsystems contain primitive viewpoints not yet implemented for version 3; therefore we consider here the pure version 3 two-stage model. Table 9.1 also presents probability threshold information for this version, so that a direct comparison can be made between versions 2 and 3. Probability threshold parameters are similar for

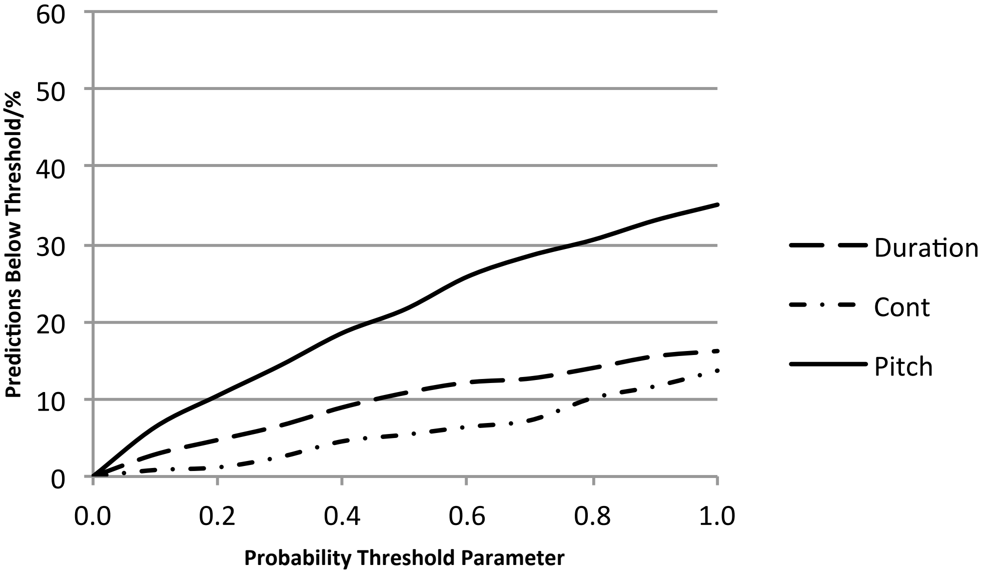


Figure 9.3: Plot of probability threshold parameter against percentage of predictions in test data set 'A+B' below the threshold for the best version 2 model for the prediction of bass given soprano using the augmented `Pitch` domain and corpus 'A+B'.

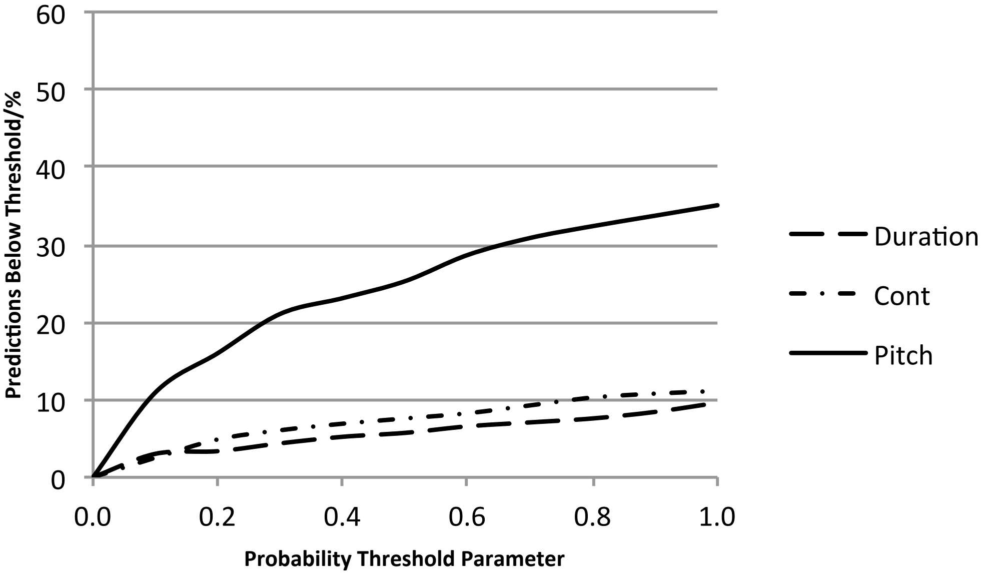


Figure 9.4: Plot of probability threshold parameter against percentage of predictions in test data set 'A+B' below the threshold for the best version 2 model for the prediction of alto/tenor given soprano/bass using the augmented `Pitch` domain and corpus 'A+B'.

Duration and Pitch, while being rather different for Cont. Overall, 5.00% predictions in the data set are below the threshold, which is a definite improvement over version 2 (6.04%).

The version 3 plots of probability threshold parameter against percentage of predictions below the threshold are fairly similar to the version 2 plots in Figures 9.3 and 9.4. The main differences are found in the alto/tenor given soprano/bass prediction stage, where for version 3 the Cont line is a little higher and the Duration and Pitch lines a little lower.

9.4 Generated Melodies

9.4.1 Generation Without Probability Thresholds

We begin by examining three melodies, nominally in F major with four phrases of eight notes each, automatically generated by the best version 0 model generating Duration and Pitch together with both probability threshold parameters set to 0, using corpus 'A+B'. The first of the melodies in Figure 9.5 displays a metrical structure akin to that of the corpus in the opening two phrases (although a second beat start to the second phrase is unusual). Most of the second half of the melody, however, is completely atypical of the corpus: it seems that once a shorter note duration has been generated, a long sequence of shorter notes then ensues, which in this case is rhythmically and metrically incoherent. On the other hand, whereas the first half of the melody fails to establish the key of F major, it is more convincingly established in the second half. The very end of the melody is both metrically and tonally acceptable. The second melody has similar faults, this time completely failing to establish the key; indeed, it finishes on an F♯. The third melody is isochronous, and largely achieves tonal coherence. Other than the B♮ in the first bar (which can actually be forgiven) the first phrase is undoubtedly in F major. The second phrase attempts a modulation to the dominant (typical of the corpus), but does not quite pull it off; and the third phrase returns solidly to F major. Unfortunately the melody concludes very strangely, with the penultimate bar comprising a descending leap of a tenth followed by two ascending leaps totalling more than an octave. The final bar then disappears into the tonal wilderness.

9.4.2 Generation With Optimised Probability Thresholds

We now investigate whether the use of optimised probability thresholds really does improve the perceived quality of the melodies; see Figure 9.6. The average cross-entropy of these melodies is 2.71 bits/note compared with 3.40 bits/note for the previous three. The first melody is a direct comparison with those in Figure 9.5. The key of F major is well established, the metrical structure is mostly good and there are no unreasonably large intervals, making this arguably the best melody we have yet seen. The most obvious flaw is the sequence of three dotted semibreves, although the syncopation in

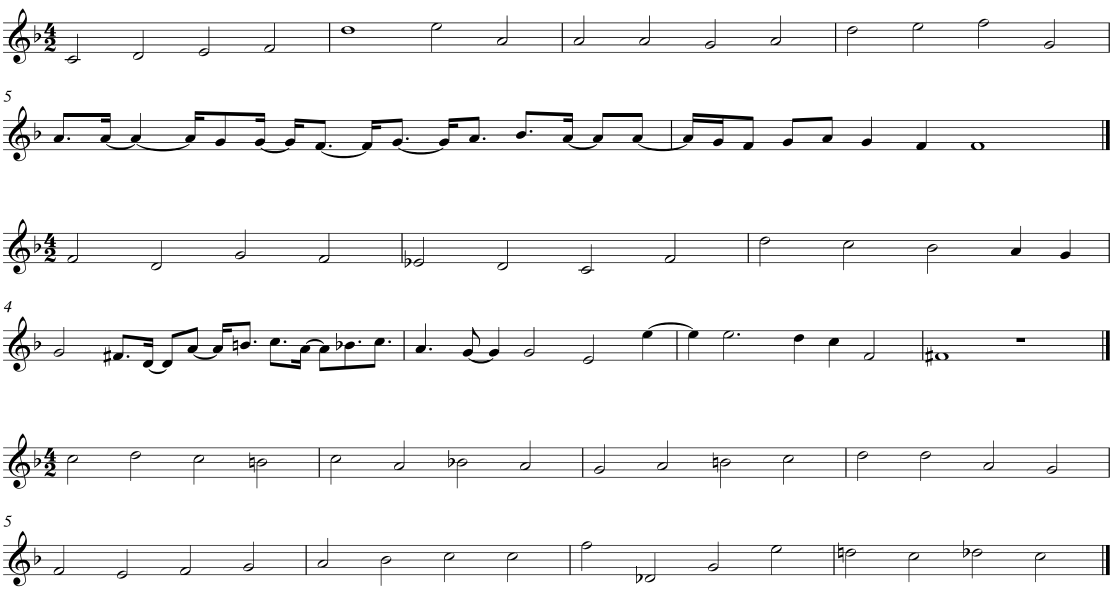


Figure 9.5: Three melodies, nominally in F major with four phrases of eight notes each, automatically generated by the best version 0 model generating `Duration` and `Pitch` together with both probability threshold parameters set to 0, using corpus 'A+B'.

the second and third bars is not exactly typical of the corpus either. The fact that the final note is the mediant rather than the tonic is unusual, but acceptable. The second melody is nominally in D major and starts on the last beat of the bar. This melody fails to establish the correct key, starting instead in A major and changing abruptly to G major in the fourth bar. The isochronous G major section proceeds almost exclusively by step, making it rather boring, while the syncopation in the second bar is atypical of the corpus. The third melody, in triple time, establishes its key of E♭ major from the start before making a legitimate modulation to the dominant. Although it loses its way after that, the melody manages to finish on the tonic. There are six instances of notes straddling a bar line, which hardly ever happens in the corpus. The overall impression, though, is that the three melodies generated using optimised probability thresholds are more rhythmically, metrically and tonally similar to the archetypes in the corpus than those generated with a threshold of zero, as expected. Having said that, none of these melodies comes close to the quality of those in the corpus.

9.5 Generated Harmonisations

9.5.1 Generation Without Probability Thresholds

Figure 9.7 nicely illustrates why it was necessary to introduce probability thresholds, showing as it does a harmonisation of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36) automatically generated by the best version 1 model with all probability threshold parameters set to 0, using corpus 'A+B'. The first four chords

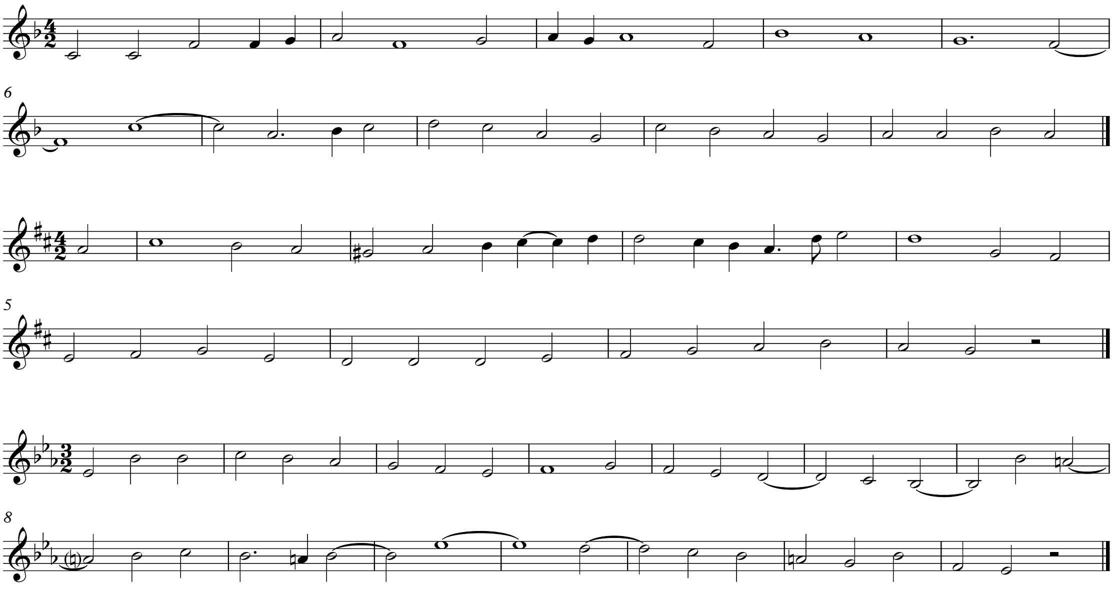


Figure 9.6: Three melodies, nominally with four phrases of eight notes each, automatically generated by the best version 0 model generating `Duration` and `Pitch` together, with their probability threshold parameters set to 0.02 and 0.115 respectively, using corpus 'A+B'.

are perfectly good; but much of the rest of the harmony is discordant. Once a low probability chord has been generated, there is a fair chance that it will not have been seen in the corpus as context for the subsequent soprano note, which tends to produce another low probability chord. There is also the possibility of extremely uncharacteristic rhythms, as exemplified by the last beat of the final bar. Similar problems arise with versions 2 and 3.

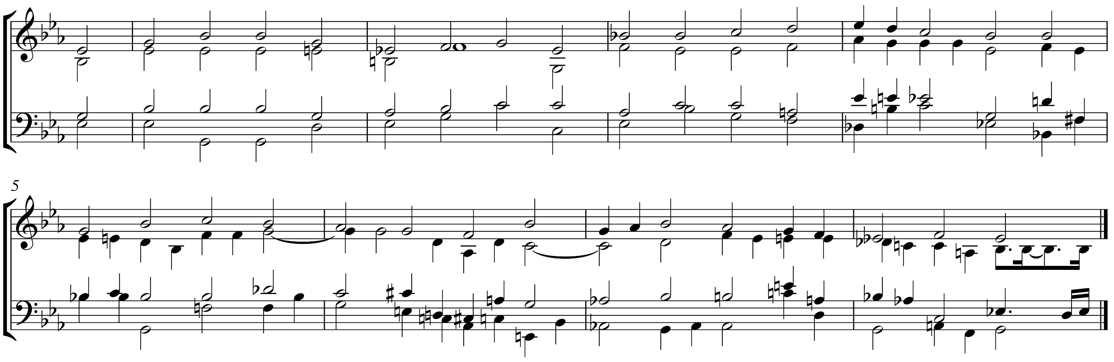


Figure 9.7: Harmonisation of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36) automatically generated by the best version 1 model with all probability threshold parameters set to 0, using corpus 'A+B'.

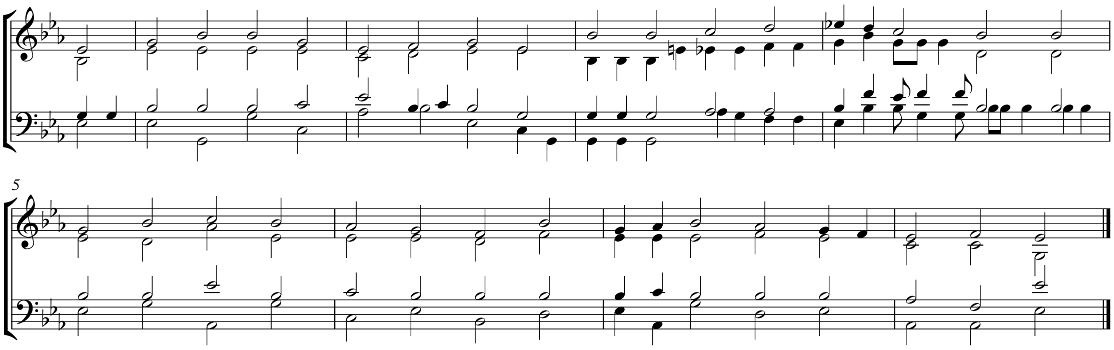


Figure 9.8: Relatively successful harmonisation of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36) automatically generated by the best version 1 model with `Duration`, `Cont` and `Pitch` probability threshold parameters set to 0.06, 0.53 and 0.045 respectively, using corpus 'A+B'.

9.5.2 Generation With Optimised Probability Thresholds

9.5.2.1 Version 1

One of the more successful harmonisations of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36), automatically generated by the best version 1 model with optimised probability threshold parameters, is shown in Figure 9.8. This is undoubtedly a huge improvement on Figure 9.7. It is far from perfect, however, with the second phrase being particularly uncharacteristic of the corpus. There are two parallel fifths in the second bar and another at the beginning of the fourth bar. The bass line is not very smooth, due to the many large ascending and descending leaps; but this situation could be improved simply by changing the octave of some of the bass notes. A new viewpoint, based on `ScaleDegree` but able to distinguish between octaves, was suggested in §7.2. It is likely that this viewpoint will prove as beneficial in the modelling of harmony as it is expected to be in the modelling of melody.

At the other end of the spectrum, one of the less successful harmonisations of this hymn tune is presented in Figure 9.9. The first phrase is a complete mess rhythmically, and not very good harmonically. There is a big improvement after that; but the harmony is not as good as in Figure 9.8 (see, *e.g.*, the second beat of the fourth bar). There is a parallel octave at the beginning of the second phrase, a parallel fifth at the end of the fifth bar, and the piece concludes badly in the relative minor. Having said all that, the harmony is still much better than that of Figure 9.7. The use of optimised probability thresholds definitely improves the quality of the harmony, although the harmonisations still fall a long way short of the standard of the archetypes.

9.5.2.2 Version 2

One of the more successful harmonisations of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36), automatically generated by the best version 2 model with

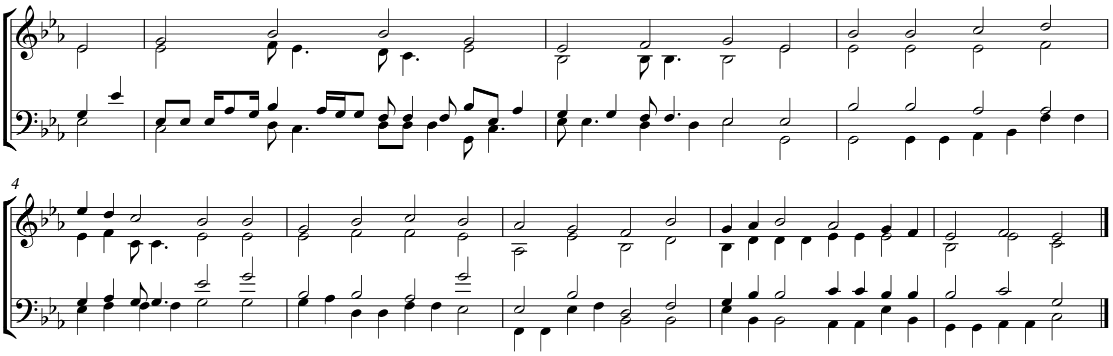


Figure 9.9: Relatively unsuccessful harmonisation of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36) automatically generated by the best version 1 model with `Duration`, `Cont` and `Pitch` probability threshold parameters set to 0.06, 0.53 and 0.045 respectively, using corpus 'A+B'.

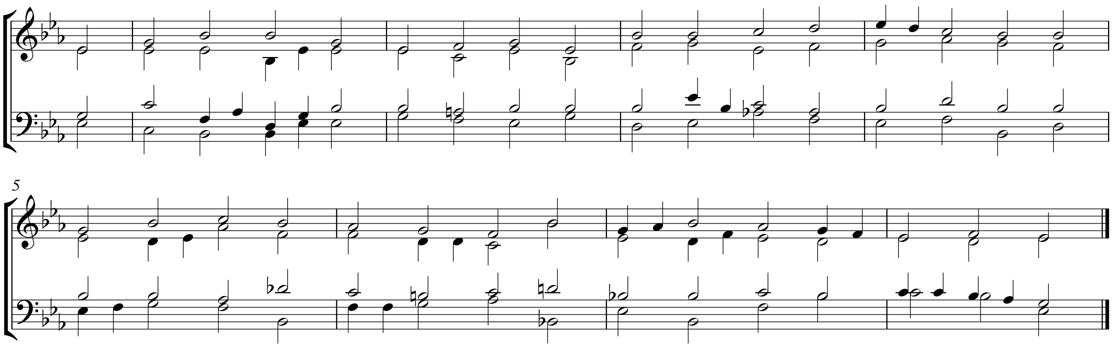


Figure 9.10: Relatively successful harmonisation of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36) automatically generated by the best version 2 model with optimised probability threshold parameters, using corpus 'A+B'.

optimised probability threshold parameters, is shown in Figure 9.10. The first thing to notice is that the bass line is more characteristic of the corpus than anything we have yet seen. This could well be due to the fact that this version employs specialist systems for the prediction of bass given soprano. It is rather jumpy in the last phrase, however, and in the final bar there is a parallel unison with the tenor. The second chord of the second bar does not fit in with its neighbouring chords, and there should be a root position tonic chord on the third beat of the fourth bar. On the positive side, there is a fine example of a passing note at the beginning of the fifth bar; and the harmony at the end of the third phrase, with the chromatic tenor movement, is rather splendid. Overall, this is consistently better harmony than we saw in §9.5.2.1.

This version is also capable of generating some very poor harmony indeed. We see in Figure 9.11 that the bass line hovers around a single note for much of the first phrase, and at the beginning of the third bar there is a parallel unison with the tenor. There is an upward leap of nearly two octaves in the bass, followed immediately by a downward leap of a similar magnitude, in the fourth bar. In the second half of the piece, however, the bass line is stylistically appropriate. On the whole, the four-part harmony in the

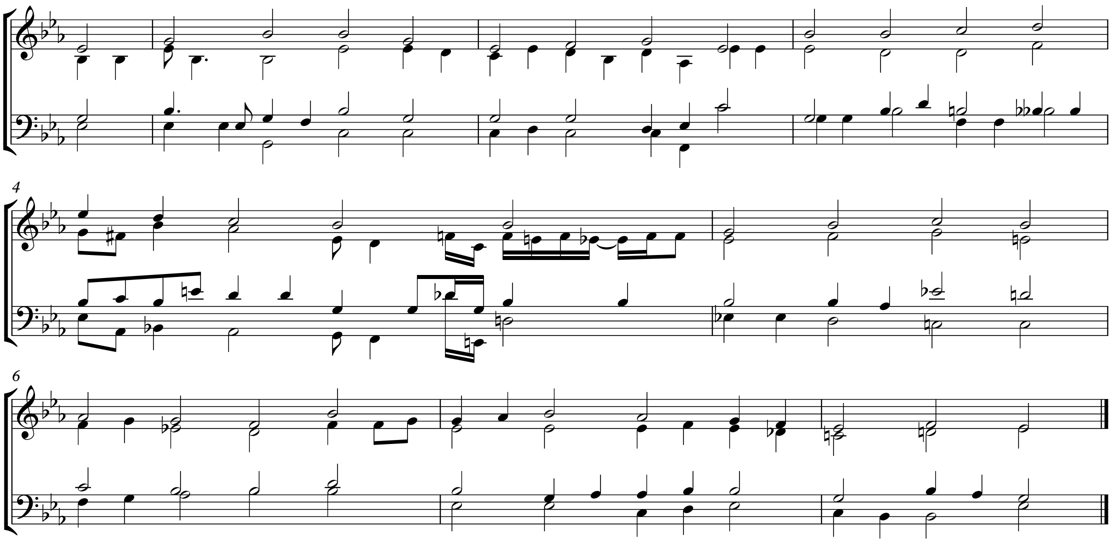


Figure 9.11: Relatively unsuccessful harmonisation of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36) automatically generated by the best version 2 model with optimised probability threshold parameters, using corpus 'A+B'.

first four bars is dreadful. For example, there is a parallel octave in the second half of the first bar; the dissonance at the third beat of the third bar is particularly nasty; and chaos ensues in the second half of the fourth bar. The four-part harmony in the final four bars matches the style of the corpus much more closely, although the dissonance of the last chord of the fifth bar is unsuitable and there is a parallel octave at the beginning of the sixth bar.

9.5.2.3 Version 3

The best version 3 model generates the bass first, followed by alto/tenor. One of the more successful harmonisations of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36), automatically generated by this model with optimised probability threshold parameters, is shown in Figure 9.12. The bass line is again mostly characteristic of the corpus. Worthy of mention in this regard is the contrary motion between soprano and bass at, for example, the beginning of the seventh bar. There is a good deal wrong with the harmony, beginning with the rather odd D♭ followed by a large descending leap in the tenor in the first bar. There is a discord on the second beat of the third bar followed by a parallel fifth in the soprano and alto. There is no proper cadence at the end of the second phrase, with discords persisting over the phrase boundary. There are parallel unisons in the tenor and bass from the middle of the fifth bar until the beginning of the sixth; and there is a parallel octave between the soprano and bass on the fourth beat of the penultimate bar. On the other hand, three of the four cadences are stylistically valid (the rather unusual first cadence, with the tenor overlapping the bass, is found in Bach). This is certainly no better, and arguably slightly worse, than the equivalent version 2 harmonisation in Figure 9.10.

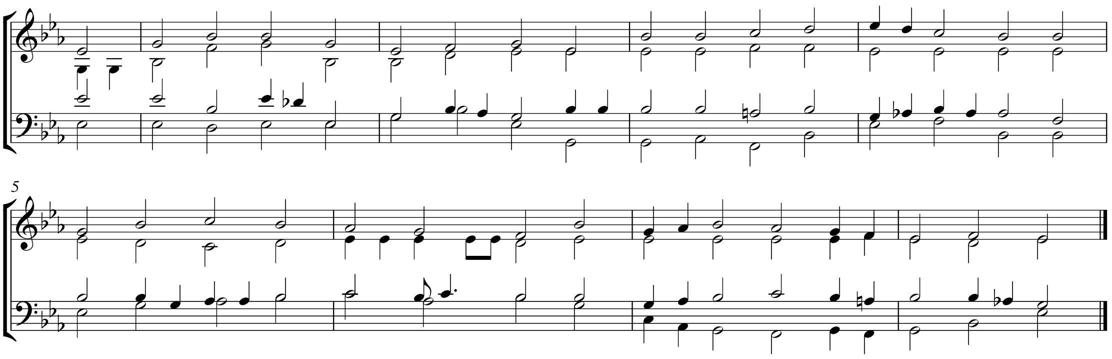


Figure 9.12: Relatively successful harmonisation of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36) automatically generated by the best version 3 model with optimised probability threshold parameters, using corpus 'A+B'.

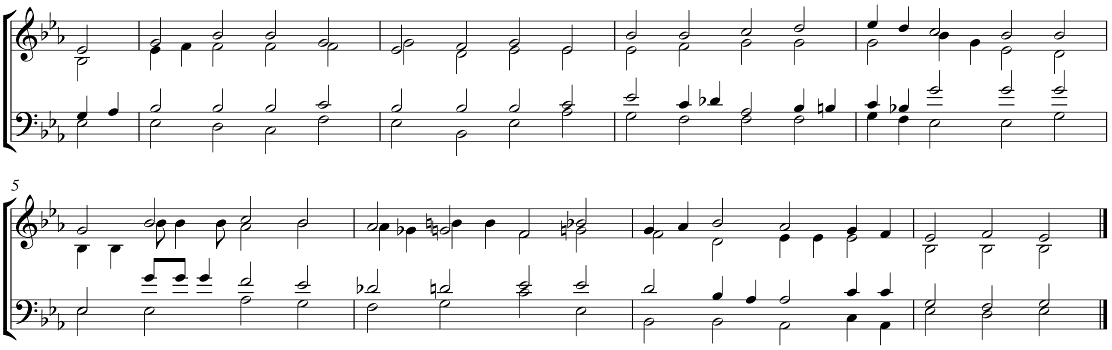


Figure 9.13: Relatively unsuccessful harmonisation of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36) automatically generated by the best version 3 model with optimised probability threshold parameters, using corpus 'A+B'.

In the relatively unsuccessful harmonisation of Figure 9.13, the second half of the first bar is discordant. This is followed by a chord in which the melody is obscured by a crossing alto part. Much of the second phrase is harmonically chaotic and it fails to end in a proper cadence. The second beat of bar five is rhythmically uncharacteristic; and the final three beats of the third phrase contain chords which are completely unrelated, again failing to form a cadence. The final phrase, which contains a parallel octave between the soprano and bass, is harmonically weak throughout. One positive thing which can be said about this harmonisation is that it is far less rhythmically chaotic than the unsuccessful version 2 one shown in Figure 9.11.

9.6 Conclusion

The generation of melody and harmony by means of random sampling was explained early in this chapter, including the use of probability thresholds to modify the procedure. Probability thresholds were required because music with a high cross-entropy (and subjectively low quality) was being consistently generated. By optimising the probability thresholds in §9.3 such that mean cross-entropy of generation for each subtask approximately matched that of the prediction of the corpus (using ten-fold cross-validation), it was subsequently possible to generate melody and harmony of higher quality which was more amenable to comparison.

An interesting finding arising from the work on probability thresholds is that when predicting a test data set, the vast majority of predictions are of the highest probability. This situation improves when moving from a single stage of prediction (version 1) to prediction in more than one stage (versions 2 and 3). In addition, the percentage of predictions below optimised thresholds falls from 10.18 for version 1 to 5.00 for version 3. This suggests that version 3 is particularly good at separating appropriate from inappropriate predictions in terms of their probabilities, which in turn indicates that it should be able to generate better harmony.

Whereas the improvement to the generated melodies was immediately obvious, the quality of the generated harmony was not consistently high; therefore harmonisations at either end of the spectrum were analysed for each of versions 1, 2 and 3. The qualitative evidence presented in §9.5.2 points to versions 2 and 3 being capable of producing better harmony than version 1, partially corroborating the quantitative evidence of Chapter 6. The evidence also suggests that version 2 can create better and worse harmony than version 3; that is, version 3 appears to produce more consistent results. It should be borne in mind, however, that it was possible to present only a tiny sample of generated harmonisations here; so the results of this chapter should be regarded as indicative only.

The fact that it is not possible to consistently generate high quality music in the style of the corpus is ample evidence that the models are not yet good enough. Indeed, it was never expected that the ultimate harmonisation model would be found amongst versions 1 to 3. With this in mind, the following chapter outlines further ways in which these models may be improved. In the meantime, the current models can be coerced into generating music of higher quality more consistently by ramping up the probability threshold parameters. This has the effect of creating music of generally lower cross-entropy than the corpus, with the result that there is less variety in different harmonisations of the same melody. To the listener, the music can become more predictable and therefore less interesting. Appendix F contains harmonisations produced using probability threshold parameters set to 1, the most extreme case, where generation becomes deterministic.

Chapter 10

Towards an Improved Music Modelling Framework

10.1 Introduction

The main goal of this research was to model the harmony of four-part non-homophonic music. A plethora of modelling techniques, both existing and devised during this research, could have been employed in pursuit of this goal; but time limitations dictated that compromises had to be made with regard to what was implemented. It is intended that in the future additional techniques, along with new and improved viewpoints, will be incorporated into the system in an attempt to improve its performance and demonstrate the worth of the techniques. In addition, some issues have yet to be resolved, making future research an interesting challenge.

Various ways in which versions 0 to 3 may be improved are detailed in §10.2. This is followed in §10.3 by an exposition of versions 4 to 9, which further extend the multiple viewpoint framework. Finally, the chapter is summarised in §10.4.

10.2 Versions 0 to 3

Various topics are discussed in this section on versions 0 to 3 including viewpoints, attribute prediction, model combination, viewpoint selection and representation.

10.2.1 Viewpoints

New viewpoints and potential improvements to existing viewpoints are proposed here for future implementation and investigation.

ExtendedScaleDegree We saw in §7.2 that viewpoint `ScaleDegree` could be improved by linking it with `Tessitura`, which affords a means of distinguishing between octaves. A new viewpoint was suggested, now named `ExtendedScaleDegree`, which is potentially even better than `ScaleDegree` ⊗ `Tessitura`. Like `ScaleDegree`, its values are relative to the tonic. Unlike the mod 12 `ScaleDegree`, however, this reference note is fixed at some MIDI value lower than the lowest note in the domain, and all notes above it have a unique value. Each key needs its own lowest MIDI value, which is subtracted from `Pitch` to produce the new viewpoint. This is a far more fine-grained viewpoint which always matches the melodic contour. For version 1 onwards, it retains the true vertical intervals within a chord. Its formal version 0 mathematical definition is:

$$\Psi_{\texttt{ExtendedScaleDegree}}(e_1^j) = \Psi_{\texttt{Pitch}}(e_1^j) - \Psi_{\texttt{Tonic}}(e_1^j).$$

PrimaryFirstInBar As suggested in §7.6, the success of `Interval` ⊗ `FirstInBar` is probably due more to a correlation between interval and phrase beginnings rather than to a more general correlation involving the beginnings of bars. When there is no anacrusis, `Interval` ⊗ `FirstInBar` and `Interval` ⊗ `FirstInPhrase` are both equipped to model the tendency for an interval larger than a major second to occur between phrases (albeit that the probability distributions of the former viewpoint are contaminated by first in bar data from elsewhere in the phrase). When there is an anacrusis, `Interval` ⊗ `FirstInBar` loses the ability to model this tendency (since the interval occurs before the first in bar), but instead gains the capability of capturing the arguably stronger tendency for a pitch leap to occur between an anacrusis and the first beat of the following bar (an ability which does not exist in `Interval` ⊗ `FirstInPhrase`). The suggested new test viewpoint to identify the primary first in bar of a phrase, now called `PrimaryFirstInBar`, may be better at modelling these tendencies when linked with `Interval`.

TactusDuration When using statistical models involving `Duration` and `DurRatio` to generate melodies or harmonies, it is very easy, for example, for a switch from a long sequence of minims to one of crotchets (or *vice versa*) to occur, which is atypical of the corpus. This is because once such a duration change has been established, the new durations form the N-gram contexts employed in prediction. A new viewpoint based on tactus beat length would obviate this problem. The formal definition of `TactusDuration` is:

$$\Psi_{\texttt{TactusDuration}}(e_1^j) = \Psi_{\texttt{Duration}}(e_1^j) \times \frac{\Psi_{\texttt{Pulses}}(e_1^j)}{\Psi_{\texttt{BarLength}}(e_1^j)}.$$

Tempo The speed at which music is performed has a strong influence on the range of note values which can be sensibly used. Figure 10.1 presents excerpts from the second and fourth movements of Beethoven's Pianoforte Sonata in F minor (Op. 2, No. 1), beginning at bars 56 and 25 respectively. The shortest note values in the entire slow movement (Adagio) are triplet demisemiquavers, whereas in the fast movement (Prestissimo) they are triplet quavers. It is probably physically impossible to perform triplet demisemiquavers adequately at the faster tempo, especially more than a few at a

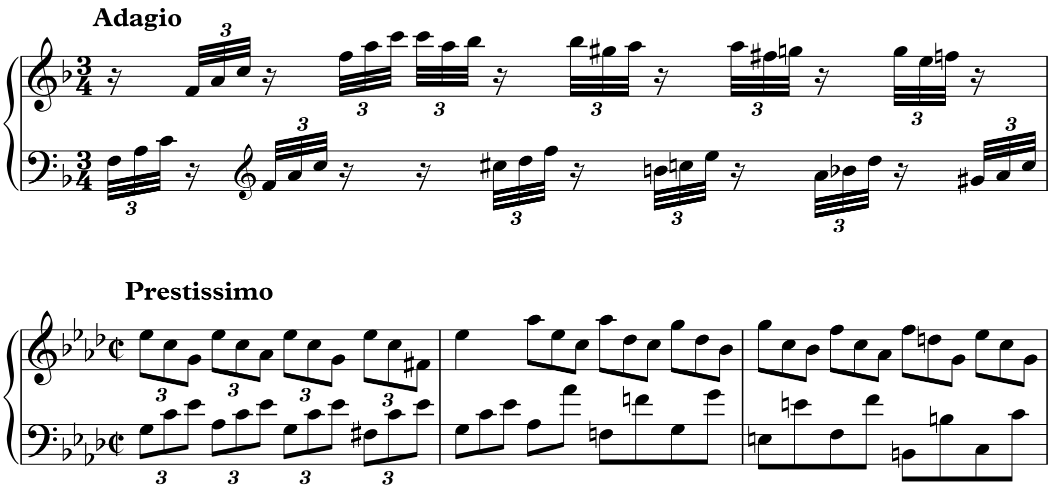


Figure 10.1: Excerpts from the second and fourth movements of Beethoven's Pianoforte Sonata in F minor (Op. 2, No. 1), beginning at bars 56 and 25 respectively. No phrasing, dynamics or pedal indications are shown. Notice that the shortest note values in the entire slow movement (Adagio) are triplet demisemiquavers, whereas in the fast movement (Prestissimo) they are triplet quavers.

time. So far, tempo has been ignored by the implemented modelling process; therefore a new basic viewpoint is required to enable this attribute to be taken into account, which has unsurprisingly been named `Tempo`. Using the raw metronome mark would be far too fine-grained, so it is proposed to use existing tempo ranges, commonly described by terms such as Andante, Allegro and Lento.

Harmony Harmonic function symbols such as Vb (dominant chord in first inversion) and the related figured bass (most often used to indicate harmonic structure to a continuo keyboard player) have been important tools in the teaching and practice of harmony for a very long time (*e.g.*, Piston, 1976). In view of this, it is unsurprising that harmonic function symbols have been used extensively in previous related research (Allan, 2002; Biyikoğlu, 2003; Clement, 1998; Ponsford et al., 1999); it therefore seems appropriate that a viewpoint employing a domain of such symbols should be investigated in conjunction with existing and other proposed new viewpoints. Let us consider how chords may be appropriately labelled. One option is to annotate the corpora by hand, which of course would be very time consuming, while another is to use an existing chord labelling program from other research; such a program would very likely discriminate finely between chords, resulting in a large domain. A third option is to use text files from Bach (1998) as the corpus, as they are already annotated with chord labels. It would be preferable, however, to retain the existing corpora for comparison with work already done. The favoured option at present is to define `Harmony` in terms of simple rules which map sets of notes in simultaneities to a domain of root position harmonic function symbols. It is reasonable to use root position symbols, since chord inversion information can be inferred from a combination of root position symbol and (separately predicted) bass note; this results in a much smaller domain compared with the domain of harmonic symbols containing inversion information.

Metre The current definition of `Metre` is somewhat arbitrary. As indicated in §7.2, it is conceivable that its definition could be improved by experimenting with different values for the various metrical positions in the bar, across time signatures.

Duration, LastInPhrase and LastInPiece The way in which `Duration`, `LastInPhrase` and `LastInPiece` are currently handled leads to inconsistencies at phrase and piece endings during the generation of harmony, as described in §3.4.4. A solution to this problem must be sought during future work.

`IOI`$\ominus$`FirstInBar` and `IOI`$\ominus$`Tactus` It was noted in §7.1 that software implementation errors affecting `IOI` $\ominus$ `FirstInBar` and `IOI` $\ominus$ `Tactus` (applicable to versions 0, 1 and 2) were discovered at a very late stage. In each case, models are constructed using the threaded `IOI` value, whereas prediction probability distributions are calculated using the local `IOI` value. The software will be corrected as a matter of urgency in future work.

Enharmonic Equivalence Although the objective of this research is to construct models of musical style, the pool of primitive viewpoints used is drawn largely from earlier work on the cognitive modelling of music. In the latter case, dealing with music solely as it is heard and assuming a tempered tuning, there is no need to distinguish between enharmonically equivalent notes such as G♯ and A♭. In this research, on the other hand, derived viewpoints which are able to distinguish between such notes may well prove beneficial, and will be seriously considered in future work.

10.2.2 Attribute Prediction

Currently, the attribute prediction order is fixed at `Duration` followed by `Cont` followed by `Pitch`. In future work, however, it is intended that the order of attribute prediction can be varied. After all, a person listening to a note is able to comprehend its pitch before it ends (assuming that the note is long enough for pitch cognition to occur). Experimentation will reveal whether or not the current prediction order is optimal. An alternative to predicting one attribute at a time, at the other extreme, is to make one prediction of all attributes at the same time. These extremes can be contrasted and compared in future research.

10.2.3 Model Combination

10.2.3.1 N-gram Model Combination

The current implementation employs back-off smoothing with escape method C and exclusion within the PPM framework (Cleary and Witten, 1984) to combine N-gram models of various order into a viewpoint model. This was good enough, given time constraints, for the comparisons made to date; but to construct the best models possible, we should take account of the findings of Pearce (2005) with respect to melody. He found that the best LTM (using the single viewpoint system {`Pitch`}), averaged over all data sets, employed interpolated smoothing using escape method C without exclusion. At 2.957 bits/note, this was slightly better than the 3.025 bits/note for back-off smoothing with escape method C and exclusion; therefore both interpolated smoothing and the option of not using exclusion should be made available in future. On the other hand, the best STM employed interpolated smoothing using escape method AX with exclusion, which at 3.145 bits/note was slightly better than the 3.192 bits/note for back-off smoothing with escape method C and exclusion; therefore escape method AX (at least) should also be implemented. Finally, Pearce (2005) found that the use of the unbounded order PPM* also improved these models (to 2.878 and 3.139 bits/note respectively), which suggests that the PPM* option should be made available in future.

There are possible alternatives to back-off and interpolated smoothing for the determination of viewpoint model prediction probability distributions. Pearce (2005) uses a method based on a weighted geometric mean to combine viewpoint model distributions into a single distribution, and he uses the same method to combine long- and short-term model distributions. The method was found to be beneficial to the performance of these higher level models, so it is conceivable that it could also benefit lower level ones. It is anticipated that the application of this method to the combining of N-gram models within the PPM framework will lead to the production of sharper (*i.e.*, less uniform) combined distributions. Whether or not these sharper distributions result in lower cross-entropy models remains to be seen. The use of this combination method requires completed distributions for each of the model orders employed. One possible way of achieving this is to immediately back off to the uniform distribution in each case. The weighted arithmetic mean method used by Conklin (1990) was found by Pearce (2005) to combine viewpoint distributions less well; for completeness, however, it will also be tried within the PPM framework.

10.2.3.2 Viewpoint Model and LTM/STM Combination

Limitations to this implementation mean that the same combination method (*i.e.*, weighted geometric or arithmetic) and bias are used within the LTM and STM, and the same combination method is used both within and between the LTM and STM. It is intended that more flexibility in this area will be introduced in the future.

10.2.4 Different Viewpoints in LTM and STM

In the same way that the use of separately selected multiple viewpoint systems for the prediction of `Duration` and `Pitch` can enhance performance (see, *e.g.*, §5.3.3), it is expected that the use of different systems for LTM(+) and STM in BOTH(+) could result in an improvement. Preliminary work using version 1 models indicates that combining completely separately selected LTM and STM does not produce a better model; but it is conceivable that selecting an STM given an already selected LTM could improve performance (*i.e.*, the LTM is taken into account during the selection of the STM such that complementary systems emerge).

10.2.5 Viewpoint Selection

At present, version 0 viewpoint selection begins with `Duration` and `Pitch` already in the multiple viewpoint system (for models predicting both of those attributes). These viewpoints can be deleted as other viewpoints are added, if doing so reduces the cross-validation cross-entropy. This approach violates the general principle of adding the best viewpoint at each stage, however. It is conceivable, for example, for `Pitch` to end up in the selected system not because it was particularly a good viewpoint, but because it could not be deleted without making at least one note impossible to predict. An obvious way around this is to begin the selection process by repeatedly replacing one of the basic viewpoints with other primitive viewpoints able to predict the same attribute for all of the notes. The one which results in the lowest cross-entropy is chosen, and then the process is repeated for the other basic viewpoint. Similar procedures apply to versions 1, 2 and 3.

Viewpoint selection is currently performed using fixed bias values which are optimised afterwards. The effect of instead optimising the biases after each round of addition and deletion can be investigated. It is very likely that the search path and ultimate multiple viewpoint system will be different, with the latter hopefully performing better.

It is recognised that the approach to viewpoint selection used in this research, which builds up viewpoints in rather the same way as Markov random field feature selection (see §2.5.3.3), is a form of hill climbing which can find a locally optimal solution, but is not guaranteed to find a globally optimum one. An approach which is less time efficient, but has a more comprehensive (although not exhaustive) search capability, will now be briefly described. Instead of adding one viewpoint to a single viewpoint set at each iteration, the best (say) five additions can be taken into account by creating five different viewpoint sets from each original one. This is a parallel, branching search of the space of possible multiple viewpoint systems; the amount of branching can be reduced as the depth of the search increases in order to achieve completion in a reasonable time. At the conclusion of the search, the best performing of the systems at the leaves of the search tree is chosen.

It is highly likely that for version 3 in particular, the introduction of a branching search strategy without any other modification will result in impractically long processing times. A means of speeding up viewpoint selection is described in §3.4.5.3; that is, the removal from further consideration any viewpoint which, when added to a system, increases the cross-entropy above a certain margin. Defining δ as the improvement in cross-entropy over the last round of additions and deletions, a margin equal to the smaller of 0.04 or 1.7δ bits/symbol was settled upon by trial and error for versions 0 to 2. In an attempt to further speed up viewpoint selection for version 3, the margin in this case was reduced to the smaller of 0.01 or 0.4δ bits/symbol. It is likely that the margin can be further reduced considerably without adversely affecting the outcome too much, especially in conjunction with the branching search strategy, the effectiveness of which, it is hoped, will outweigh any decline in performance due to the decreased margin.

Another simple way to improve time efficiency (not necessarily a good strategy for branching search) is to add more than one viewpoint to a multiple viewpoint system at a time. The safest way to do this is to add viewpoints which predict different attributes; for example, if the viewpoint which lowers the cross-entropy the most is able to predict `Cont` and `Pitch`, then the best of the viewpoints predicting `Duration` only is also added. In this case, more than one round of deletion may be required before reverting to addition. The addition of several viewpoints with overlapping prediction capabilities can also be investigated.

Finally, the criterion for curtailing viewpoint selection (see §5.2.6) can undoubtedly be improved. At present, viewpoint selection ends when a viewpoint addition results in a reduction in cross-entropy of < 0.0015 bits/prediction. Unfortunately, this rather crude criterion discriminates against threaded viewpoints such as `ScaleDegree` $\ominus$ `LastInPhrase` which, while possibly significantly increasing the probability of a few structurally important predictions, may achieve a negligible reduction in cross-entropy. The solution to this is to base the measured reduction in cross-entropy only on the predictions that any given viewpoint is involved with. In this way, it is hoped that music generated by such improved models will sound better structured.

10.2.6 Viewpoint Model Representation

A three-dimensional representation was introduced in §3.2.4.3 to aid visualisation of the way in which linked viewpoints may be constructed. In that representation, the four layers corresponded with the four parts in the harmonic texture (SATB). It was noted, however, that additional layers for viewpoints common to or relying on all four parts could be envisaged: see Figure 10.2, which is the same as Figure 3.3 except for the addition of a single common layer. The proposed new viewpoint `Harmony` is shown on this layer along with `Tonic`, `BarLength` and `KeySig`. Other viewpoints may also be suitable for inclusion on this or other common layers. There are issues to be resolved with respect to the number of such layers and how viewpoints are to be distributed between them, however. The solution may differ according to the order of prediction of attributes, to adhere to the already established principle (in version 2) of completely predicting a layer before proceeding to the prediction of other layers. It is conceivable that only `Harmony` will sit on its own layer.

10.2.7 Minor Keys

The current implementation uses corpora of music in a major key only (although this does not rule out excursions into minor keys). To make the system more general, it is necessary to include minor key melodies/harmonisations in the corpora, and to be able to predict/generate music in minor keys. Consideration will be given to allowing the linking of three primitive viewpoints when `Mode` is a constituent: since the value of `Mode` is given (not predicted) in this research, its presence in a linked viewpoint will distinguish between major and minor without increasing the size of the prediction domain.

10.2.8 Rests

Rests are problematic, as they are not considered to be events within the current viewpoint formulation; since rests are not common in hymn tunes and their harmonisations, however, music containing rests has thus far simply been excluded from this research. In order to make the implementation more generally applicable, however, this issue must be addressed.

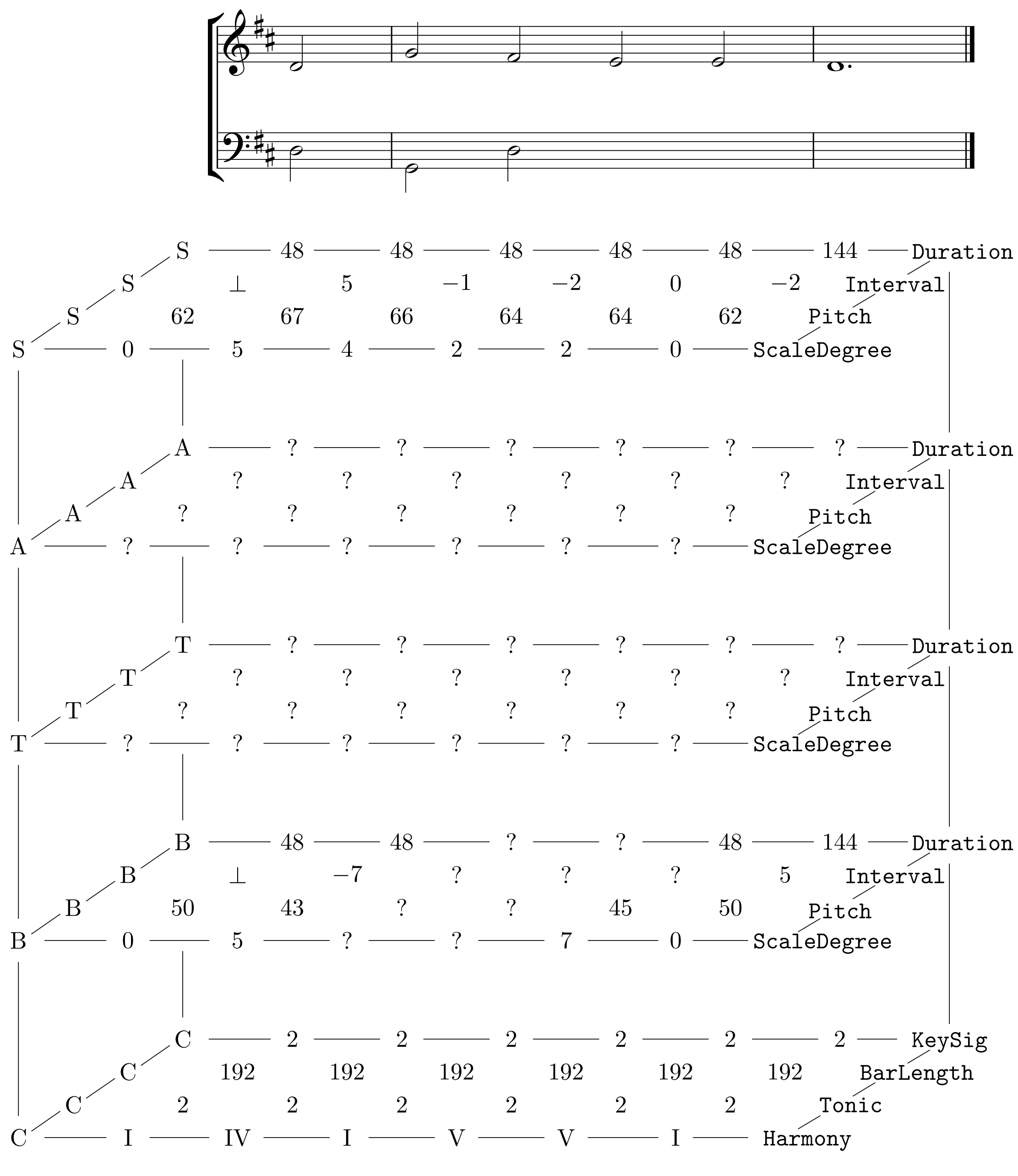
10.2.9 Version 2

As it stands, version 2 breaks the harmonisation task into a limited number of subtasks; that is, the various layers can be predicted in more than one pass (*e.g.*, bass first followed by alto and tenor together). A subtask structure is hypothesised for future consideration which breaks each pass into three subtask models: the first deals with cadences (the usual first step for musicians engaged in harmonising a melody); the second is concerned with all chords except the cadence and a few pre-cadence chords in each phrase; and the third bridges the gap by specialising in the pre-cadence chords.

10.2.10 Version 3

At present, to avoid too much complexity, prediction layers are assigned the same viewpoint, whereas support layers may have any combination of viewpoints (see §3.4.3). Consideration will be given to relaxing the prediction layer restriction, which could conceivably result in better (but more complex) models.

Since primitive and threaded viewpoints `IOI` $\ominus$ `FirstInBar`, `Interval` $\ominus$ `FirstInBar`, `ScaleDegree` $\ominus$ `FirstInBar`, `Contour` $\ominus$ `Tactus`, `Pitch` $\ominus$ `Tactus`, `InScale` and `Tessitura` appear in the best version 1 and 2 systems (as noted in §6.9), it is likely that the version 3 models can be improved by adding these viewpoints to the restricted version 3 pool.



S — 48 — 48 — 48 — 48 — 48 — 144 — Duration
S ⊥ 5 −1 −2 0 −2 Interval
S 62 67 66 64 64 62 Pitch
S — 0 — 5 — 4 — 2 — 2 — 0 — ScaleDegree

A — ? — ? — ? — ? — ? — ? — Duration
A ? ? ? ? ? ? Interval
A ? ? ? ? ? ? Pitch
A — ? — ? — ? — ? — ? — ? — ScaleDegree

T — ? — ? — ? — ? — ? — ? — Duration
T ? ? ? ? ? ? Interval
T ? ? ? ? ? ? Pitch
T — ? — ? — ? — ? — ? — ? — ScaleDegree

B — 48 — 48 — ? — ? — 48 — 144 — Duration
B ⊥ −7 ? ? ? 5 Interval
B 50 43 ? ? 45 50 Pitch
B — 0 — 5 — ? — ? — 7 — 0 — ScaleDegree

C — 2 — 2 — 2 — 2 — 2 — 2 — KeySig
C 192 192 192 192 192 192 BarLength
C 2 2 2 2 2 2 Tonic
C — I — IV — I — V — V — I — Harmony

Figure 10.2: A three-dimensional representation of a partial harmonisation of the final phrase of hymn tune *Tallis' Ordinal* (Vaughan Williams 1933, hymn no. 453). The bottom layer is "common."

10.2.11 Direct Comparison With Previous Work

In future work, the corpus of Bach chorale harmonisations that Allan (2002) used in his HMM work will also be used with this multiple viewpoint implementation (as an alternative to, rather than a replacement of, the extant corpora) so that direct comparisons between the systems can be made. For example, music students could be played output from each system, and asked: "Which one is most like Bach?" or "Which one is by Bach?"

10.3 Versions 4 to 9

The first three versions of the multiple viewpoint framework for harmony, implemented and investigated in this research, are described in §3.4. A further six increasingly complex versions are detailed below with a view to future implementation.

10.3.1 Version 4: An Alternative Inter-layer Linking Method

So far (in versions 1 to 3), we have assumed that the i^{th} element of a sequence is linked with the i^{th} element of another sequence. This remains the case for intra-layer linking, but for inter-layer linking it is well worth considering an alternative. Allan (2002) did some studies using Markov chains with additional context; but the additional context was temporally placed as close as possible to what was being predicted, resulting in for example harmonic symbol $n-1$ being linked (in our parlance) to melodic symbol n. Smoothed models employing such contexts outperformed models using melodic context only and harmonic context only. A typical N-gram would look like this (the viewpoint name and a formal representation of the context are also shown):

$$\begin{array}{ccccc} & 5 & 4 & 2 & 2 \\ \mathrm{I} & \mathrm{IV} & \mathrm{I} & ? & \\ n-3 & n-2 & n-1 & n & n+1 \end{array}$$

$$(\texttt{ScaleDegree})_S \otimes (\texttt{Harmony})_{Cp}$$

$$\begin{aligned}\{&\langle \text{soprano}, -2, \texttt{ScaleDegree}, \langle 5\rangle\rangle, \langle \text{soprano}, -1, \texttt{ScaleDegree}, \langle 4\rangle\rangle,\\ &\langle \text{soprano}, 0, \texttt{ScaleDegree}, \langle 2\rangle\rangle, \langle \text{common}, -3, \texttt{Harmony}, \langle \mathrm{I}\rangle\rangle,\\ &\langle \text{common}, -2, \texttt{Harmony}, \langle \mathrm{IV}\rangle\rangle, \langle \text{common}, -1, \texttt{Harmony}, \langle \mathrm{I}\rangle\rangle\}.\end{aligned}$$

This makes sense, since an event occurring at the same time as the event to be predicted is likely to be a good predictor of it. An important consequence of this arrangement is that there is now a limited look-ahead capability; in this example, only those elements with a $(\texttt{ScaleDegree})_S$ value of 2 at position $n+1$ will be admitted to the prediction probability distribution. Constructing a model which looks ahead is thought to be acceptable, for reasons given in the description of version 5 in §10.3.2 below. Another consequence is that a start symbol is required at the beginning of the prediction layer, in order to make use of the first symbol of the support layer(s) when predicting the first real symbol of the prediction layer; and similarly a finish symbol is required at the end of the support layer(s) so that the final symbol of the prediction layer can be predicted. Alternatively, wild card symbols can be linked with symbols at the extremes of the sequence, such that any symbol from the relevant domain matches a wild card symbol; but since multiple matches are then likely, a principled way of determining a distribution must be found. With one or more wild card symbols in a context, the first step is to find all possible matches in the N-gram model, after which, for the purposes of PPM, corresponding prediction counts are added together. Given one or more wild card symbols in a prediction, probabilities of predictions within the distribution which have the same symbols in non-wild card positions are combined.

For this version (with wild card symbols shown as exclamation marks), the inter-layer section of the representation shown in Figure 3.4 must change to:

inter-layer

$(\textbf{Interval})_{\mathbf{S}}$		$\langle\langle 5\rangle,$	$\langle\langle -1\rangle,$	$\langle\langle -2\rangle,$	$\langle\langle 0\rangle,$	$\langle\langle -2\rangle,$	$\langle\langle !\rangle,$
$\otimes(\textbf{Duration} \otimes \textbf{ScaleDegree})_{\mathbf{Bp}}$	$\perp$	$\langle 48,\ 0\rangle\rangle$	$\langle 48,\ 5\rangle\rangle$	$\langle 48,\ 0\rangle\rangle$	$\langle ?,\ ?\rangle\rangle$	$\langle ?,\ ?\rangle\rangle$	$\langle ?,\ ?\rangle\rangle$
$(\textbf{Pitch} \otimes \textbf{ScaleDegree})_{\mathbf{S}}$	$\langle\langle 62,\ 0\rangle,$		$\langle\langle 66,\ 4\rangle,$	$\langle\langle 64,\ 2\rangle,$	$\langle\langle 64,\ 2\rangle,$	$\langle\langle 62,\ 0\rangle,$	$\langle\langle !,\ !\rangle,$
$\otimes(\textbf{Interval} \otimes \textbf{ScaleDegree})_{\mathbf{Bp}}$	$\langle !,\ !\rangle\rangle$	$\perp$	$\langle -7,\ 5\rangle\rangle$	$\langle 7,\ 0\rangle\rangle$	$\langle ?,\ ?\rangle\rangle$	$\langle ?,\ ?\rangle\rangle$	$\langle ?,\ ?\rangle\rangle.$

The second of these viewpoints reveals something interesting. The first element is (in some sense) defined, but it cannot be used to predict the second undefined element, which in turn cannot be used to predict the third; but can the first element be used to predict the third? When using threaded viewpoints, distant defined symbols may be used as context (intervening undefined symbols are ignored); in this non-threaded case involving `Interval`, however, it would not make sense and will not be permitted (similarly for `Contour`, `DurRatio` and `IOI`). Irrespective of this, since the third element must be predicted without reference to the second, it is highly unlikely that the probability distribution will favour a good match between the soprano and bass (vertically) at prediction position n. Indeed, backing off to a 0^{th}-order model in any circumstances in this version will increase the likelihood of a vertical mismatch between the prediction and support layers. Another potential drawback is the likely increase in the size of the `Pitch` domain resulting from the use of this element configuration, leading to relatively sparse statistics. Overall, whether or not these drawbacks are outweighed by the advantages of the limited look-ahead capability remains to be seen. In either event, this version opens the door to further possibilities for improvement.

To summarise, this is a weighted (PPM) model in which the i^{th} viewpoint element of the prediction layer is linked with the $(i+1)^{\text{th}}$ element of the support layer(s). The concept of a wild card symbol has been introduced.

10.3.2 Version 5: Looking Further into the Future

Version 4 introduced a limited look-ahead capability by including a future support layer symbol in the element to be predicted. This version extends this idea by allowing future context; this is mathematically sound since it has been proven that if a random process is Markovian in one direction, it is also Markovian in the opposite direction (Manning and Schütze, 1999, p. 355). It is considered to be acceptable in other ways, too; for example, a composer can look ahead when harmonising a melody, and a listener's understanding of what is currently being heard can be modified by what is heard a little later (Narmour, 1992). There are also practical considerations with respect to the use of subtask models. If, for example, cadences are dealt with first on any layer, it must be possible for the models to look ahead in order to knit the rest of the harmony in a phrase with its cadence; by using only past and current context, it is very likely that a mismatch would occur.

The following example assumes that a subtask model has already generated harmonic function symbols at the cadence, and that the current subtask model is about to generate the symbol immediately preceding the cadence. There are three of ways of introducing future context, all of which can be tried and evaluated. The strictest application of multiple viewpoints, version 5a, extends version 3 to allow, for example, $(\texttt{ScaleDegree})_S \otimes (\texttt{Harmony})_{Cp}$ N-grams such as:

$$\begin{array}{cccccc}
0 & 5 & 4 & 2 & 2 & 0 \\
\mathrm{I} & \mathrm{IV} & \mathrm{I} & \textbf{?} & \mathrm{V} & \mathrm{I} \\
n-3 & n-2 & n-1 & n & n+1 & n+2.
\end{array}$$

If we follow the precedent of version 4, we obtain for version 5b N-grams such as:

$$\begin{array}{ccccccc}
 & 5 & 4 & 2 & 2 & 0 & ! \\
\mathrm{I} & \mathrm{IV} & \mathrm{I} & \textbf{?} & \mathrm{V} & \mathrm{I} & \\
n-3 & n-2 & n-1 & n & n+1 & n+2 & n+3.
\end{array}$$

We could also link the i^{th} viewpoint element of the prediction layer with the $(i-1)^{\mathrm{th}}$ element(s) of the support layer, giving rise to version 5c. In this case, wild card symbols (see §10.3.1) are needed at the beginning of the support layer(s) and the end of the prediction layer (a formal representation of the context is also shown):

$$\begin{array}{ccccccc}
0 & 5 & 4 & 2 & 2 & 0 & \\
 & \mathrm{IV} & \mathrm{I} & \textbf{?} & \mathrm{V} & \mathrm{I} & ! \\
n-3 & n-2 & n-1 & n & n+1 & n+2 & n+3
\end{array}$$

$$\begin{aligned}
\{\langle \text{soprano}, -3, \texttt{ScaleDegree}, \langle 0\rangle\rangle, \langle \text{soprano}, -2, \texttt{ScaleDegree}, \langle 5\rangle\rangle,\\
\langle \text{soprano}, 0, \texttt{ScaleDegree}, \langle 2\rangle\rangle, \langle \text{soprano}, 1, \texttt{ScaleDegree}, \langle 2\rangle\rangle,\\
\langle \text{soprano}, 2, \texttt{ScaleDegree}, \langle 0\rangle\rangle, \langle \text{common}, -2, \texttt{Harmony}, \langle \mathrm{IV}\rangle\rangle,
\end{aligned}$$

$$\langle \text{common}, -1, \texttt{Harmony}, \langle \text{I} \rangle \rangle, \langle \text{common}, 1, \texttt{Harmony}, \langle \text{V} \rangle \rangle,$$
$$\langle \text{common}, 2, \texttt{Harmony}, \langle \text{I} \rangle \rangle, \langle \text{common}, 3, \texttt{Harmony}, \langle ! \rangle \rangle \}.$$

It is anticipated that version 5a will be the best of the three, because of the drawbacks inherited by versions 5b and 5c from version 4 (described in §10.3.1 above).

To keep things relatively simple, it was originally intended that in all of the above options the order of the model should be increased by alternately adding past and future context. If, however, the final two `Harmony` symbols in the above examples had not already been generated, it would then be necessary to back off to first- or zeroth-order models to achieve a context match, which is unsatisfactory. There are at least two ways of offering the chance of matches at higher orders. The first is to increase model order by initially adding past context up to some maximum order, followed by the addition of only future context until its size equals or is one element smaller than that of the past context. Back-off proceeds by removing only future context in the first instance, which affords the opportunity of matching large past contexts. The drawback with this idea, however, is that in circumstances where the future part of the context matches, but the past portion does not, back-off will result in the removal of all of the matching future context. The second way is to use the original back-off scheme (alternate removal of past and future pieces of context), but to place wild card symbols in contexts at positions where future symbols have not been predicted or generated. Both of these methods will be evaluated, although it is expected that the latter will be better.

To summarise, this is a weighted (PPM) model which allows future context; there are three alternative ways in which the prediction layer can be linked with the support layer(s).

10.3.3 Version 6: Utilising Cross-entropy to Construct the Back-off Sequence

In this version, we consider the possibility that past and future contexts of approximately the same size are not necessarily going to produce the best results; it is therefore now permissible to add either past or future context each time the model order is increased. The choice of one of the two possible context configurations at each order depends on their relative contribution to the lowering of the ten-fold cross-validation cross-entropy; in this way, the back-off sequence is optimised. Versions 6a, 6b and 6c are developments of versions 5a, 5b and 5c (although it is expected that in practice only the best performing version will be developed). For version 6a, a typical $(\texttt{ScaleDegree})_S \otimes (\texttt{Harmony})_{Cp}$ N-gram might then look like this, assuming that past context is more important:

$$\begin{array}{ccccc} 0 & 5 & 4 & 2 & 2 \\ \text{I} & \text{IV} & \text{I} & ? & \text{V} \\ n-3 & n-2 & n-1 & n & n+1. \end{array}$$

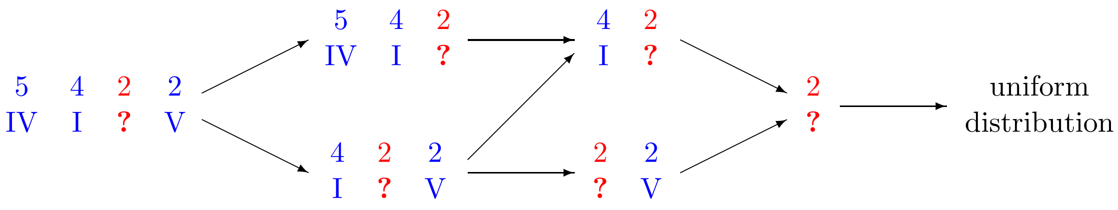


Figure 10.3: Graph illustrating the three possible version 6 back-off sequences beginning with the N-gram on the extreme left.

If any future context symbols are unknown, there are two realistic options: the first is to back off until all symbols in the context are known; and the second is to substitute wild card symbols for unknown symbols, thereby increasing the likelihood of a higher-order match. The off-line cross-entropy minimisation procedure will ensure the best outcome in each case; but it is expected that the wild card option will prove better.

The graph in Figure 10.3 illustrates the three possible back-off sequences starting with a sub-context of the one in the example N-gram above (one of these routes is chosen by the cross-entropy minimisation procedure). Notice that, as in all previous versions, as a back-off sequence is traversed, contexts are strict sub-contexts of preceding contexts. As before, if a context cannot be matched with one in the model, back-off proceeds until a match occurs. Construction of the probability distribution begins with the matched context and continues by further backing off until it is completed.

Optimising the back-off sequence of a single viewpoint model is all very well; but how does it fit in with viewpoint selection? Ideally, the back-off sequence would be constructed afresh every time a viewpoint is tried, since the optimum sequence may vary with the set of viewpoints to be combined (with their back-off sequences already optimised in a particular way). Noting, however, that by version 3 viewpoint selection was already taking a very long time, the ideal procedure is impractical; especially considering that, for example, it takes four ten-fold cross-validations to optimise the back-off sequence of a viewpoint model of maximum 2$^{\text{nd}}$-order. Compromising such that the back-off sequence is optimised only the first time a viewpoint is tried will shorten the running time considerably, but almost certainly not by enough. It is not realistically possible to combine back-off sequence optimisation with viewpoint selection, as things stand, other than for a very small pool of primitive viewpoints. The obvious answer is to return to a viewpoint selection procedure much closer to that used by Pearce (2005), which tries a modestly sized predetermined set of primitive and linked viewpoints at each addition stage. By the time this version is implemented we should have a good idea which viewpoints perform particularly well. An alternative solution is simply to optimise the back-off sequences of the viewpoint models within the best version 5 overall model. The basic idea is to start with an empty multiple viewpoint system, then add viewpoints from the corresponding version 5 system one at a time in order of decreasing effectiveness (subject to the need to predict at all positions in a musical sequence); the back-off sequence of a viewpoint model is optimised as it is added. Of course, if the addition of a viewpoint would increase the ten-fold cross-validation cross-entropy (*i.e.*, if it has been made redundant by optimising the previously added viewpoints) then it should be discarded.

To summarise, this is a weighted model which develops PPM by partially ordering individual N-gram models (one per model order) according to the cross-entropy of ten-fold cross-validation, with the intention of creating a close to optimal route through the search space for the purpose of context matching and the determination of complete probability distributions.

10.3.4 Version 7: Using Differently Shaped Contexts of the Same Order Within the Back-off Sequence

In this version, we develop Prediction by Partial Match by introducing the idea that it is not necessary to move to the next-lower order of model every time back-off is required. Versions 7a, 7b and 7c are developments of versions 6a, 6b and 6c, in which different N-gram models of the same order may appear in a back-off sequence. There are many possible ways of constructing such sequences. The simplest method is a development of the one outlined in version 6, where instead of adding the better of two alternative i^{th}-order models, the option of adding both of the models to the back-off sequence is also considered (models of the same order occupy a contiguous region of the back-off sequence, and higher-order models appear earlier in the sequence than lower-order models, as is customary in PPM). In the latter case, we must consider the sequential order of the two models to be added. Although placing the better performing i^{th}-order model before the other in the sequence seems like a reasonable thing to do, it is conceivable that switching the models could reduce the cross-entropy. Let us assume that one of the i^{th}-order models under consideration (model 1) on its own manages to predict one hundred symbols during cross-validation, each with a probability of 0.7; and that the other (model 2) on its own predicts ten of these one hundred symbols (and no others), each with a probability of 0.8. Adding model 1 to the back-off sequence would very likely lower the cross-entropy more than the addition of model 2; but adding both models, with model 2 at the head of the sequence, would produce the lowest possible cross-entropy. Using exclusion, the latter case predicts ten symbols with a probability of 0.8 and ninety symbols with a probability of 0.7, whereas placing model 1 at the head predicts one hundred symbols with a probability of 0.7. Both model orderings are therefore evaluated. Unfortunately, this means that we cannot now automatically assume that the i^{th}-order model at the head of the sequence forms the basis of the $(i+1)^{\text{th}}$-order models; it is reasonable to assume that if there are two i^{th}-order models, the the one which performs better on its own should be used as the basis of higher-order models. A high-level algorithm for this back-off sequence construction method is

Algorithm 10.1 A high level algorithm for the simplest method of construction of a version 7 viewpoint model back-off sequence.

```
create $-1^{\text{th}}$-order model $m_{-1}$ from viewpoint domain    ▷ Uniform distribution.
create $0^{\text{th}}$-order model $m_0$ from corpus
back-off sequence $bs \leftarrow m_0 : m_{-1}$
$k \leftarrow$ maximum order of model
for $i \leftarrow 1$ to $k$
    $a \leftarrow 2$    ▷ No. allowable $i^{\text{th}}$-order models extending best $(i-1)^{\text{th}}$-order model.
    for $i^{\text{th}}$-order model $m_i \leftarrow m_{i_1}$ to $m_{i_a}$
        create model $m_i$ from corpus
        temporary back-off sequence $ts \leftarrow m_i : bs$
        calculate ten-fold cross-validation cross-entropy using $ts$
    $m_i \leftarrow$ best performing $i^{\text{th}}$-order model
    $mx_i \leftarrow$ cross-entropy associated with best performing $i^{\text{th}}$-order model
    $b \leftarrow 2$    ▷ No. possible sequential arrangements of $i^{\text{th}}$-order models.
    for $i^{\text{th}}$-order model subsequence $bs_i \leftarrow bs_{i_1}$ to $bs_{i_b}$
        $ts \leftarrow bs_i : bs$
        calculate ten-fold cross-validation cross-entropy using $ts$
    $bs_i \leftarrow$ best performing $i^{\text{th}}$-order model subsequence
    $bsx_i \leftarrow$ cross-entropy associated with best performing $i^{\text{th}}$-order model subsequence
    if $bsx_i < mx_i$
      then $bs \leftarrow bs_i : bs$
      else $bs \leftarrow m_i : bs$
```

presented in pseudo-code in the style of Corman et al. (2001) in Algorithm 10.1.

How valid is the use of this type of back-off sequence from a theoretical point of view, though? To try to answer this question, let us consider the back-off sequence shown in Figure 10.4, which makes use of N-grams seen in the graph illustrating version 6 back-off sequences (Figure 10.3) for purposes of comparison. At the head of the sequence is a 2nd-order N-gram employing both past and future context. This backs off to another 2nd-order N-gram, the context of which is not a sub-context of the preceding one. This is new to PPM, but can be justified in the following way. The union of these two 2nd-order N-grams is the 3rd-order N-gram at the extreme left of Figure 10.3, which demonstrates the legitimacy of backing off to either of the 2nd-order N-grams in question. The version 7 sequence starts off as equivalent to the version 6 sequence in which the 3rd-order context is not matched; that is, the version 7 sequence can be thought of as having a phantom super-context which is not matched (in this case because it is not even tried). Since both 2nd-order contexts are sub-contexts of the super-context, it is also legitimate to back off between the 2nd-order models. Similarly, since all contexts in the sequence are sub-contexts of the super-context, any i^{th}-order N-gram can back off to any $(i-1)^{\text{th}}$-order one. There is one small fly in the ointment, however. In versions 0 to 6, once a context has been matched all subsequent contexts are also guaranteed to match, which is

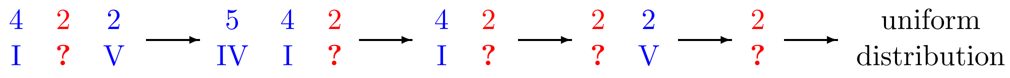


Figure 10.4: A possible version 7 maximum 2nd-order back-off sequence.

not necessarily the case in version 7. When such a non-match occurs, the only possible course of action is to back off again, taking all of the escape probability to the next model (non-matches are effectively disregarded).

Of course, there are other more complex ways of constructing back-off sequences which might perform better. It is possible that some lower-order models should appear earlier in a back-off sequence than some higher-order models; this is a radical departure from PPM as currently practised. In this case, in addition to trying each i^{th}-order model at the head of the sequence, they will be tried in other positions in the sequence with a view to minimising cross-entropy. It is unlikely that the optimum position will be far from the sequence head; therefore to avoid unreasonable time complexity it is proposed that a model is moved one position at a time from the head, and that evaluation ends as soon as the cross-entropy is higher than in the last trial. When trying both i^{th}-order models at the same time, it is proposed that one of the models remains in its optimum position while the other is moved one position at a time from the sequence head, as before.

In a further development, the restriction requiring that $(i+1)^{\text{th}}$-order contexts are based on the better of the i^{th}-order contexts can be removed. All possible context variations could be tried (*e.g.*, there are six different 5th-order context arrangements), but this is likely to result in excessively long processing times. A principled means of improving performance whilst at the same time restricting the number of models tried may well be beneficial. One such means is to undertake a branching search of the space of possible back-off sequences. Two different back-off sequences are constructed at the 2nd-order stage; one with new contexts based on one 1st-order context, and the other with them based on the other. This branching occurs at every increase in model order. Criteria will need to be developed to prune unpromising branches so as to keep running time within reasonable limits. At a predetermined maximum model order, the back-off sequences at the leaves of the tree can be compared according to their ten-fold cross-validation cross-entropies.

So far, a back-off sequence has been considered to be static; that is, the order of a sequence is determined off-line, and then remains fixed. If, however, a way could be found to optimise a back-off sequence for each individual prediction, an increase in performance is bound to result. Changing a static back-off sequence into a dynamic one can be achieved by creating complete probability distributions for each individual context (by immediately backing off to the uniform distribution), and then ordering the contexts according to what Conklin (1990) calls the relative entropy of their distributions (see §2.2.4) at every prediction step; lower relative entropy means a more certain model, which, provided it is a good model, should translate into better prediction performance. Since the order in which back-off occurs is not known in advance, the first step is to determine which of the models have matching contexts, and then the models are ordered with respect to the relative entropy of the distributions of the matched contexts. At this point the individual context distributions are discarded, with back-off smoothing proceeding as usual from the context at the head of the back-off sequence (alternatively, interpolated smoothing can be used). If instead the weighted geometric mean method (see §10.2.3.1 above) is used for combining the individual context distributions in this dynamic paradigm, there is no need to order the matched contexts; we would then talk about constructing and using PPM sets rather than back-off sequences. Note that it would be far easier to construct a PPM set (since the order is inconsequential); therefore the efficacy of the weighted geometric mean combination method should be evaluated at an early stage in any future work.

Finally, PPM sets (and dynamic back-off sequences) can be constructed by means of an adaptation of the viewpoint selection algorithm used by Pearce (2005) for the construction of multiple viewpoint systems. Any of the construction methods discussed earlier could result in N-gram models being added which might turn out in the longer run to be detrimental to the performance of the viewpoint model; a procedure based on the viewpoint selection algorithm (see §3.4.5) would be able to weed out such models. The first step is to add the -1^{th}-order model (uniform distribution) and the 0^{th}-order model to the initially empty PPM set. Deletion of the -1^{th}-order model from the set is very likely to reduce the ten-fold cross-validation cross-entropy (note that this model cannot be deleted from a dynamic back-off sequence, since it is required to ensure completed distributions). Following this, the addition of both possible 1^{st}-order models is tried, with the one contributing to the lowest ten-fold cross-validation cross-entropy being admitted to the set. At this point, the effect of deleting the 0^{th}-order model is evaluated; if a lower cross-entropy results, it is removed from the set. Next, the effect of adding the remaining 1^{st}-order model is evaluated. If it is added to the set another round of deletion ensues, after which both of the 2^{nd}-order models (extending the context of the initially chosen 1^{st}-order model) are evaluated. Again, the one which lowers the cross-entropy more is added to the set. There are further such rounds of addition and deletion until both models of the desired maximum order have been tried, and construction is complete. A variation in which all possible context configurations are evaluated can also be tried; and a branching search version of this algorithm is also a reasonable (if more time consuming) option, since the “greedy algorithm” approach of minimising cross-entropy at each step is not guaranteed to find the lowest cross-entropy solution overall.

The investigation and evaluation of all of these options will be a central feature of future research. It should be noted that because even the simplest of these options will significantly increase the time taken to construct a back-off sequence or PPM set, the remarks made in §10.3.3 about combining viewpoint selection with back-off sequence optimisation are even more pertinent here. It is conceivable that this work could have an impact on PPM and its applications both within the domain of music and beyond.

To summarise, this is a weighted model which develops PPM by allowing more than one N-gram model of the same order (*i.e.*, with the same context size, but different shape) within the partial ordering of individual N-gram models according to their tenfold cross-validation cross-entropies, with the intention of creating a close to optimal route through the search space for the purpose of context matching and the determination of complete probability distributions. There are several different ways of ordering the models, including a means of dynamically optimising the ordering for each prediction. One proposed combination method obviates the need for ordering.

10.3.5 Version 8: Extending the Use of Differently Shaped Contexts

So far, the use of differently shaped contexts of the same order has been applied to a notion of a linked viewpoint which is not too far removed from that used by Conklin and Witten (1995) and Pearce (2005). Intra-layer linking is exactly the same, and inter-layer linking (which in any case did not exist in relation to melody alone) employs linking between the prediction layer and support layers which is either conventional or slightly offset. This extension loosens the definition of a linked viewpoint to allow more independence between the layers; that is, inter-layer linking becomes more flexible, but intra-layer linking remains as strict as ever.

We begin by examining the implications of wild card symbols, which have necessarily been introduced into previous versions. Let us consider the example N-gram for version 5a (see §10.3.2 above), which has a total of ten symbols in its context. If the two future `Harmony` symbols have not yet been generated, the most satisfactory solution is to replace them with wild card symbols, which is equivalent to removing those symbols from the context altogether. This results in the following rather oddly shaped context/prediction configuration[1] which has an eight symbol context:

$$\begin{array}{cccccc} 0 & 5 & 4 & 2 & 2 & 0 \\ \mathrm{I} & \mathrm{IV} & \mathrm{I} & ? & & \\ n-3 & n-2 & n-1 & n & n+1 & n+2. \end{array}$$

If we are effectively allowing certain symbols to be removed from the context because they do not yet exist, then why not also allow symbols which do exist to be removed from the context? More to the point, why should such symbols necessarily be put into the context in the first place? At the same time, the “must have” symbols can be removed from the prediction, thereby becoming available for use in the context. It is thought

[1] It is no longer appropriate to call this an N-gram, since it is not clear which integer should replace the N.

that this additional flexibility, applied at this stage only to inter-layer linking, will afford performance benefits. It should be noted that we have arrived, *de facto*, at more general dynamic Bayesian network models[2] rather than N-gram models; see Appendix G.

Other researchers have used strangely shaped contexts, for example Hild et al. (1992), who created a neural network system (HARMONET) for four-part harmonisation. Individual harmonies were learnt with respect to a fixed local context (or window), consisting of previous harmonic symbols; past, current and future melodic (or soprano) symbols; and symbols indicating position in phrase and whether stressed or not stressed. An adaptation of their figure describing this window is shown below:

$$\begin{array}{ccccc} & & s & s & s \\ H & H & H & ? & \\ & & & phr & \\ & & & str & \\ n-3 & n-2 & n-1 & n & n+1. \end{array}$$

Presumably this particular configuration was used because it was found to be a good predictor of harmony. In order to be able to make use of oddly shaped contexts such as this, which may well turn out to be useful predictors, this version of the framework allows contexts to be built up by adding context from any one of the linked layers at each stage, rather than having to add context from all of those layers at the same time. It is worth reiterating that for each of the individual layers under consideration there is one completely conventional intra-layer linked viewpoint, which can be different for each layer. Contexts can therefore be built up (using cross-entropy as a guide, as before) in a way similar to that of building features in a Markov random field (Della Pietra et al., 1997). Many more differently shaped contexts of the same order[3] are now possible. There must, though, be restrictions (however arbitrary) on what constitutes an allowable context, otherwise the construction of back-off sequences would become intractable. Here is a description of how valid contexts are built up, assuming that a layer is predicted by only one subtask model:

1. Contexts are drawn from prediction layers and those support layers which appear in the viewpoint name, as usual. Past, current and future context is permissible from the support layers, but only past context from the prediction layers.

2. Let us assume that a viewpoint element at chord position n is to be predicted. In support layers, there are three options for the first symbol to be used in a context: that at position n or the nearest defined symbol in the past or future. In prediction layers, the only option is the nearest defined symbol in the past.

[2] Although DBNs are, strictly speaking, 1st-order models, there is no fundamental reason for this restriction.

[3] Model order must now be defined differently, however; this is done later in the section.

3. In general, for any layer, additional context symbols are drawn from defined symbols adjacent to those already in the context. The symbol at position n in a support layer is an exception: it can be used in spite of the fact that it may not always be defined, and it may be missed out whether or not it is always defined.

The question of what to do if context at position n is undefined is an interesting one. There are two possibilities: back off until the entire context is fully defined; or allow undefined symbols at this point in the context, on the basis that they may provide valuable information. For example, if `ScaleDegree` $\ominus$ `Tactus` appears in a support layer at position n and is undefined, the implication is that this is not a tactus beat. The latter option appears to be better, since it involves less back-off which should result in "sharper" probability distributions. It is therefore proposed that, to this limited extent, undefined symbols will be permitted in version 8 contexts.

The situation with respect to context construction is slightly different if, for example, a subtask model is employed to predict cadences first, and then the rest of the layer is predicted by one or two other subtask models. One possibility is for one subtask model to predict all of the rest of the harmony except for the chord immediately preceding the cadences, while another specialises in the prediction of the pre-cadence chords. The former subtask model would follow the numbered procedure above for context construction, while it would make more sense for the latter to allow future context in the prediction layers. If there is no pre-cadence model, it would be useful for the model predicting the bulk of the harmony to employ future prediction layer context. In this case, since there is no such future context until a cadence is reached, the preferred solution is once again to allow wild card symbols in future context positions. Backing off to a context which completely comprises known symbols is a possible, but very likely less effective, alternative.

Previously, the order of a model increased by one on the addition of a compound inter-layer linked viewpoint element comprising the same number of symbols as the number of primitive viewpoints it contained. A consequence of this version is that the order of a model must increase by one on the addition of a compound intra-layer linked viewpoint element, which clearly comprises fewer symbols (possibly only one). This is not a problem when it comes to comparing performance with other versions, however, since we are primarily concerned with comparing the best models, not with comparing models of the same (or equivalent) order. Here is a simple example showing all possible 2nd-order contexts (including future prediction layer context) comprising $(\texttt{ScaleDegree})_S$ and $(\texttt{Harmony})_{Cp}$, assuming that they are extensions of this 1st-order context:

$$\begin{array}{cc} 4 & \\ & ? \\ n-1 & n. \end{array}$$

The five 2$^{\text{nd}}$-order contexts are:

$$\begin{array}{ccc} 5 & 4 & \\ & & ? \\ n-2 & n-1 & n \end{array} \quad \begin{array}{cc} 4 & \\ \mathrm{I} & ? \\ n-1 & n \end{array} \quad \begin{array}{cc} 4 & 2 \\ & ? \\ n-1 & n \end{array} \quad \begin{array}{ccc} 4 & & 2 \\ & ? & \\ n-1 & n & n+1 \end{array} \quad \begin{array}{ccc} 4 & & \\ & ? & \mathrm{V} \\ n-1 & n & n+1. \end{array}$$

The dynamic Bayesian network models using such contexts are constructed automatically in a similar way to N-gram models in earlier versions. They are then placed (or not, depending upon their effect on cross-entropy) within a back-off sequence, dynamic back-off sequence or PPM set by means of the construction method found to perform most effectively (within reasonable time complexity bounds) in version 7 trials (see §10.3.4).

Restrictions on which contexts are allowable will be different depending on the type of overall model that one is aiming to build; models for harmonic analysis, the generation of harmony and the cognition of harmonic movement are all likely to have different restrictions. For example, one's understanding of a recently heard harmonic progression might change as a result of hearing a subsequent chord (*i.e.*, an alternative parsing becomes more likely). It might therefore be legitimate to allow future context to condition a current note or chord; but there must surely be a limit on how much future context can be used, based on human short term musical memory limitations. Issues such as these will be considered further during future research.

This is the most radical version so far, and one of its consequences is that the idea of a link between elements at particular chord positions in different layers has been shattered. This means that the inter-layer section of the solution array representation shown in Figure 3.4 can no longer be completed in any meaningful way. The formal context representation introduced in §3.2.4 is able to cope with this, however. Consider the following simple $(\texttt{ScaleDegree})_S \otimes (\texttt{Harmony})_{Cp}$ context based on the HARMONET window above:

$$\begin{array}{ccccc} & & 4 & 2 & 2 \\ \mathrm{I} & \mathrm{IV} & \mathrm{I} & ? & \\ n-3 & n-2 & n-1 & n & n+1. \end{array}$$

This would be represented as follows:

$$\begin{aligned} \{&\langle \text{soprano}, -1, \texttt{ScaleDegree}, \langle 4\rangle\rangle, \langle \text{soprano}, 0, \texttt{ScaleDegree}, \langle 2\rangle\rangle, \\ &\langle \text{soprano}, 1, \texttt{ScaleDegree}, \langle 2\rangle\rangle, \langle \text{common}, -3, \texttt{Harmony}, \langle \mathrm{I}\rangle\rangle, \\ &\langle \text{common}, -2, \texttt{Harmony}, \langle \mathrm{IV}\rangle\rangle, \langle \text{common}, -1, \texttt{Harmony}, \langle \mathrm{I}\rangle\rangle\}. \end{aligned}$$

Conventional inter-layer linking can be seen as a special case within this framework of description, where for every prediction layer element in the set there must also be, from each support layer, an element at the same relative chord position. In other words, the notion of allowing contexts to be built up by adding viewpoint elements from individual layers is, within this framework of description, a generalisation of the more conventional idea of an inter-layer linked viewpoint.

There are at least three reasons for believing that this flexible approach to inter-layer linking is likely to be better than a more conventional approach. Firstly, it allows context to be used which would otherwise be unavailable if conventionally linked to an undefined symbol. In the following example (`Interval` in the soprano and `Harmony` in the common layer), the first chord position is completely unavailable for use in a conventionally linked viewpoint context, because `Interval` is undefined:

$$\begin{array}{cccc} \perp & 5 & -1 & -2 \\ \mathrm{I} & \mathrm{IV} & \mathrm{I} & ? \\ n-3 & n-2 & n-1 & n. \end{array}$$

On the other hand, the current version allows the first `Harmony` symbol (as well as the "must have" symbol) to be added to the context:

$$\begin{array}{cccc} \perp & 5 & -1 & -2 \\ \mathrm{I} & \mathrm{IV} & \mathrm{I} & ? \\ n-3 & n-2 & n-1 & n. \end{array}$$

When predicting or generating towards the beginning of a piece, there is limited past context available; this flexible approach can provide valuable additional context, as demonstrated above. The benefits are potentially even greater with respect to the linking of threaded viewpoints with other viewpoints, bearing in mind that threaded viewpoints can be undefined in many places. Consider the following, which occurs towards the end of a longer sequence (with `Interval` $\ominus$ `FirstInBar` in the soprano and `Harmony` in the common layer). Conventionally, only one chord position within this extract can be used as context if a symbol at the final chord position is being generated:[4]

$$\begin{array}{cccccc} \perp & -4 & \perp & \perp & \perp & -5 \\ \mathrm{I} & \mathrm{IV} & \mathrm{I} & \mathrm{V} & \mathrm{V} & ? \\ n-5 & n-4 & n-3 & n-2 & n-1 & n. \end{array}$$

This simplifies to:

$$\begin{array}{cc} -4 & -5 \\ \mathrm{IV} & ? \\ n-1 & n. \end{array}$$

On the other hand, the current version opens up the intriguing possibility of building up short term context whilst also taking into account much longer term context:

$$\begin{array}{cccccc} \perp & -4 & \perp & \perp & \perp & -5 \\ \mathrm{I} & \mathrm{IV} & \mathrm{I} & \mathrm{V} & \mathrm{V} & ? \\ n-5 & n-4 & n-3 & n-2 & n-1 & n. \end{array}$$

[4] This conventionally linked viewpoint cannot be used to predict, for example, at position $n-1$, as it is undefined.

Simplification, as above, is no longer possible; but it is possible to use this flexibly linked viewpoint to generate a symbol at, for example, the penultimate position of the sequence, which could not previously be done (recalling that undefined context is permitted at the prediction point):

$$\begin{array}{cccccc} \perp & -4 & \perp & \perp & \perp & -5 \\ \mathrm{I} & \mathrm{IV} & \mathrm{I} & \mathrm{V} & ? & ? \\ n-4 & n-3 & n-2 & n-1 & n & n+1. \end{array}$$

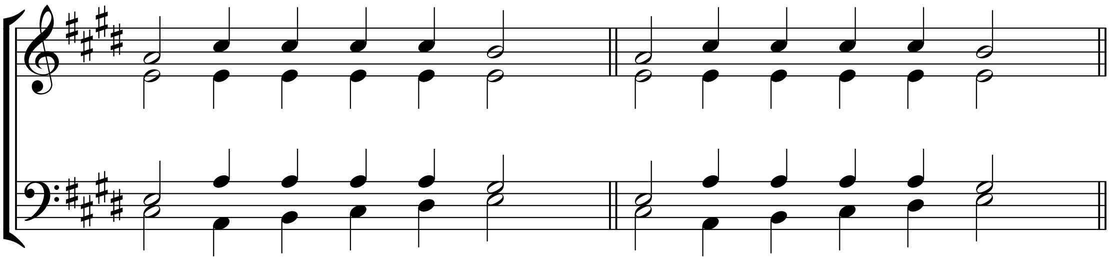
Secondly, flexible inter-layer linking is able to avoid information redundancy; for example, a `ScaleDegree` symbol is likely to share mutual information with a harmonic symbol at the same chord position, since a harmonic symbol represents a set of simultaneously sounding scale degrees. Symbols are added to a context on the basis of acquisition of the greatest amount of information, rather than having to add a symbol because it is strictly inter-layer linked to another.

Thirdly (as a result of the second reason), contexts produced by this flexible method are likely to be relatively sparse, thereby covering a longer length of sequence than conventionally formed contexts containing the same number of symbols. This helps to capture long-term dependencies within the music. For contexts of the same overall length, the fewer symbols they contain the more often they will be found in the corpus, and therefore in the model (*i.e.*, data sparsity is less of an issue in the current version). Here is a conventional context and an extreme example of a context allowable under this version, containing the same number of symbols:

$$\begin{array}{cccc} 5 & 4 & 2 & 2 \\ \mathrm{IV} & \mathrm{I} & ? & \mathrm{V} \\ n-2 & n-1 & n & n+1 \end{array} \qquad \begin{array}{cccccc} & & & 2 & 2 & 0 \\ \mathrm{I} & \mathrm{IV} & \mathrm{I} & ? & & \\ n-3 & n-2 & n-1 & n & n+1 & n+2. \end{array}$$

In previous work (*e.g.*, Brooks Jr. et al., 1993), long contexts have been found to reproduce long sequences from the corpus, which is undesirable. The use of smoothed PPM models makes this less likely, however. In addition, for models which harmonise given melodies, long contexts with melodic symbols in them can only be repeatedly matched in this way if the melody happens to be in the corpus.

The symbol × has been chosen to represent the more flexible inter-layer linking in order to clearly distinguish it from the strict linking used within layers. Figure 10.5 shows a constituent dynamic Bayesian network model of viewpoint model $(\texttt{ScaleDegree})_S \times (\texttt{ScaleDegree} \ominus \texttt{Tactus})_{ATp} \times (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_B$ predicting the second and fourth alto and tenor notes of the second bar of hymn tune *Merton* (Vaughan Williams 1933, hymn no. 5). It demonstrates that not only do contexts change shape as a back-off sequence is traversed, but in this version (to a much greater extent than in previous versions) they may also change shape with prediction position. For the first prediction the past context is vertically aligned as close as possible to the prediction point, since the relevant viewpoints are both defined at this point. For the second, however,



S	5		9					9		9	
A	0	?					0		?		
T	0	?					5		?		
B						0					0
	$n-1$	n	$n+1$	$n+2$	$n+3$	$n+4$	$n-2$	$n-1$	n	$n+1$	$n+2$

Figure 10.5: The second bar of hymn tune *Merton* (Vaughan Williams 1933, hymn no. 5) appears twice in fully expanded form. A constituent dynamic Bayesian network model of viewpoint model $(\texttt{ScaleDegree})_S \times (\texttt{ScaleDegree} \ominus \texttt{Tactus})_{ATp} \times (\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_B$ is shown predicting the second and fourth alto and tenor notes of the bar.

while the $(\texttt{ScaleDegree})_S$ context remains close to the prediction the $(\texttt{ScaleDegree} \ominus \texttt{Tactus})_{ATp}$ context has to move away slightly to its nearest defined position. The $(\texttt{ScaleDegree} \ominus \texttt{LastInPhrase})_B$ context is fixed in absolute terms (until the prediction point reaches the end of the phrase), while in relative terms it moves closer to the prediction point as the musical sequence is traversed; so the viewpoint "knows" when it is approaching the end of a phrase.

To summarise, this is a weighted model which develops the concept of a linked viewpoint to allow a more flexible form of linking between layers. It also makes use of the earlier developments of PPM by partially ordering individual dynamic Bayesian network models according to the ten-fold cross-validation cross-entropy, with the intention of creating a close to optimal route through the search space for the purpose of context matching and the determination of complete probability distributions.

10.3.6 Version 9: Differently Shaped Contexts Within Individual Layers

An extremely radical version is now proposed, which loosens the definition of intra-layer linking in a similar way to that of inter-layer linking in version 8. In this version, contexts are built up by adding context from the individual primitive viewpoints that make up the intra-layer linked viewpoints. This potentially makes the contexts even more sparse and longer ranged, and also makes available even more previously unusable context. A consequence of this version is that the intra-layer linked viewpoint rows of the solution array representation shown in Figure 3.4 can no longer be completed in any meaningful way; even the set representation for contexts must be modified to accommodate this generalisation. Each element in the set is now represented as (layer, relative chord position, primitive viewpoint, symbol) rather than (layer, relative chord position, intra-layer linked viewpoint, symbol tuple) as before (see §3.2.4.3). In this version, the model order increases by one every time a primitive viewpoint element is added to the context. The wording in §10.3.5 of the numbered description of how valid contexts are built up can stand, but it must be realised that in version 8 the word "symbol" refers to a conventional intra-layer linked viewpoint element, whereas in this version it refers to a primitive viewpoint element.

An example of a complex linked viewpoint context which could be formed within the framework of this version is shown below:

$$\begin{array}{rcccc}
\texttt{Pitch}_S & 67 & 66 & & \\
\texttt{ScaleDegree}_S & & 4 & & 2 \\
\texttt{Interval}_B & & 7 & ? & \\
\texttt{ScaleDegree}_B & & 0 & ? & 7 \\
\texttt{Harmony}_C & & \mathrm{I} & \mathrm{V} & \\
 & n-2 & n-1 & n & n+1.
\end{array}$$

Now that intra-layer linking is as flexible as inter-layer linking, the symbol × represents both forms, resulting in the following name for this viewpoint:

$$(\texttt{Pitch} \times \texttt{ScaleDegree})_S \times (\texttt{Interval} \times \texttt{ScaleDegree})_{Bp} \times (\texttt{Harmony})_C.$$

The formal representation of the context is:

$$\{\langle \text{soprano}, -2, \texttt{Pitch}, 67\rangle, \langle \text{soprano}, -1, \texttt{Pitch}, 66\rangle, \langle \text{soprano}, -1, \texttt{ScaleDegree}, 4\rangle,$$
$$\langle \text{soprano}, 1, \texttt{ScaleDegree}, 2\rangle, \langle \text{bass}, -1, \texttt{Interval}, 7\rangle, \langle \text{bass}, -1, \texttt{ScaleDegree}, 0\rangle,$$
$$\langle \text{bass}, 1, \texttt{ScaleDegree}, 7\rangle, \langle \text{common}, -1, \texttt{Harmony}, \mathrm{I}\rangle, \langle \text{common}, 0, \texttt{Harmony}, \mathrm{V}\rangle\}.$$

To summarise, this is a weighted model which further develops the concept of a linked viewpoint to allow the more flexible form of linking within as well as between layers. It also makes use of the earlier developments of PPM by partially ordering individual dynamic Bayesian network models according to the ten-fold cross-validation cross-entropy, with the intention of creating a close to optimal route through the search space for the purpose of context matching and the determination of complete probability distributions. Please note that an explanation is given in Appendix G as to how versions 8 and 9 fit into the graphical model framework.

Table 3.2 (see §3.4.3) lists features introduced into the multiple viewpoint framework for harmony, and shows how they ideally relate to the different model versions. From a time complexity point of view, however (see Chapter 4), it may be necessary to implement a subset of the indicated features for the more complex versions.

10.4 Summary

In this chapter, ideas for improving versions 0 to 3 were discussed in detail, covering such topics as viewpoints, attribute prediction, model combination, viewpoint selection and representation. This was followed by an exposition of versions 4 to 9, which further extend the multiple viewpoint framework by introducing such innovations as future context and differently shaped contexts of the same order.

Chapter 11

Conclusions

11.1 Thesis Review

This research has been concerned with the construction and evaluation of statistical models of melody and four-part non-homophonic harmony, and in particular with the development of representational, modelling and search techniques employed in the construction of such models. Chapter 1 enumerated the problems associated with using explicit rules in the modelling of aspects of musical composition, and went on to explain how a machine learning approach had the potential to overcome these problems. It would be necessary to develop existing representational and modelling techniques in order to produce convincingly good models, however. The representational formalism chosen is called *multiple viewpoint systems* (Conklin, 1990). This framework not only represents basic musical attributes like note duration and pitch, but also derived attributes such as intervals. Pearce (2005) successfully used this framework to produce cognitive models of melodic expectancy, and his models were later developed to carry out phrase segmentation (Potter et al., 2007).

The primary motivation for this research was the proposal and evaluation of theories of musical styles in the form of computational models, which is an activity belonging to the interdisciplinary field of computational musicology. The idea was to develop representations, modelling techniques and machine learning techniques, applicable to melody and four-part harmony, to the extent that statistical models induced from different musical corpora were capable of producing melodies and harmonisations which were stylistically characteristic of each individual corpus. The computational approach employed allowed the use of scientific methodologies. A secondary motivation was the expectation that these developments could, in future, be applicable to the computational modelling of the cognition of melody and harmony.

So far, multiple viewpoint systems have mostly been used to model monodic sequences, and the pool of available viewpoints has been quite limited. Four-part harmony, however, is more complex, consisting as it does of four interrelated sequences. Consequently, the main aims of the research were as follows: firstly to investigate the effect of a large pool of viewpoints on melodic models; secondly, to propose ways in which the multiple viewpoint framework could be developed, such that the complexities of harmony could be adequately addressed; thirdly, to analyse the time complexity of software implementing these developments; fourthly, to design and carry out experiments to determine the best performing of these developments; and finally, to analyse the best performing multiple viewpoint systems from a music theoretic point of view, searching for regularities which confirm, conflict with and possibly transcend the commonly agreed rules of harmony, as well as seeking inspiration for new or improved viewpoints.

A review of previous research relating to the computational modelling of music was presented in Chapter 2. Firstly, various computational techniques that have been employed in such modelling were examined, including constraint satisfaction, genetic algorithms, finite context grammars, multiple viewpoint systems and graphical models. After this, there was a discussion about corpora and the representation of musical structure (including the multiple viewpoint representation). Next, the evaluation of computational models of music was discussed; and finally, a summary of previous research on the computational modelling of music was presented, including descriptions of work using constraint-based, evolutionary, connectionist, and multi-agent methods. Particular emphasis was placed on the statistical modelling of melody and harmony, with approaches using Markov models, dictionary-based models, HMMs, HHMMs, multiple viewpoint systems, Bayesian networks and Markov random fields being described.

In Chapter 3, the central ideas of the research were laid out in detail. First of all, the structure of a statistical model of harmony was analysed in detail. The hierarchical nature of the structure afforded a convenient means of discussing different aspects of the research at the appropriate level of abstraction. Discussion of viewpoint models included definitions of fifteen new primitive viewpoints; details of how Prediction by Partial Match (Cleary and Witten, 1984) would be used to combine different orders of N-gram model into a viewpoint model; and the introduction of a representational scheme for the harmonic modelling framework (including a distinction between intra- and inter-layer linking). A short section on melodic modelling (version 0) was followed by an exposition of the development of the multiple viewpoint framework for harmony. Version 1 was the simplest possible application of viewpoints to harmony. This was the base-level version, which was developed into increasingly complex versions 2 and 3 with the objective of improving prediction performance. In order to keep viewpoint selection times from becoming excessive, not all of the viewpoints were implemented in the latter case. Explained next was the need for harmony to be expanded in order to be modelled, and the related requirement for viewpoint `Cont` to be introduced. This was followed by a detailed discussion of viewpoint selection, in which candidate multiple viewpoint systems are evaluated by calculating the average cross-entropy of a ten-fold cross-validation of the corpus. Finally, there was a brief outline of evaluation and the methodology to be employed in this research.

Chapter 4 investigated the related topics of viewpoint domains and time complexity. In the context of the multiple viewpoint framework, a viewpoint domain is the set of valid elements (or symbols, or values) for a viewpoint. There was discussion of three issues affecting domains: how to avoid excessive run times by keeping the size of the $(\texttt{Pitch})_{SATB}$ domain relatively small; the need, in general, for domains to be specially constructed before each prediction; and the fact that linked viewpoint domains are not, in general, constructed simply by taking the Cartesian product of the constituent viewpoint domains. Next, detailed guidance was given on how to reliably construct domains for melody alone, and for versions 1, 2 and 3 of the multiple viewpoint framework for harmony. Finally, an empirical analysis of the time complexity of version 1 of our computer model was carried out, demonstrating the effect of number of viewpoints, corpus size and type of domain (seen, augmented or full).

An analysis of the prediction performance of version 0 (melody only) was presented in Chapter 5. Different types of model with various values of $\hbar$ were directly compared after the selection of multiple viewpoint systems using method VS3 (see §3.4.5.2) and the optimisation of biases. For the prediction of `Duration` and `Pitch` using a single multiple viewpoint system, it was found that for the LTM, weighted geometric viewpoint combination is much better than weighted arithmetic (*i.e.*, it produces a much lower cross-entropy); also, LTM+ is much better than LTM. For the STM, weighted arithmetic combination just has the edge on weighted geometric. The best method for combining LTM with STM is LS1 (combining viewpoints within the LTM and STM first). BOTH+ is much better than BOTH; indeed, it is the best performing type of model overall. This being the case, BOTH+ was chosen to be the subject of all following investigations. Corpus 'B' produces much lower cross-entropies than corpus 'A' for no easily identifiable reason; but by a small margin, corpus 'A+B' produces the lowest cross-entropy overall, as would be expected for a larger corpus. By combining separately selected systems for the prediction of `Duration` and `Pitch`, it is possible to create slightly better models than those with a single system for the prediction of `Duration` and `Pitch` together. The best system overall, using corpus 'A+B' with an $\hbar$ of 6 and 3 for `Duration` and `Pitch` respectively, has a cross-entropy of 2.86 bits/note. It is anticipated that selecting a separate STM given an already selected LTM+ could further enhance performance (*i.e.*, the LTM+ would be taken into account during the selection of the STM such that a complementary system emerges).

Chapter 6 presents a similar analysis for versions 1, 2 and 3. The version 1 comparisons in §6.2 demonstrated that, as expected, the use of the augmented `Pitch` domain results in far higher cross-entropies than those produced by the seen domain, especially for the prediction of `Pitch`. Since this is not a like for like comparison, the higher cross-entropies are not necessarily an indication of worse performance; indeed, the larger domain, being more representative of a larger (though hypothetical) corpus, produces more realistic probabilities. Performance is enhanced by using separately selected multiple viewpoint systems to predict individual basic attributes rather than predicting them all using a single system. This effect is more pronounced for the augmented Pitch domain than for the seen domain. Using the best systems selected using corpus ‘A’ in conjunction with corpus ‘A+B’ results in a large improvement in prediction performance.

The first set of version 2 viewpoint selection runs in §6.3, for attribute prediction together using the seen Pitch domain, compared different combinations of two-stage prediction. By far the best performance is obtained by predicting the bass part first followed by the inner parts together. Generally, other comparisons produced results similar to those obtained for version 1; exceptionally, however, the use of the larger corpus ‘A+B’ leads to a much smaller improvement in prediction performance.

From this point on, only separate attribute prediction using the augmented Pitch domain was investigated. In comparing version 1 with version 2 (see §6.4), only Cont and Pitch were taken into consideration, since the prediction of Duration was not directly comparable. On this basis, version 2 is better than version 1 when using corpus ‘A’; but when corpus ‘A+B’ is used, their performance is identical. Similarly, §6.5 showed that version 3 (predicting alto/tenor/bass given soprano) performs much better than version 1 when using corpus ‘A’; but the version 3 performance advantage is greatly reduced on changing to corpus ‘A+B’. We can infer from this that version 1 creates more general models, better able to scale up to larger corpora which may deviate somewhat from the characteristics of the original corpus. We found in §6.6 that the performance of version 3 (predicting bass given soprano followed by alto/tenor given soprano/bass) was better than that of version 2, although the margin was reduced on using corpus ‘A+B’. Overall models with a superior performance could be constructed by combining the better of the version 2 and 3 subtask models (version 3.2+).

On removing Duration from consideration in order to directly compare version 1, version 2 and all of the various version 3 models, we found that for corpus ‘A’ all of the version 3 sub-versions outperformed versions 1 and 2 (see §6.7). Version 3.2+ (version 3+ predicting bass followed by alto/tenor) performed best overall, having a cross-entropy 0.50 bits/chord lower than the worst-performing version 1. The use of corpus ‘A+B’ changed things considerably. Version 1 benefitted from the biggest improvement, while version 3.2+ suffered the largest deterioration in performance: although version 3.2+ was still better, the margin had been reduced to 0.17 bits/chord. A further modest increase in corpus size could see version 1 having the lowest overall cross-entropy, although version 3.2+ is likely to remain preeminent for viewpoint selection using larger corpora. On reinstating the prediction of Duration and making comparisons to the fullest possible extent, we found nothing which altered these conclusions.

A comparison in §6.8 of the best version 0 to 3 multiple viewpoint systems for the prediction of Duration, Cont and Pitch separately led to three main conclusions. Firstly, there appears to be little correlation between Duration and Cont, as evidenced by the scarcity of the primitives Duration and DurRatio in Cont-predicting systems and the dearth of the primitive `Cont` in `Duration`-predicting systems. Secondly, primitives derived from `Pitch` are heavily involved in `Duration`-predicting systems, whereas `Duration` and `DurRatio` are almost absent from `Pitch`-predicting systems. This suggests that a change in the basic attribute prediction order, such that `Pitch` is predicted before `Duration`, could be beneficial. Finally, since viewpoints $\texttt{IOI} \ominus \texttt{FirstInBar}$, $\texttt{Interval} \ominus \texttt{FirstInBar}$, $\texttt{ScaleDegree} \ominus \texttt{FirstInBar}$, $\texttt{Contour} \ominus \texttt{Tactus}$, $\texttt{Pitch} \ominus \texttt{Tactus}$, `InScale` and `Tessitura` appear in the best version 1 and 2 systems, it is likely that the version 3 models can be improved by adding these viewpoints to the restricted version 3 pool.

In Chapter 7 we examined version 0 (melodic) multiple viewpoint systems and speculated on why certain viewpoints had been selected from a music theoretic point of view. On its own `ScaleDegree` is a good viewpoint, as it effectively learns which degrees of the chromatic scale are present in the major scale (*i.e.*, those occurring most frequently in the corpus). Linking viewpoints such as `Tessitura` and `Interval` with `ScaleDegree` endows an ability (albeit imperfect) to differentiate between octaves, which improves performance. Viewpoint `InScale` is a good substitute for `ScaleDegree` in the STM.

There are strong metrical regularities relating to both note length and pitch, as evidenced by the performance of $\texttt{Duration} \otimes \texttt{Metre}$, $\texttt{Duration} \otimes \texttt{TactusPositionInBar}$, $\texttt{DurRatio} \otimes \texttt{TactusPositionInBar}$, $\texttt{ScaleDegree} \otimes \texttt{Metre}$ and $\texttt{Interval} \otimes \texttt{TactusPositionInBar}$. There are also regularities at phrase boundaries. Viewpoints which perform particularly well in this respect are $\texttt{DurRatio} \otimes \texttt{Phrase}$, $\texttt{ScaleDegree} \otimes \texttt{Phrase}$, $\texttt{ScaleDegree} \otimes \texttt{Piece}$, $\texttt{Interval} \otimes \texttt{Phrase}$, $\texttt{Interval} \otimes \texttt{LastInPhrase}$, $\texttt{Duration} \otimes (\texttt{ScaleDegree} \ominus \texttt{FirstInPhrase})$ and $\texttt{Interval} \otimes \texttt{FirstInPhrase}$. There is reason to believe that viewpoints $\texttt{DurRatio} \otimes \texttt{Interval}$ and $\texttt{Interval} \otimes \texttt{FirstInBar}$ are similarly able to model what is happening at phrase boundaries (although not exclusively), using `DurRatio` as a proxy for `Phrase` and `FirstInBar` as a proxy for `FirstInPhrase`.

In Chapter 8 we examined version 1, 2 and 3 multiple viewpoint systems in a similar way. Let us begin by reviewing version 1 viewpoints, where we find that metrical importance also correlates with `Cont`. Metrical structure, sometimes inferred from the corpus, is key to the success of $(\texttt{Duration} \otimes \texttt{PositionInBar})_{SATB}$, $(\texttt{Cont} \otimes \texttt{Metre})_{SATB}$ and $(\texttt{Cont} \otimes \texttt{TactusPositionInBar})_{SATB}$. If pairs of chords in different parts of the corpus have the same intervals in corresponding parts, there is a fair chance that the pairs of chords are the same relative to the tonic. $(\texttt{Cont} \otimes \texttt{Interval})_{SATB}$ can therefore usefully gather statistics on `Cont` for the purpose of its prediction, following which `Pitch` can also be predicted. There is room for improvement in this regard, since `Interval` is blind to key. This deficiency is rectified by viewpoint $(\texttt{Cont} \otimes \texttt{ScaleDegree})_{SATB}$, which in addition is sure to find pairs of chords that are the same except for octave. Finally, $(\texttt{ScaleDegree} \otimes \texttt{LastInPhrase})_{SATB}$ is preferred here to $\texttt{ScaleDegree} \otimes \texttt{Phrase}$, which performs well with respect to melody. This suggests that it may be easier to determine harmonic regularities at phrase endings (*i.e.*, at cadences).

We now move on to version 2 viewpoints. All phrases begin with chords in which all notes are newly sounded; therefore a Cont prediction of $\langle F,\ F\rangle$ from (Cont $\otimes$ (ScaleDegree $\ominus$ FirstInPhrase))$_{SB}$ is overwhelmingly likely. The threading is more important than the viewpoint threaded. Other viewpoints having a predictive edge at phrase and bar levels include (DurRatio $\otimes$ (ScaleDegree $\ominus$ LastInPhrase))$_{SB}$, (DurRatio $\otimes$ (ScaleDegree $\ominus$ FirstInBar))$_{SATB}$ and (DurRatio$\otimes$(Interval$\ominus$FirstInBar))$_{SATB}$. Meanwhile, (Cont $\otimes$ PositionInBar)$_{SATB}$ is able to infer metrical structure and its correlation with Cont. Viewpoint (Interval $\otimes$ InScale)$_{SB}$ is able to model how the soprano and bass lines move in relation to each other; its use of InScale means that the STM is able to play a large part in the prediction process. (Interval $\otimes$ (ScaleDegree $\ominus$ Tactus))$_{SB}$ and (ScaleDegree $\ominus$ Tactus)$_{SATB}$ are similar in nature (the latter predicting the inner parts).

Finally, we consider version 3 viewpoints with a strong prediction performance. At one end of the scale, there are viewpoints which are composed completely of pairwise links (intra- and inter-layer, able to predict at least one attribute) already seen amongst the better performing version 0, 1 and 2 viewpoints examined in Chapters 7 and 8, such as (PositionInBar $\otimes$ LastInPhrase)$_S$ $\otimes$ (Duration $\otimes$ Metre)$_{ATB}$. At the other are complex viewpoints containing many pairwise links not previously thought of as performing particularly well, such as (Cont)$_S$$\otimes$(DurRatio$\otimes$FirstInBar)$_{AT}$$\otimes$(Duration $\otimes$(ScaleDegree$\ominus$LastInPhrase))$_B$. The evolution of such viewpoints has enabled finer distinctions to be made than were possible by version 1 and 2 viewpoints.

Of the viewpoints analysed here to uncover the cause of their exceptional performance in the prediction of melody and harmony, the vast majority are new to this research. Viewpoints common to melodic and harmonic modelling which can only predict Duration fulfil the same roles in each case (albeit that expansion of the harmony changes the statistics). Such viewpoints which predict Pitch clearly change their emphasis from melody to harmony (not least because the melody is given). There are a large number of viewpoints which are specific to harmony, not least because Cont is not relevant to melodic modelling, and also because version 3 allows more than two primitive viewpoints to be linked. We have seen many instances of predictions agreeing with intuitive or music theoretic expectations, leading to the conclusion that the selected viewpoints are performing well at the task of finding correlations of various kinds in the corpus and in individual pieces.

The generation of melody and harmony by means of random sampling was explained early in Chapter 9, including the use of probability thresholds to modify the procedure. Probability thresholds were required because music with a high cross-entropy (and subjectively low quality) was being consistently generated. By optimising the probability thresholds such that mean cross-entropy of generation for each subtask approximately matched that of the prediction of the corpus (using ten-fold cross-validation), it was subsequently possible to generate melody and harmony of higher quality which was more amenable to comparison.

An interesting finding arising from the work on probability thresholds is that when predicting a test data set, the vast majority of predictions are of the highest probability. This situation improves when moving from a single stage of prediction (version 1) to prediction in more than one stage (versions 2 and 3). In addition, the percentage of predictions below optimised thresholds falls from 10.18 for version 1 to 5.00 for version 3. This suggests that version 3 is particularly good at separating appropriate from inappropriate predictions in terms of their probabilities, which in turn indicates that it should be able to generate better harmony.

Whereas the improvement to the generated melodies was immediately obvious, the quality of the generated harmony was not consistently high; therefore harmonisations at either end of the spectrum were analysed for each of versions 1, 2 and 3. The qualitative evidence presented in §9.5.2 points to versions 2 and 3 being capable of producing better harmony than version 1, partially corroborating the quantitative evidence of Chapter 6. The evidence also suggests that version 2 can create better and worse harmony than version 3; that is, version 3 appears to produce more consistent results. It should be borne in mind, however, that it was possible to present only a tiny sample of generated harmonisations here; so the results of this chapter should be regarded as indicative only.

The fact that it is not possible to consistently generate high quality music in the style of the corpus is ample evidence that the models are not yet good enough. Indeed, it was never expected that the ultimate harmonisation model would be found amongst versions 1 to 3. In the meantime, the current models can be coerced into generating music of higher quality more consistently by ramping up the probability threshold parameters. This has the effect of creating music of generally lower cross-entropy than the corpus, with the result that there is less variety in different harmonisations of the same melody. To the listener, the music can become more predictable and therefore less interesting.

In Chapter 10, ideas for improving versions 0 to 3 were discussed in detail, covering such topics as viewpoints, attribute prediction, model combination, viewpoint selection and representation. This was followed by an exposition of versions 4 to 9, which further extend the multiple viewpoint framework.

11.2 Research Contributions

This research directly contributes to the disciplines of basic artificial intelligence and computational musicology, while making indirect contributions to cognitive science and applied artificial intelligence (see §1.2.1).

11.2.1 Basic AI

Basic AI is concerned with computational techniques relevant to the simulation of intelligence. From this point of view, we can regard the machine learning of musical style from a corpus (which exhibits apparently intelligent behaviour) as a means of testing more generally applicable computational techniques.

Pearce (2005) introduced a viewpoint (feature) selection algorithm based on forward stepwise selection which evaluated all 54 of the primitive and linked viewpoints in his pool at each iteration. Bearing in mind that the pool has now been expanded to 609 viewpoints, it was necessary to modify the algorithm to make it more time efficient (see §3.4.5). A procedure similar to that described by Pickens and Iliopoulos (2005) for Markov random field induction was developed, whereby at each iteration we try only primitive viewpoints, and primitive viewpoints already in the partially selected multiple viewpoint system linked with one other primitive viewpoint. The algorithm was further developed to cope with the more complex version 3 viewpoints. Other minor variants of this procedure were also described. Algorithms 3.1 to 3.7 were formally presented in §3.4.5.4. The modified algorithm is unlikely to find the globally best multiple viewpoint system; but this was also the case for the original algorithm. This contribution allows viewpoint selection involving large numbers of viewpoints (not only musical ones) to be carried out within a reasonable timescale.

A great deal of guidance was given in Chapter 4 about the construction of derived and linked viewpoint domains, based on the principle that between them, the members must be able to predict all of, and only, the members of the basic domain. This guidance culminated in the formal presentation of Algorithms 4.1 to 4.4 (see §§4.5.2 and 4.5.3), for use in the construction of the most complex (version 3) inter-layer linked viewpoint domains. An unreliable domain construction procedure might, for example, result in one or two predictions being missing from some basic viewpoint probability distributions, which means that at some point a succession of probabilities will be erroneously combined; hence the importance of this contribution.

Three versions of the multiple viewpoint framework for harmony have been expounded and implemented as software. In principle, these developments are also applicable to any set of interrelated sequences, such as time-stamped economic or financial data. Version 1 is not a contribution as such, because it makes use of vertical viewpoint elements in exactly the same way as Conklin (2002); it is a baseline model for purposes of comparison. An empirical analysis of the time complexity of this version (as implemented) has been carried out, however (see §4.6), which is a contribution. The time complexity of a complete prediction run is $O(vn_c^2)$, and that of a viewpoint selection run is $O(v_pvn_c^2)$ (with certain provisos; see §4.6.4), where n_c is the number of events in the corpus, v is the number of viewpoints in the multiple viewpoint system and v_p is the number of primitive viewpoints in the pool used during viewpoint selection.[1] Although version 2 draws its inspiration from previous work (Allan, 2002; Hild et al., 1992; Phon-Amnuaisuk and Wiggins, 1999), the use of subtasks is new to the multiple viewpoint framework. Specifically, this version is able to predict or generate harmony one or more

[1] The effect of maximum N-gram model order was not investigated.

parts at a time, in any order. Version 3 is novel, in that it allows the inter-layer linking of different viewpoints on different layers. In addition, linking with support layers (given parts) is not compulsory.

A further six versions of the multiple viewpoint framework for harmony have been developed, but not yet implemented (see Chapter 10). Version 4 tries an alternative (offset) inter-layer linking method inspired by Allan (2002), but new to the multiple viewpoint framework. This arrangement introduces a limited look-ahead capability, which is increased in version 5 by allowing future context. Version 6 uses cross-entropy to guide the construction of back-off sequences; and version 7 allows the use of differently shaped contexts of the same order within a back-off sequence. Versions 8 and 9 introduce more flexible inter- and intra-layer linking respectively. The analysis inherent in the proposal of these novel additional versions is a contribution.

In §6.2.3 and §6.3.3 it is shown that the selection of specialist multiple viewpoint systems to individually predict `Duration`, `Cont` and `Pitch` results in better (lower cross-entropy) overall harmonic models than the selection of a single system to predict all three attributes (the difference in performance is greater when using the augmented `Pitch` domain compared with the seen domain). A similar, but very small, effect is noted with respect to melodic modelling in §5.3.3. It is highly likely that this result is applicable beyond the modelling of music.

Finally, although version 1 performs least well in conjunction with the corpus used during viewpoint selection (see §11.2.2), it appears that its more general models are better able to scale up to larger corpora (which may deviate somewhat from the characteristics of the original corpus) than those of versions 2 and 3. This can be an advantage, since viewpoint selection with a large corpus is extremely time-consuming. It is considered highly likely that this result too is applicable beyond the realms of music modelling.

11.2.2 Computational Musicology and Cognitive Science

The interdisciplinary field of computational musicology is one in which computational techniques are employed in pursuit of answers to musicological questions (Volk et al., 2011), while cognitive science seeks to model the intelligence of living beings. Inasmuch as the research described here makes progress in the statistical modelling of musical style, it is also potentially useful for the cognitive modelling of music (see, *e.g.*, Pearce 2005); therefore these parts of the research contribute directly to computational musicology and indirectly to cognitive science.

Fifteen new primitive viewpoint types (nine of them threaded) were introduced in §3.2.4.2 including basic viewpoint type `Cont`, which was required to retain information which would otherwise be lost on carrying out a full expansion of the corpus. In addition, whereas there were only a limited number of ways in which such viewpoints could be linked in, for example, Pearce (2005), in this research any primitive viewpoint may be linked with any other, provided that such links are able to predict at least one attribute. We see in Chapter 7 that many of the better performing linked viewpoints with respect to the modelling of melody are new to this research, thereby demonstrating their contribution. Some of them are new linked viewpoints comprising existing primitive viewpoints, while others contain new primitive viewpoints. Chapter 8 provides evidence that new viewpoints are also amongst the better performing in relation to harmonic modelling, further demonstrating their contribution.

Taking the Cartesian product of the pitches seen in the soprano, alto, tenor and bass parts in the corpus produces 157,320 vertical $(\texttt{Pitch})_{SATB}$ viewpoint elements. The use of a domain of this size results in exceedingly slow run times, and in any case a vast number of such pitch combinations would never be seen in music. Utilising only elements seen in the corpus and test data works well for melody (Pearce, 2005) but is severely limiting for harmony: 882 different elements are present in corpus and test data 'A', increasing by about 400 on the addition of corpus and test data 'B'. A means of taking account of elements as yet unseen would be of benefit in the modelling of harmony. The novel solution described in §4.3.1 is based on the notion that a chord seen in one key (on a degree of scale basis) should be applicable to any key. Seen chords are transposed up and down a semitone at a time until one of the parts goes out of its range. Elements produced in this way which are not currently in the *augmented* domain are added to it. This contribution allows viewpoints such as `ScaleDegree` to make use of vertical $(\texttt{Pitch})_{SATB}$ viewpoint elements not seen in the corpus when harmonising melodies, as well as ensuring more realistic prediction probability distributions.

In previous research, the LTM and STM have been combined by amalgamating the viewpoint distributions within each of these models first, using one bias, and then combining the two resulting distributions using another. Pearce (2005) proposed two possible alternative methods, but did not empirically compare them. The first effects a pair-wise combination of the distributions of identical viewpoints in the LTM and STM first, using one bias, and then combines the resulting distributions using another. The second combines all viewpoint distributions at once, irrespective of whether they are in the LTM or STM, using a single bias. A comparison in §5.2.4 demonstrates that the original scheme is best, thereby making a contribution.

A comparison in §6.8 of the best version 0 to 3 multiple viewpoint systems for the prediction of `Duration`, `Cont` and `Pitch` separately led to the following conclusions of musicological interest: there appears to be little correlation between `Duration` and `Cont`; primitives derived from `Pitch` are heavily involved in `Duration`-predicting systems; while `Duration` and `DurRatio` are almost absent from `Pitch`-predicting systems.

It is demonstrated in §6.3.1.1 that a better performance is achieved by predicting the bass part first followed by the inner parts together than by starting with the prediction of alto or tenor. This reflects the usual human approach to harmonisation. It is interesting to note that this heuristic, almost universally followed during harmonisation, therefore has an information theoretic explanation for its success. Bearing in mind that `Duration` prediction is not comparable in versions 1 and 2, a comparison of `Cont` and `Pitch` prediction in §6.4.3 shows that version 2 (with bass predicted first) performs better than version 1. This is further vindication of the bass first approach to harmonisation.

In §6.5.1 and §6.6.1 it is shown that version 3 performs better than versions 1 and 2. Furthermore, a worthwhile performance enhancement is attainable by creating a hybrid overall model comprising the better of version 2 and 3 subtask models. In Chapter 9, further quantitative evidence with respect to probability thresholds supports the conclusion that the baseline version 1 has been improved upon. Finally, the proposed viewpoint-based method for carrying out information theoretic music analysis outlined in §11.3.1 is a contribution.

11.2.3 Applied AI

Applied AI is concerned with the designing and building of products incorporating AI techniques. It is a field relevant to this research, but not specifically addressed by it in this thesis. At the very least, the direct contributions to basic AI and computational musicology could be utilised in the construction of systems designed to assist with the process of composition. It is also possible that such contributions could be adapted for use in music information retrieval systems and real-time improvisational or accompanying systems (*e.g.*, improving or adding more variety to the auto-harmonisation feature found in some MIDI keyboards). They may therefore also be considered indirect contributions to applied AI.

11.3 Future Work

Ways in which the multiple viewpoint framework may be developed (and hopefully improved) were discussed in some detail in Chapter 10, which in itself will entail a great deal of future work. Some more general thoughts about the direction of the research are outlined below. Details of how information theoretic music analysis may be achieved using multiple viewpoints are given in §11.3.1, along with an example. §11.3.2 to §11.3.5 briefly cover polyphonic music, cognitive models, real time generation of harmony and neural networks.

11.3.1 Music Analysis

We consider here how information theory may be applied to the reductional music analysis of Lerdahl and Jackendoff (1983) or Schenker (1979). Since analytical complexity increases with the length of a piece, it would seem prudent to keep pieces short in the first instance. A corpus and test data comprising harmonised hymn tunes, as used in the current research, is a suitable starting point. A multiple viewpoint approach seems appropriate, with search guided by cross-entropy. It should be possible to quickly adapt the existing harmonic modelling software to carry out analysis in this way. The work would focus on reduction to analytical layers close to the musical foreground (or surface) to begin with. The analysis used to illustrate the proposed technique is from Lerdahl and Jackendoff (1983), but with the tactus changed from crotchet to minim as in Vaughan Williams (1933), hymn no. 102.

11.3.1.1 First Reduction Layer

The top system of Figure 11.1 shows the first two phrases of one of Bach's harmonisations of *Passion Chorale*; the middle system shows Lerdahl and Jackendoff's reduction, resulting from the removal of sub-tactus unessential notes; and if the alternative choice were made in each case, we would end up with what is shown on the bottom system. The harmony is modelled in expanded form, such that there is a sequence of block chords, using version 1. This time we are not assuming that the melody is given, and only `Pitch` is predicted (its domain is assumed not to be constrained by previously predicted `Cont` values). In this example, `Pitch` prediction probabilities are assigned to each chord by an LTM (using a bias of 2) comprising viewpoint models of arbitrary maximum order 3, learned from corpus 'A+B'.[2] Schenker (1979) insists that normal voice leading rules hold at deeper levels of analysis, so statistics gathered from the musical surface are also likely to be applicable at deeper levels. Whereas before the idea was to find a multiple viewpoint system to minimise cross-entropy, the idea now is to find a system which performs well at distinguishing between good and poor chord choices during the reduction process.

The single primitive viewpoint `ScaleDegree` was used to determine cross-entropies in the first instance, for the entire piece; see Table 11.1. The fact that the Lerdahl and Jackendoff reduction has a higher cross-entropy than the original harmony is a bit of a surprise; but the important thing is that the alternative reduction has a much higher cross-entropy, giving a difference of 1.48 bits/chord between the reductions. By linking Boolean viewpoint `Tactus` with `ScaleDegree`, the difference increases to 2.02 bits/chord; and by adding a second viewpoint, `Duration` ⊗ `ScaleDegree`, the difference increases to 2.33 bits/chord. Noticing the downward trend in cross-entropy for the original harmony and the Lerdahl and Jackendoff reduction, it may be tempting to think that minimising the cross-entropy of the original harmony will automatically increase the difference in reduction cross-entropies; but this is not the case. In the final row, replacing `Duration` ⊗ `ScaleDegree` by `Duration` ⊗ `Interval` reduces the cross-entropy of the original harmony to 6.18 bits/chord; but the difference in reduction cross-entropies is only 1.86 bits/chord.

Simply by using trial and error, it has been possible to quickly find a small multiple viewpoint system which seems to perform well at distinguishing good and poor chord choices during analysis. Obviously more than one chorale should be used in the view-

[2] Since we are analysing *Passion Chorale* (Vaughan Williams 1933, hymn no. 102), it has been replaced in the corpus by *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97).

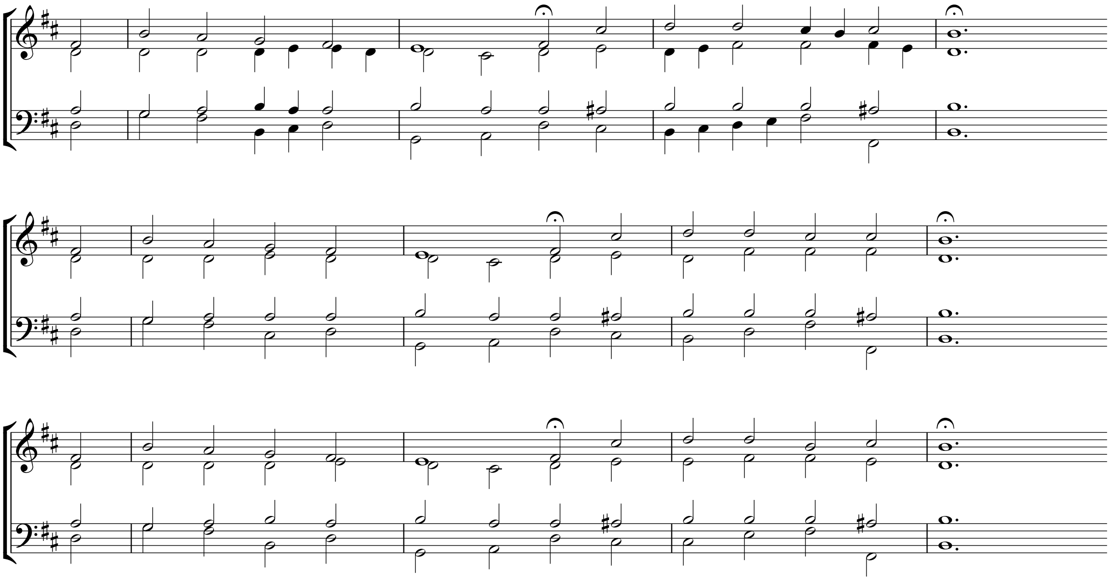


Figure 11.1: The top system shows the first two phrases of one of Bach's harmonisations of *Passion Chorale*; the middle system shows Lerdahl and Jackendoff's first reduction layer; and the bottom system shows the result of alternative analytical choices.

Multiple viewpoint system	Orig. harm.	L & J redn.	Alt. redn.	Δ redn. x-ent.
{`ScaleDegree`}	8.34	8.76	10.24	1.48
{`ScaleDegree` ⊗ `Tactus`}	7.80	8.36	10.38	2.02
{`ScaleDegree` ⊗ `Tactus`, `Duration` ⊗ `ScaleDegree`}	7.69	8.27	10.60	2.33

Table 11.1: Cross-entropies (bits/chord) are shown for the original *Passion Chorale* harmony (*Orig. harm.*), the Lerdahl and Jackendoff reduction (*L & J redn.*) and an alternative reduction (*Alt. redn.*), using three different simple viewpoint systems. The difference in reduction cross-entropies (Δ *redn. x-ent.*) is also shown.

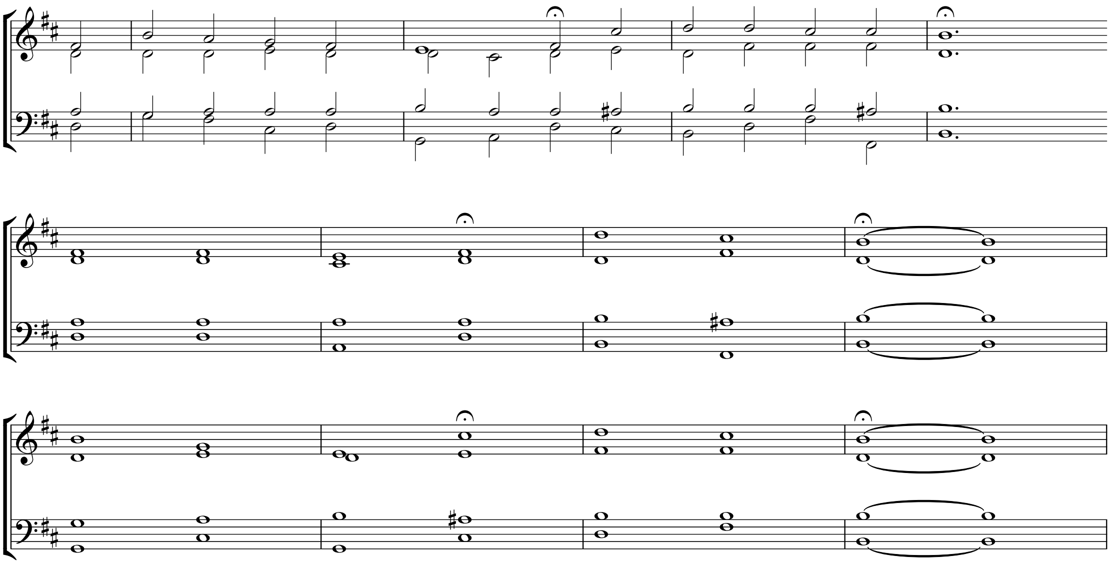


Figure 11.2: Continuation of the analysis of *Passion Chorale*. The top system shows the first two phrases of Lerdahl and Jackendoff's first reduction layer (the starting harmony); the middle system shows their second reduction layer; and the bottom system shows the result of alternative analytical choices.

point selection process. Ideally, we would employ ten-fold cross-validation of the corpus, encouraging generalisation to a good deal of unseen data. By modifying the existing viewpoint selection algorithm to maximise the difference in cross-entropy between good and poor reductions, it should be possible to find even better performing systems.

11.3.1.2 Second Reduction Layer

We now use Lerdahl and Jackendoff's first reduction layer as the new starting harmony. The middle system of Figure 11.2 shows their next reduction, and the bottom system shows alternative chord choices. Here, the deeper Lerdahl and Jackendoff reduction does indeed have a lower cross-entropy than the starting harmony (see Table 11.2); and the difference in reduction cross-entropies is very large, even when using primitive viewpoint `ScaleDegree` alone. Adding `Interval` ⊗ `ScaleDegree` increases the difference to 4.26 bits/chord. Again, automatic viewpoint selection should come up with even better systems.

Having found a suitable multiple viewpoint system, we are in a position to perform a search to find the most likely reduction. Obviously such searches would be done on pieces not used during the learning process, but it is convenient to use the same example. Since altered chords become part of the context for subsequent predictions, we cannot simply make choices in a linear fashion as we progress through the piece. In the top system of Figure 11.1, for example, we would need to try four combinations of chords in the second half of the first bar, and sixteen combinations in the third bar. The use of a maximum order 3 model means that the third bar is completely independent of the first,

Multiple viewpoint system	Start. harm.	L & J redn.	Alt. redn.	Δ redn. x-ent.
{`ScaleDegree`}	8.76	7.88	11.54	3.66
{`ScaleDegree`, `Interval` ⊗ `ScaleDegree`}	8.66	7.75	12.01	4.26

Table 11.2: Continuation of the analysis of *Passion Chorale.* Cross-entropies (bits/chord) are shown for the new starting harmony (*Start. harm.*), the deeper Lerdahl and Jackendoff reduction (*L & J redn.*) and an alternative reduction (*Alt. redn.*), using two different simple viewpoint systems. The difference in reduction cross-entropies (Δ *redn. x-ent.*) is also shown.

however, which reduces the search space. In the middle system, by treating musical phrases as essentially independent the search space need not become unmanageable. The assumption of phrase independence will need to be abandoned at some deeper level of analysis, though. The resulting reductions would then be evaluated by comparison with those of an expert human analyst. An obvious option for follow-up research is to do further work on information theoretic music analysis at levels closer to the Ursatz (Schenker, 1979). Once the background level of these simple pieces of music have been successfully reached, we could perhaps go on to tackle more complex music such as Bach's forty-eight preludes and fugues (see §11.3.2 below) or Mozart's piano sonatas.

11.3.2 Polyphonic Music

The existing implementation accommodates the limited independent movement of parts in the corpora by means of viewpoint `Cont`. It is likely that this viewpoint will also cope with the more extreme independent movement inherent in polyphony. In order to investigate the effectiveness of the system in this sphere, a corpus of fully polyphonic music will be created (Bach's forty-eight preludes and fugues, for example, could be a useful resource). The issue of how to implement rests will definitely need to be resolved before the system can be utilised in this way, and the system must be capable of modelling music in two, three or four parts (at least).

11.3.3 Cognitive Models

Most of the primitive viewpoints currently used are also employed in the construction of successful cognitive models of melodic expectancy (Pearce, 2005). In §5.2.2, it was noted that all of the (linked) viewpoints automatically selected for long-term models were new to this research. If the better performing of these viewpoints were added to the available pool for use in the construction of cognitive models of melodic expectancy (Pearce, 2005), it is conceivable that such models (already good) could be improved. It is anticipated that these viewpoints *in tandem* with the techniques developed during this research will be especially conducive to the creation of a future overall computational model for the cognition of harmonic movement. Cognitive science research methodologies would be brought to bear in the development of such a model, which will be different in terms of high level structure from a model which harmonises melodies. It is clear that certain structures can be immediately ruled out; for example, it would make no sense to predict a complete bass line before starting to predict harmonic function symbols. Predicting harmonic symbols at cadences before those at the beginning of the harmonic sequence would be equally nonsensical. In a cognitive model, prediction must be done in approximately chronological order,[3] with the whole of a simultaneity being predicted (possibly by the application of a sequence of subtask models) before moving on to the next.

11.3.4 Real Time Generation of Harmony

It is conceivable that, given sufficient time and effort, the system could be adapted such that a real time audio input replaces the given melody in machine-readable format. Each note of the input in turn would be converted to the usual format and then harmonised using a suitable statistical model. The machine-readable harmony would then be converted to MIDI messages, which are sent to a MIDI instrument to produce audible harmony. The harmony is unlikely to be as good as that generated for a complete given melody in machine-readable format, since no structural information is available for the melody beforehand. There is also the issue of how to deal with intentional and unintentional changes in note duration, including rubato, accelerando and rallentando. Some form of beat tracking mechanism (Dixon and Cambouropoulos, 2000) would need to be implemented.

11.3.5 Neural Networks

Conklin and Cleary (1988) suggest the intriguing possibility of implementing multiple viewpoint systems as neural networks, in order to exploit parallel computation and weight adaptation. Although there are no immediate plans to work on this, it is worth keeping in mind for the more distant future.

[3] Prediction is not necessarily completely chronological because of the phenomenon of retrospective listening (Narmour, 1992).

Appendix A

The Computational Modelling of Creativity

Wiggins (2006a, p. 451) gives some relevant and useful definitions, which are reproduced here. *Creativity* is defined as "the performance of tasks which, if performed by a human, would be deemed creative." *Computational creativity* is then defined as "the study and support, through computational means and methods, of behaviour exhibited by natural and artificial systems, which would be deemed creative if exhibited by humans." A *creative system* is defined as "a collection of processes, natural or automatic, which are capable of achieving or simulating behaviour which in humans would be deemed creative." *Creative behaviour* is "one or more of the behaviours exhibited by a *creative system*." *Novelty* is "the property of an artefact (abstract or concrete) output by a *creative system* which arises from prior non-existence of like or identical artefacts in the context in which it is produced." Finally, *value* is "the property of an artefact (abstract or concrete) output by a *creative system* which renders it desirable in the context in which it is produced."

Boden (2004) gives a descriptive account of human creativity, and discusses the role of computers in aiding our understanding of it. She defines creativity as "the ability to come up with ideas or artefacts that are *new, surprising and valuable*" (Boden, 2004, p. 1). Of course, it is possible for someone, unaware that a particular idea or artefact has already come into being, to come up with that idea or artefact independently; the concepts P-creativity (psychological) and H-creativity (historical) have been introduced to take this into account. Irrespective of the latter distinction, three main types of creativity are identified: combinational, exploratory and transformational. Combinational creativity involves the bringing together of existing, possibly disparate, ideas in unfamiliar ways; exploratory creativity involves the exploration of a *conceptual space* (a structured style of thought, relevant to a particular domain); and transformational creativity involves transforming a conceptual space in such a way that the previously unthinkable is made accessible.

Wiggins (2006a) formalises many of the above ideas with the intention of creating a framework within which creative systems can be compared in detail, and so better understood. He agrees that novelty and value are important aspects of creativity, but does not see surprise as a fundamental property of creative systems, arguing that it is the perceiver of a created artefact who might be surprised, and that this surprise is likely to be due to the novelty of the artefact. Only two types of creativity are considered in this formalisation: exploratory and transformational.

Within this framework, an exploratory creative system is expressed as follows:

$$\langle \mathcal{U}, \mathcal{L}, [\![.]\!], \langle\!\langle ., ., . \rangle\!\rangle, \mathcal{R}, \mathcal{T}, \mathcal{E} \rangle$$

where $\mathcal{U}$ is a multidimensional space called the *universe* capable of representing all possible complete and partial concepts, including $\bot$, the empty concept; $\mathcal{L}$ is a language capable of expressing sets of rules; $\mathcal{R}$ is a set of rules which defines a conceptual space $\mathcal{C}$ (which must comply with both $\bot \in \mathcal{C}$ and $\mathcal{C} \subseteq \mathcal{U}$); $\mathcal{T}$ is a set of rules governing the traversal or searching of $\mathcal{C}$; $\mathcal{E}$ is a set of rules which assesses the value of concepts found in $\mathcal{C}$; $[\![.]\!]$ is an interpretation function such that, for example, $\mathcal{C} = [\![\mathcal{R}]\!](\mathcal{U})$; and $\langle\!\langle ., ., . \rangle\!\rangle$ is another interpretation function such that $c_{out} = \langle\!\langle \mathcal{R}, \mathcal{T}, \mathcal{E} \rangle\!\rangle (c_{in})$, where c_{in} and c_{out} are ordered subsets of $\mathcal{U}$.

Having separate rule sets $\mathcal{R}$ and $\mathcal{T}$ provides the means to model the behaviour of more than one creative agent within the same conceptual space: "so now we have, for example, the ability to simulate two composers working in different ways within the same style..." (Wiggins, 2006a, p. 453). It is appropriate at this point to describe in more detail how the traversal mechanism works. $\mathcal{R}$ and $\mathcal{E}$ are included in the interpretation function so that $\mathcal{T}$ can reason about them (although it need not do so). Search is initiated by applying the resulting function to the set containing only the empty concept:

$$c_{out} = \langle\!\langle \mathcal{R}, \mathcal{T}, \mathcal{E} \rangle\!\rangle (\{\bot\}).$$

At this point, only the concepts in the set c_{out} have been found; valued concepts can be determined using $[\![\mathcal{E}]\!](c_{out})$. To continue the search, c_{out} becomes c_{in}; the function can choose to operate on any number of concepts in c_{in}, from anywhere in the ordering. The result is a new c_{out}. The complete set of valued concepts is given by

$$[\![\mathcal{E}]\!](\langle\!\langle \mathcal{R}, \mathcal{T}, \mathcal{E} \rangle\!\rangle^{\Diamond}(\{\bot\}))$$

where $\mathcal{F}$ is a set-valued function

$$\mathcal{F}^{\Diamond}(X) = \bigcup_{n=0}^{\infty} \mathcal{F}^{n}(X).$$

Wiggins (2006a) argues that there are two distinct types of transformational creativity; one involving changes to $\mathcal{R}$ ($\mathcal{R}$-transformational) and one involving changes to $\mathcal{T}$ ($\mathcal{T}$-transformational). Changing $\mathcal{R}$ changes the set of concepts that constitutes the conceptual space $\mathcal{C}$. After such a change, it might be possible, for example, for a creative agent to find valid (and possibly valued) artefacts that could not previously have been found in $\mathcal{C}$. Any search strategy $\mathcal{T}$ is not guaranteed to find all concepts in $\mathcal{C}$; therefore it is possible, for example, for a change in $\mathcal{T}$ to result in the finding of concepts (possibly valued) that, although they were already in $\mathcal{C}$, were previously inaccessible.

Bundy (1994) suggests that transformational creativity can be understood as exploratory creativity at the meta-level; this is also formalised by Wiggins (2006a). A transformational creative system is expressed as follows:

$$\langle \mathcal{L}, \mathcal{L}_{\mathcal{L}}, [\![.]\!], \langle\langle ., ., . \rangle\rangle, \mathcal{R}_{\mathcal{L}}, \mathcal{T}_{\mathcal{L}}, \mathcal{E}_{\mathcal{L}} \rangle$$

where language $\mathcal{L}$ becomes the universe; $\mathcal{L}_{\mathcal{L}}$ is a meta-language capable of expressing sets of rules at the meta-level; $\mathcal{R}_{\mathcal{L}}$ is a rule set defining a meta-level conceptual space $\mathcal{C} \subseteq \mathcal{L}$ (with $\bot \in \mathcal{C}$); $\mathcal{T}_{\mathcal{L}}$ is a rule set governing the traversal or searching of $\mathcal{C}$; $\mathcal{E}_{\mathcal{L}}$ is a rule set which evaluates the quality of the transformational creativity; and $[\![.]\!]$ and $\langle\langle ., ., . \rangle\rangle$ perform the same functions as before (it is assumed that they are able to interpret both $\mathcal{L}$ and $\mathcal{L}_{\mathcal{L}}$).

The role of $\mathcal{E}_{\mathcal{L}}$ needs clarifying, as it is rather different from that of $\mathcal{E}$ in exploratory creativity; $\mathcal{E}_{\mathcal{L}}$ selects pairs of $\mathcal{R}_{\mathcal{L}}$ and $\mathcal{T}_{\mathcal{L}}$ such that concepts valued by $\mathcal{E}$ are found. In mathematical terms, the following must be complied with:

$$\{c | r \in \langle\langle \mathcal{R}_{\mathcal{L}}, \mathcal{T}_{\mathcal{L}}, \mathcal{E}_{\mathcal{L}} \rangle\rangle^{\Diamond}(\{\mathcal{R}\}) \wedge t \in \langle\langle \mathcal{R}_{\mathcal{L}}, \mathcal{T}_{\mathcal{L}}, \mathcal{E}_{\mathcal{L}} \rangle\rangle^{\Diamond}(\{\mathcal{T}\}) \wedge \ c \in [\![\mathcal{E}]\!](\langle\langle r, t, \mathcal{E} \rangle\rangle^{\Diamond}(\{\bot\}))\} \neq \emptyset.$$

In other words, for valid transformational creativity to occur, sets of possible $\mathcal{R}$ and $\mathcal{T}$ are enumerated from the recursive application of meta-level rules to the original $\mathcal{R}$ and $\mathcal{T}$, and then a new combination of $\mathcal{R}$ and $\mathcal{T}$ is chosen from these sets (it is possible for either $\mathcal{R}$ or $\mathcal{T}$ to remain unchanged). At least one valued concept must be found by the recursive application of this new combination of rule sets, using the empty set as the starting point.

Wiggins (2006b) compares the creative system framework described above with traditional AI state space search. All AI search algorithms are expressible in terms of four main components: the *representation* is able to describe both incomplete and complete solutions; the *solution detector* recognises a valid solution and stops the search; the *agenda* is an ordered list of states to be processed; and the *expansion operator* acts on a state to produce one or more new ones. The general algorithm described by Wiggins (2006b) is reproduced here in pseudo-code in the style of Corman et al. (2001):

```
do apply solution detector to top agenda item
   if top agenda item is a solution
      then output top agenda item
           terminate
      else remove top item from agenda
           apply expansion operator to removed item
           add new items to agenda.
```

Two straightforward ways of implementing the final step of this algorithm are to add the new items either to the top or to the bottom of the agenda, resulting in depth-first and breath-first search algorithms respectively. Other search algorithms use a *heuristic* to order the agenda.

Wiggins (2006b) concludes that there are four ways in which the creative system framework can be seen as a generalisation and clarification of traditional AI state space search:

1. If we take AI state space and conceptual space to be, to all intents and purposes, equivalent, then the creative system framework is more general inasmuch as it contains universe $\mathcal{U}$, which is capable of representing much more than is in a state space. $\mathcal{R}$-transformation can admit additional concepts from $\mathcal{U}$ into a conceptual space.

2. The ordered list of the creative system framework is the same as the agenda of state space search; but whereas in the latter case the expansion operator is only applied to the top item, in the former it can be applied to one or more items from anywhere in the list, which makes the creative system framework more general.

3. The creative system framework clarifies the distinction between rules for assessing the quality of concepts and rules governing the search strategy.

4. State space search is able to detect when a valid solution has been found, but has no way of assessing whether one solution is better than another. The creative system framework is equipped to make such comparisons.

Appendix B

Corpora and Test Data

The hymn tunes in corpora ‘A’, ‘B’ and ‘A+B’ are listed in their respective sub-corpora. Listed also are hymn tunes in the associated sets of test data. The hymn number (Vaughan Williams, 1933) is followed by the name of the tune. Note that some distinctly different tunes have the same name (*e.g.*, St. Thomas), and that some hymns have more than one tune (*e.g.*, hymn no. 138).

B.1 Corpus ‘A’

- 45 Crüger, 61 Illsley, 80 Solomon, 52 Wareham, 103 Allein Gott in der Höh sei Ehr.
- 83 Bedford, 34 Barratt, 316 Saffron Walden, 9 Winchester New, 106 Horsley.
- 82 Stockton, 43 Dundee, 35 Wer da wonet, 20 This Endris Nyght, 105 Batty.
- 31 St. Thomas, 5 Merton, 81 St. Bartholomew, 11 St. Thomas, 91 Valor.
- 21 Yorkshire (or Stockport), 63 Tantum Ergo, 485 Sandys, 40 Stuttgart, 102 Passion Chorale.
- 17 Vom Himmel hoch, 71 St. Bernard, 85 Harington (Retirement), 16 Newbury, 104 Nun lasst uns geh’n.
- 14 Puer Nobis Nascitur, 56 Richard, 139 St. Fulbert, 267 Tallis’ Canon, 86 Innsbruck.
- 75 St. Raphael, 452 Knecht, 32 Wohlauf, thut nicht verzagen, 49 St. Gregory (Zeuch meinen Geist), 93 University.
- 15 Forest Green, 201 Boyce, 23 Dent Dale, 39 Dix, 107 Caton (or Rockingham).
- 4 Luther’s Hymn (Nun freut euch), 26 Noel, 42 Was lebet, was schwebet,[1] 29 A Virgin Unspotted, 98 Song 46.

[1] Notes in small type in the score have been included.

B.2 Test Data ‘A’

- 33 Grafton, 47 St. Edmund, 37 Innocents, 36 Das walt’ Gott Vater, 97 Das ist meine Freude.

B.3 Corpus ‘B’

- 220 Brockham, 131 Ave Virgo Virginum, 132 Gott des Himmels, 206 Crediton, 217 Farley Castle.

- 119 Zu meinem Herrn, 118 Jesu meines Glaubens Zier, 177 Harts, 137 Ellacombe, 216 St. Alban.

- 128 Salzburg, 156 Veni Creator (Attwood), 195 Bremen, 173 Herr Jesu Christ, 211 Old 44th.

- 213 Ave Maris Stella, 179 Ach Gott von Himmelreiche, 138 Mach’s mit mir Gott, 158 Stroudwater, 218 Den des Vaters Sinn geboren.

- 120 Omni Die, 155 Veni Sancte Spiritus, 168 Manchester, 115 Christi Mutter stund vor Schmerzen, 169 Urbs Coelestis.

- 222 Rhyddid, 167 Duke Street, 209 Old 120th, 134 Christ ist erstanden, 180 Leighton.

- 199 Ballerma, 147 St. Magnus (Nottingham), 148 Nun freut euch, 133 Easter Hymn (original version), 187 Weimar.

- 204 All Saints, 126 Nun lasst uns Gott dem Herren, 186 Balfour, 115 Stabat Mater, 193 St. Ambrose.

- 171 St. Edmund, 133 Easter Hymn (altered version), 135 Savannah (or Herrnhut), 197 Song 67, 223 Herr, deinen Zorn.

- 151 Aeterna Christi Munera (Rouen), 196 Mount Ephraim, 162 Nicaea, 190 Carlisle, 145 In Babilone.

B.4 Test Data ‘B’

- 127 Würzburg, 144 Bromsgrove, 150 Monte Cassino, 138 Dies ist der Tag, 152 Down Ampney.

B.5 Corpus 'A+B'

- 56 Richard, 147 St. Magnus (Nottingham), 211 Old 44th, 179 Ach Gott von Himmelreiche, 186 Balfour, 39 Dix, 63 Tantum Ergo, 86 Innsbruck, 485 Sandys, 98 Song 46.
- 197 Song 67, 45 Crüger, 137 Ellacombe, 49 St. Gregory (Zeuch meinen Geist), 75 St. Raphael, 42 Was lebet, was schwebet, 115 Christi Mutter stund vor Schmerzen, 11 St. Thomas, 213 Ave Maris Stella, 316 Saffron Walden.
- 104 Nun lasst uns geh'n, 81 St. Bartholomew, 115 Stabat Mater, 35 Wer da wonet, 26 Noel, 4 Luther's Hymn (Nun freut euch), 34 Barratt, 16 Newbury, 223 Herr, deinen Zorn, 155 Veni Sancte Spiritus.
- 131 Ave Virgo Virginum, 119 Zu meinem Herrn, 17 Vom Himmel hoch, 15 Forest Green, 151 Aeterna Christi Munera (Rouen), 103 Allein Gott in der Höh sei Ehr, 133 Easter Hymn (altered version), 193 St. Ambrose, 118 Jesu meines Glaubens Zier, 31 St. Thomas.
- 120 Omni Die, 85 Harington (Retirement), 201 Boyce, 134 Christ ist erstanden, 158 Stroudwater, 180 Leighton, 222 Rhyddid, 195 Bremen, 167 Duke Street, 138 Mach's mit mir Gott.
- 139 St. Fulbert, 168 Manchester, 452 Knecht, 14 Puer Nobis Nascitur, 218 Den des Vaters Sinn geboren, 9 Winchester New, 148 Nun freut euch, 199 Ballerma, 107 Caton (or Rockingham), 61 Illsley.
- 20 This Endris Nyght, 156 Veni Creator (Attwood), 171 St. Edmund, 82 Stockton, 267 Tallis' Canon, 209 Old 120th, 52 Wareham, 71 St. Bernard, 43 Dundee, 217 Farley Castle.
- 21 Yorkshire (or Stockport), 5 Merton, 106 Horsley, 40 Stuttgart, 190 Carlisle, 135 Savannah (or Herrnhut), 80 Solomon, 91 Valor, 220 Brockham, 93 University.
- 204 All Saints, 132 Gott des Himmels, 216 St. Alban, 169 Urbs Coelestis, 128 Salzburg, 126 Nun lasst uns Gott dem Herren, 162 Nicaea, 206 Crediton, 196 Mount Ephraim, 173 Herr Jesu Christ.
- 145 In Babilone, 105 Batty, 83 Bedford, 32 Wohlauf, thut nicht verzagen, 23 Dent Dale, 29 A Virgin Unspotted, 133 Easter Hymn (original version), 102 Passion Chorale, 187 Weimar, 177 Harts.

B.6 Test Data 'A+B'

- 33 Grafton, 47 St. Edmund, 37 Innocents, 36 Das walt' Gott Vater, 97 Das ist meine Freude, 127 Würzburg, 144 Bromsgrove, 150 Monte Cassino, 138 Dies ist der Tag, 152 Down Ampney.

Appendix C

Single Attribute Performance Comparisons

C.1 Comparison of Version 1 with Version 3

Figure C.1 shows that version 3 is better than version 1 for the prediction of `Duration`. The lowest cross-entropy for the version 3 model is 0.60 bits/prediction at an $\hbar$ of 0 (optimised bias and L-S bias are 130 and 1.7 respectively) compared with 0.68 bits/prediction for the version 1 model. The situation is rather different for the prediction of `Cont`, as seen in Figure C.2. In this case, the optimal version 3 cross-entropy is 0.65 bits/prediction at an $\hbar$ of 2 (with an optimised bias and L-S bias of 2.7 and 5.3 respectively), which is the same as the version 1 cross-entropy to two decimal places. Finally, Figure C.3 shows that the version 3 model is convincingly superior for the prediction of `Pitch`. Here, the lowest cross-entropy for the version 3 model is 3.40 bits/prediction at an $\hbar$ of 3 (optimised bias and L-S bias are 1.3 and 140 respectively) compared with 3.75 bits/prediction for the version 1 model.

C.2 Comparison of Version 2 with Version 3

Figure C.4 suggests that version 3 is better than version 2 for the prediction of `Duration` in the bass given soprano, especially considering how tiny the error bars are. The lowest cross-entropy for the version 3 model is 0.60 bits/prediction at an $\hbar$ of 0 (optimised bias and L-S bias are 130 and 1.7 respectively) compared with 0.67 bits/prediction for the version 2 model. Similarly, for the prediction of `Duration` in the alto/tenor given soprano/bass (see Figure C.5) the version 3 cross-entropies are slightly lower than those of version 2. In this case, the optimal version 3 cross-entropy is 0.42 bits/prediction at an $\hbar$ of 0 (optimised bias and L-S bias are 11.4 and 2.0 respectively) compared with 0.47 bits/prediction for version 2.

Figure C.6 indicates, in general, very slightly lower cross-entropies for the version 2 prediction of `Cont` in the bass given soprano. The lowest cross-entropy for the version 3

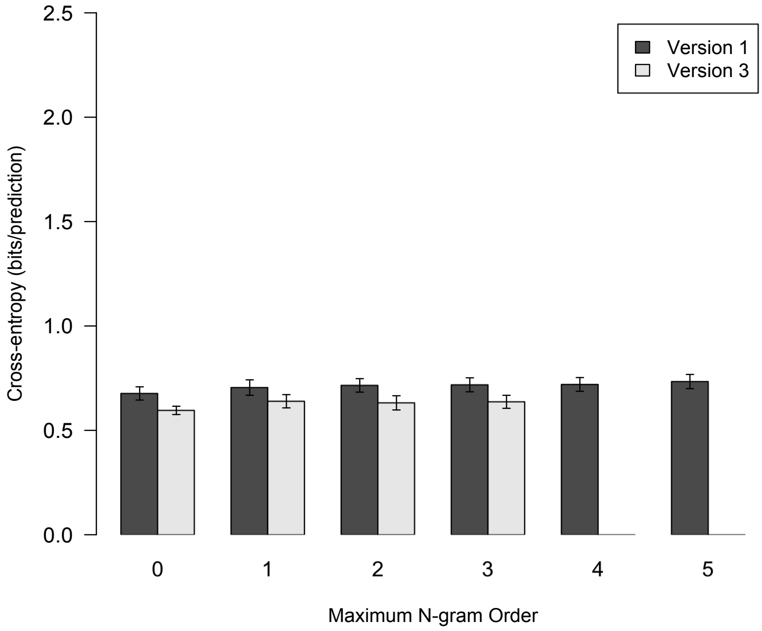


Figure C.1: Bar chart showing how cross-entropy varies with $\hbar$ for the prediction of `Duration` in the alto, tenor and bass given soprano using the augmented `Pitch` domain, comparing version 1 with version 3.

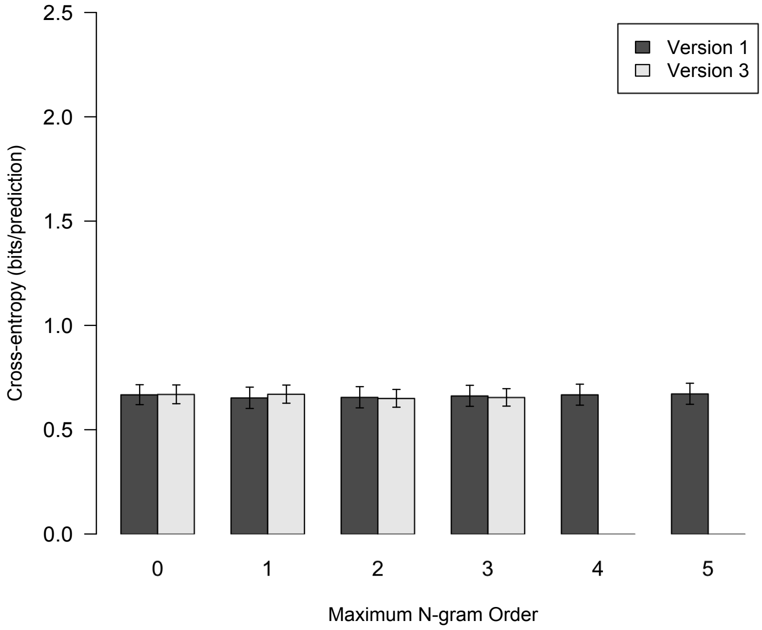


Figure C.2: Bar chart showing how cross-entropy varies with $\hbar$ for the prediction of `Cont` in the alto, tenor and bass given soprano using the augmented `Pitch` domain, comparing version 1 with version 3.

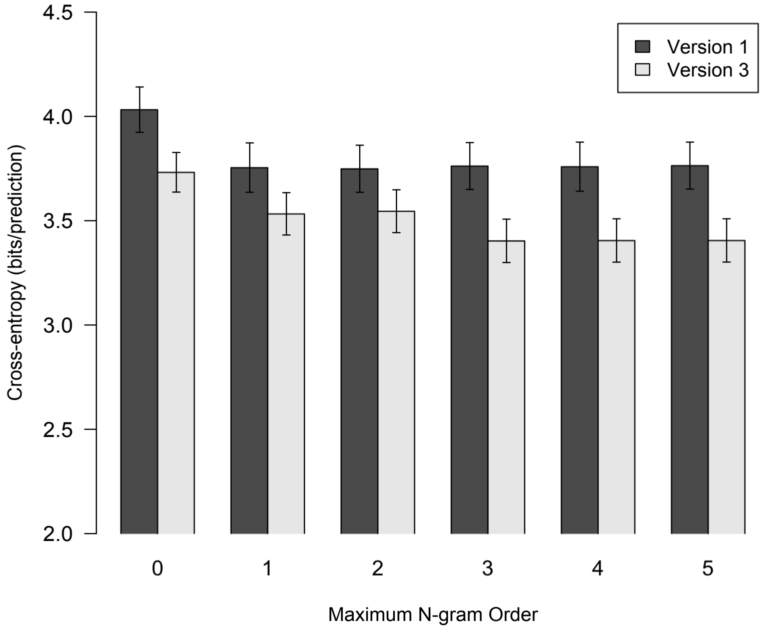


Figure C.3: Bar chart showing how cross-entropy varies with $\hbar$ for the prediction of `Pitch` in the alto, tenor and bass given soprano using the augmented `Pitch` domain, comparing version 1 with version 3.

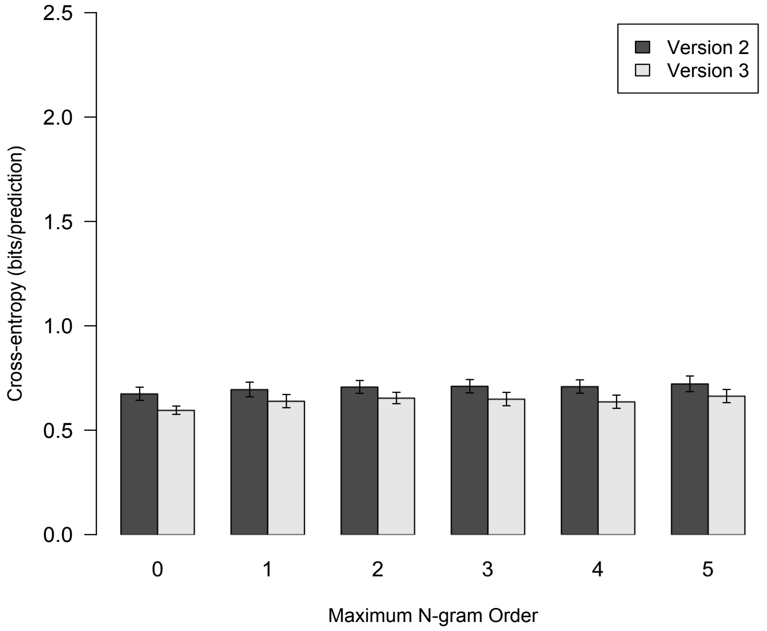


Figure C.4: Bar chart showing how cross-entropy varies with $\hbar$ for the prediction of `Duration` in the bass given soprano using the augmented `Pitch` domain, comparing versions 2 and 3.

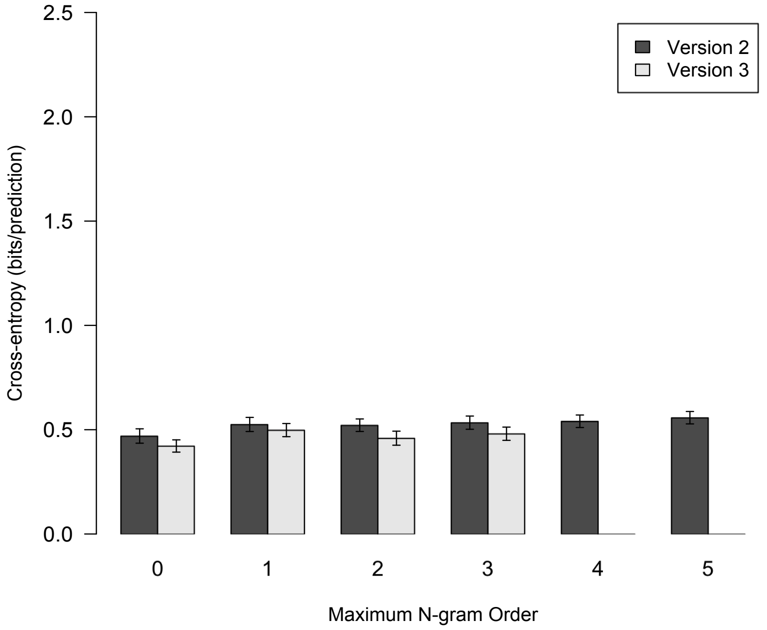


Figure C.5: Bar chart showing how cross-entropy varies with $\hbar$ for the prediction of `Duration` in the alto/tenor given soprano/bass using the augmented `Pitch` domain, comparing version 2 with version 3.

model is, however, the same as that of the version 2 model to two decimal places, at 0.26 bits/prediction for an $\hbar$ of 1 (optimised bias and L-S bias are 3.5 and 1.2 respectively). Moving on to the prediction of `Cont` in the alto/tenor given soprano/bass, Figure C.7 shows that version 2 is vastly superior. This is unexpected, considering the `Duration` results above. Bearing in mind that the set of possible version 2 systems is a subset of the set of possible version 3 systems, the fault is more likely to lie with the viewpoint selection algorithm (or the parameters used during viewpoint selection) than with the specification of the more flexible version 3. The optimal version 3 cross-entropy is 0.38 bits/prediction at an $\hbar$ of 2 (with an optimised bias and L-S bias of 2.2 and 140 respectively) compared with 0.28 bits/prediction for version 2.

We can see from Figure C.8 that version 2 is also a little better for the prediction of `Pitch` in the bass given soprano. Here, the lowest cross-entropy for the version 3 model is 1.76 bits/prediction at an $\hbar$ of 3 (optimised bias and L-S bias are 2.0 and 42.8 respectively) compared with 1.70 bits/prediction for the version 2 model. With the prediction of `Pitch` in the alto/tenor given soprano/bass, we switch to version 3 being overwhelmingly superior; see Figure C.9. It would appear that the additional flexibility of version 3 has been utilised to good effect in this case, resulting in an optimal cross-entropy of 1.66 bits/prediction at an $\hbar$ of 1 (with an optimised bias and L-S bias of 1.3 and 76.7 respectively) compared with 1.96 bits/prediction for version 2.

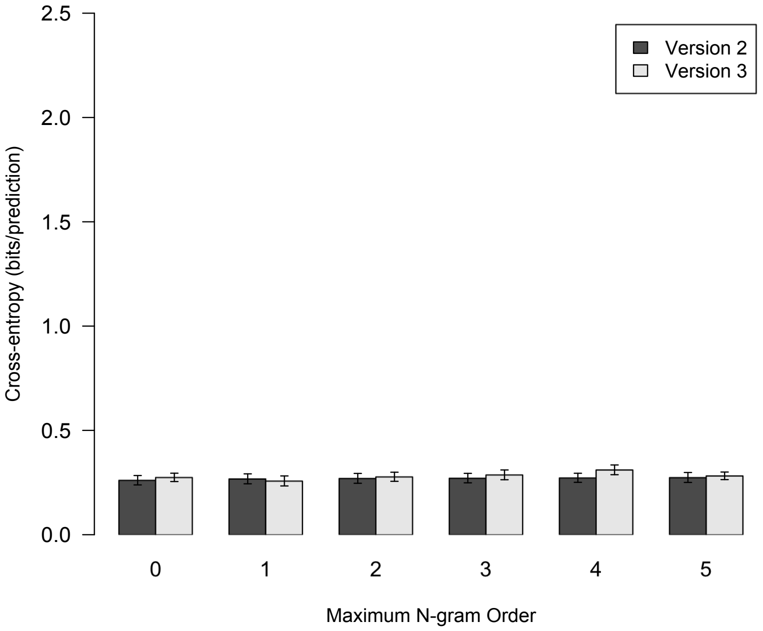


Figure C.6: Bar chart showing how cross-entropy varies with $\hbar$ for the prediction of `Cont` in the bass given soprano using the augmented `Pitch` domain, comparing versions 2 and 3.

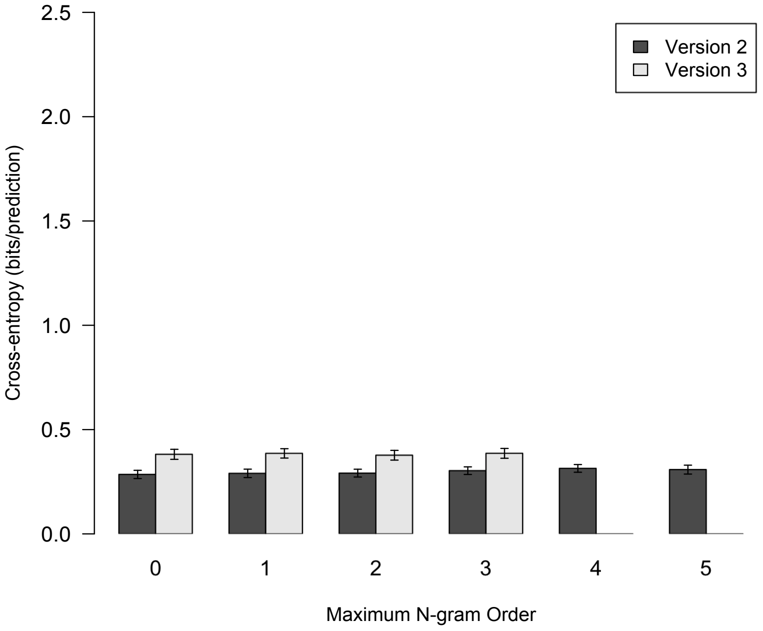


Figure C.7: Bar chart showing how cross-entropy varies with $\hbar$ for the prediction of `Cont` in the alto/tenor given soprano/bass using the augmented `Pitch` domain, comparing version 2 with version 3.

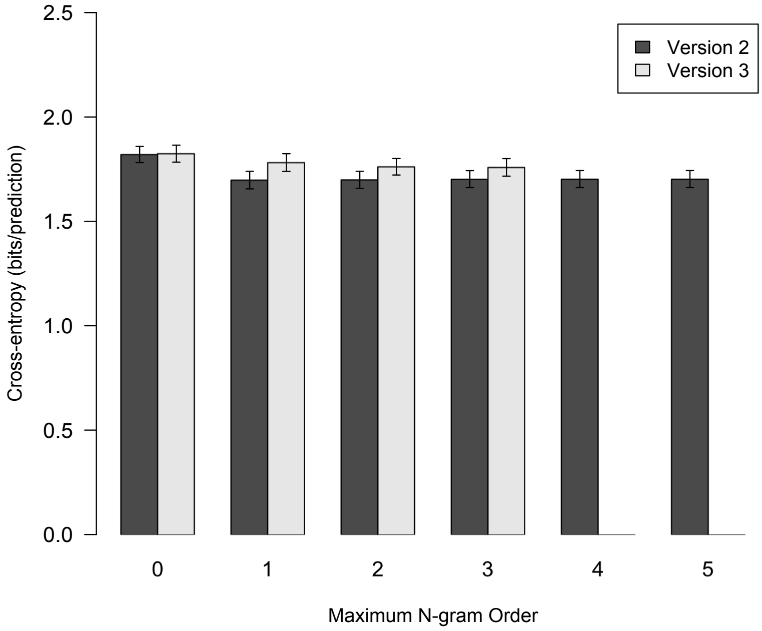


Figure C.8: Bar chart showing how cross-entropy varies with $\hbar$ for the prediction of `Pitch` in the bass given soprano using the augmented `Pitch` domain, comparing versions 2 and 3.

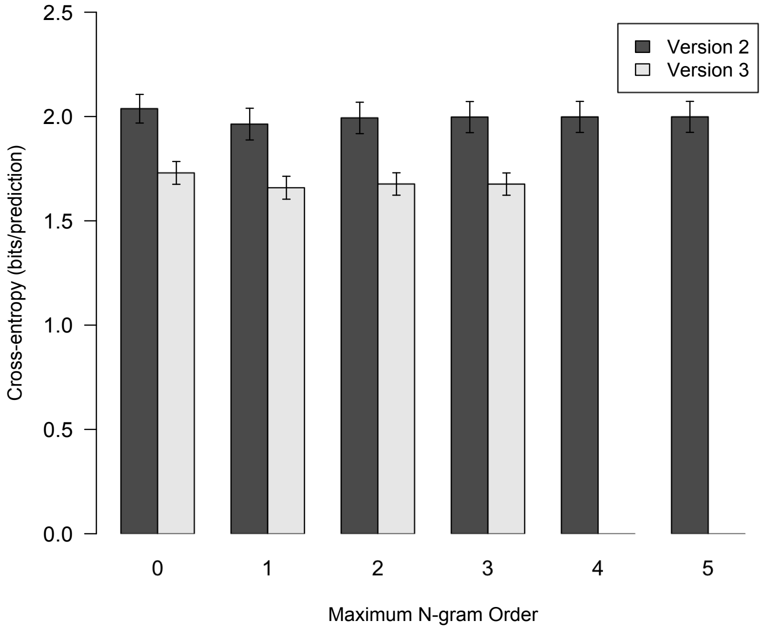


Figure C.9: Bar chart showing how cross-entropy varies with $\hbar$ for the prediction of `Pitch` in the alto/tenor given soprano/bass using the augmented `Pitch` domain, comparing version 2 with version 3.

Appendix D

Test Data Scores

Scores of the melodies and their harmonisations referred to in Chapters 7 and 8 are included here for convenient reference. The first five are test data ‘A’ and the final one belongs to test data ‘B’. Time signatures (not given in the hymnal), slurs, fermatas and non-terminal double bar lines are omitted.

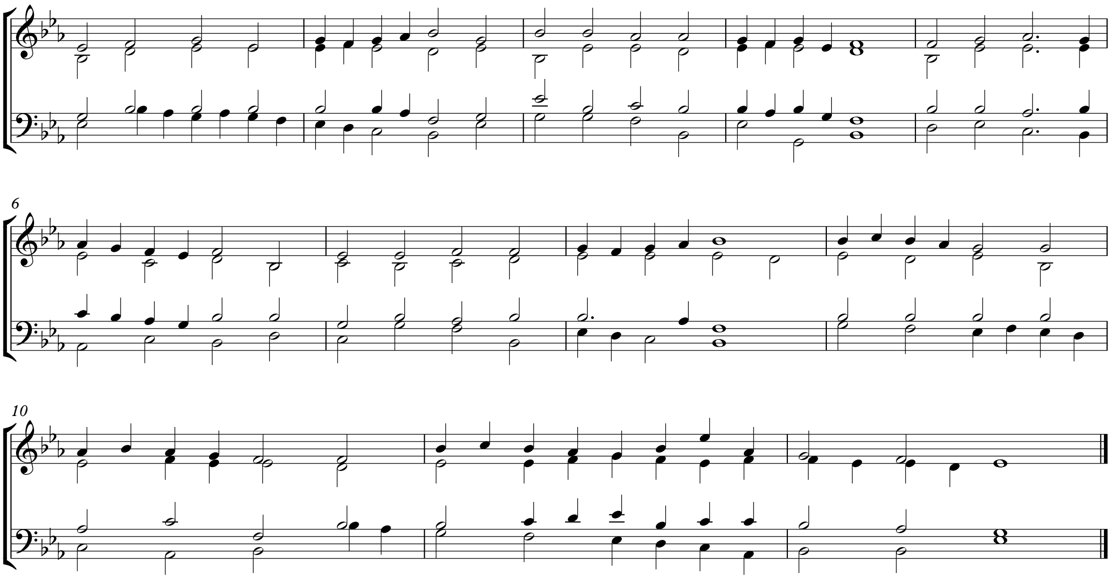


Figure D.1: Hymnal harmonisation of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33).

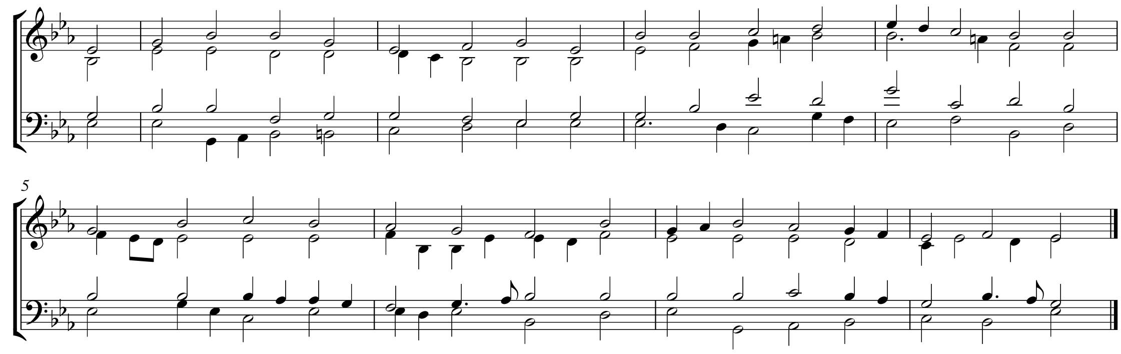


Figure D.2: Hymnal harmonisation of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36).

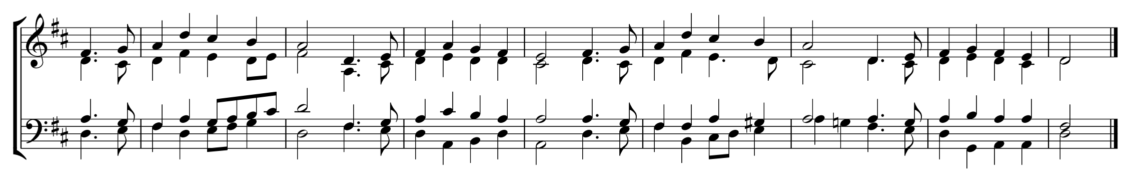


Figure D.3: Hymnal harmonisation of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37).

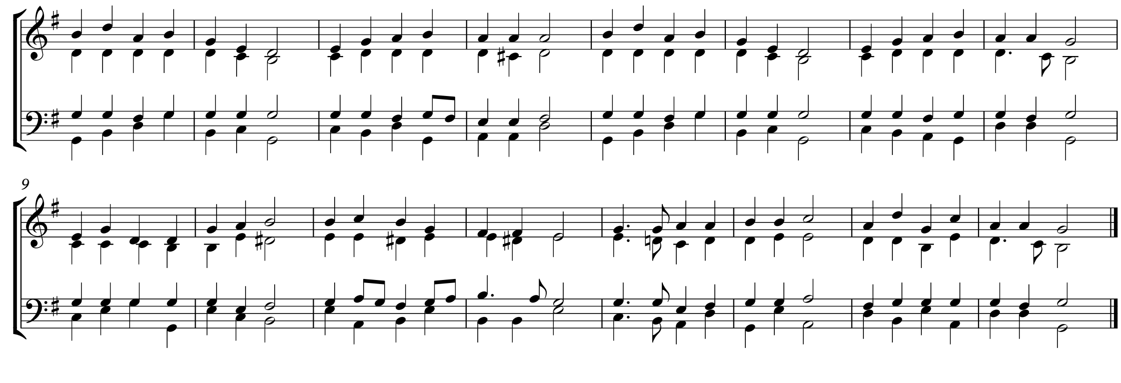


Figure D.4: Hymnal harmonisation of hymn tune *St. Edmund* (Vaughan Williams 1933, hymn no. 47).

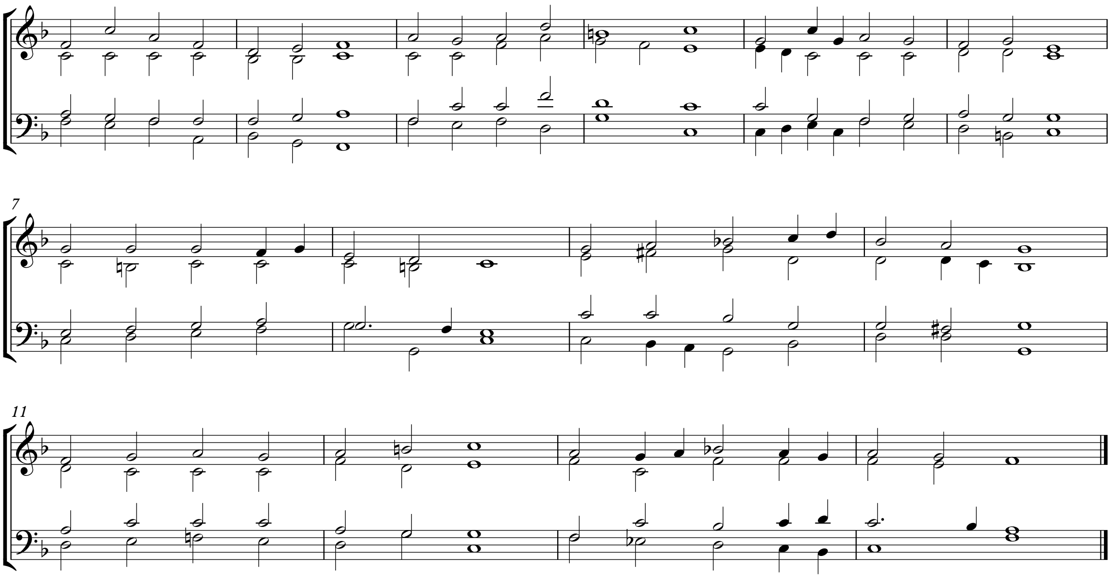


Figure D.5: Hymnal harmonisation of hymn tune *Das ist meine Freude* (Vaughan Williams 1933, hymn no. 97).

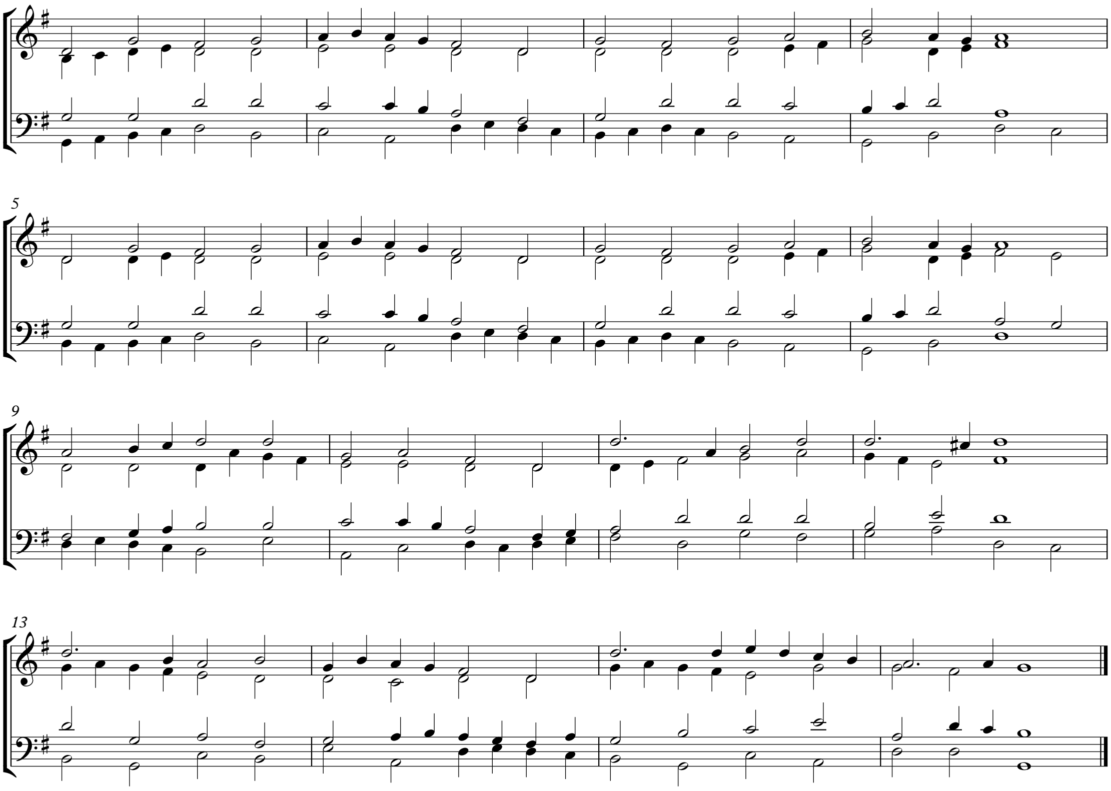


Figure D.6: Hymnal harmonisation of hymn tune *Würzburg* (Vaughan Williams 1933, hymn no. 127).

Appendix E

Mutual Information Between Viewpoints

Let us consider the extent to which there may be mutual information between different viewpoint models. Taking the cross-entropy resulting from the use of multiple viewpoint system {Duration, Pitch} as a reference point, the addition of ScaleDegree ⊗ Tessitura to the system reduces the cross-entropy by 0.49 bits/note; whereas adding Interval ⊗ LastInPhrase instead reduces it by 0.30 bits/note. Adding both of these viewpoints, however, reduces the cross-entropy by only 0.60 bits/note; therefore we can say that there is mutual information of 0.19 bits/note. Figure E.1 illustrates the mutual information between the first five viewpoints to be added to the best LTM system. Deletions complicate the matter, and have not been included in the diagram. The largest of the overlaps, 0.37 bits/note, is between ScaleDegree ⊗ Tessitura and ScaleDegree ⊗ Piece (perhaps not surprising since ScaleDegree is involved in both); of this, 0.15 bits/note is also common to Interval ⊗ LastInPhrase. Notice that although Interval ⊗ LastInPhrase causes a smaller cross-entropy reduction on its own than either of the other two Pitch predicting viewpoints, when all three viewpoints are used it has the largest information content not shared with either of the other viewpoints. There is no mutual information between the Duration and Pitch predicting viewpoints, since none of them can predict both attributes.

Now let us briefly consider deletion, taking Duration as an example. When Duration ⊗ Metre is added to {Duration, Pitch}, the cross-entropy is reduced by 0.21 bits/note. The deletion of Duration at this stage, however, would result in an increase of 0.03 bits/note (giving a total reduction of 0.18 bits/note). The further addition of DurRatio ⊗ Phrase is required before the deletion of Duration produces a fall in cross-entropy.

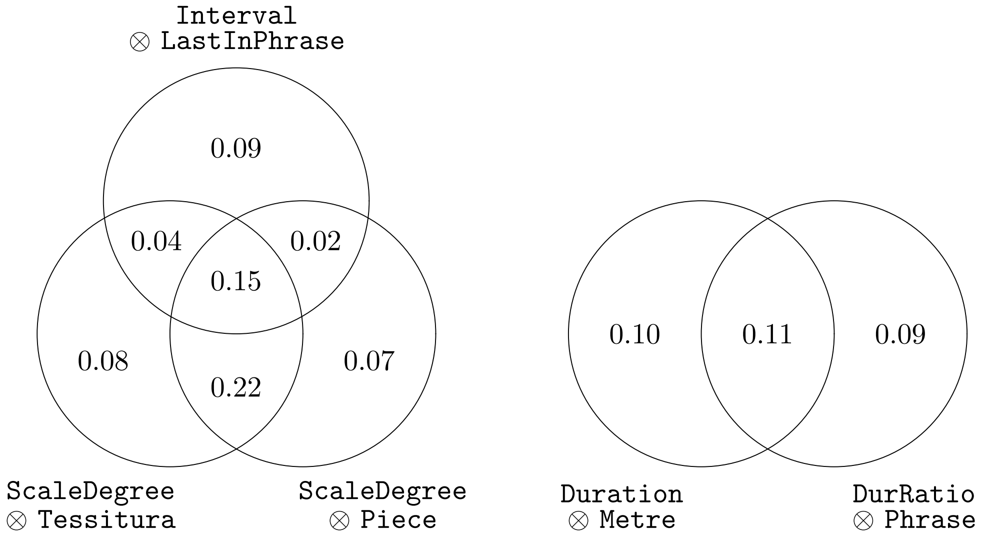


Figure E.1: Venn diagram illustrating the mutual information between viewpoint models, using an LTM with an $\hbar$ of 2, a bias of 1.4 and ten-fold cross-validation of corpus ‘A’. The figures indicate the reduction in cross-entropy (bits/note) resulting from the addition of these viewpoints to multiple viewpoint system {`Duration`, `Pitch`}.

Appendix F

Generation With Probability Thresholds of 1

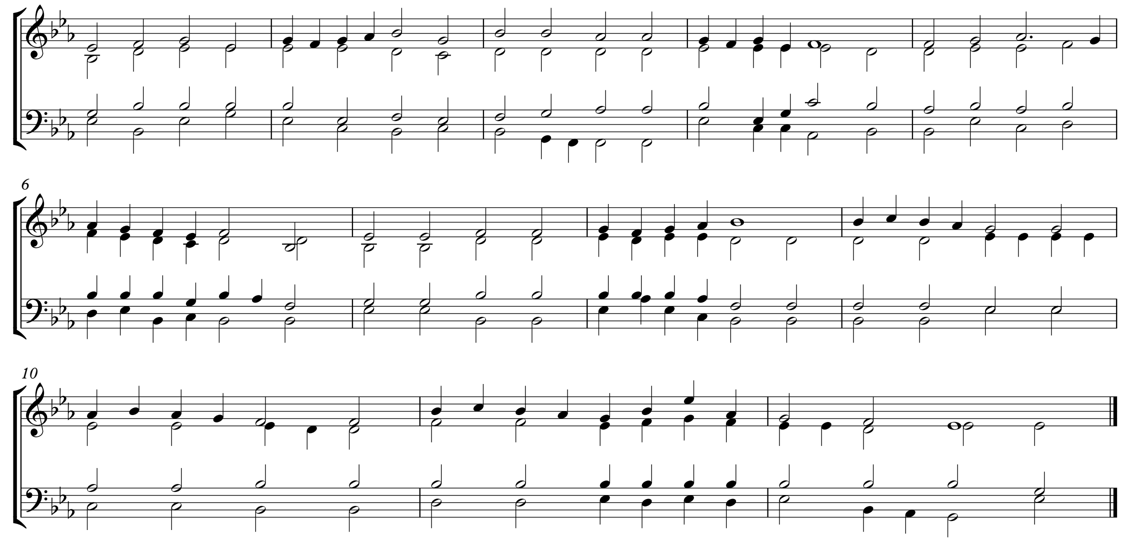


Figure F.1: Harmonisation of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33) automatically generated by the best version 1 model with all probability threshold parameters set to 1, using corpus ‘A+B’.

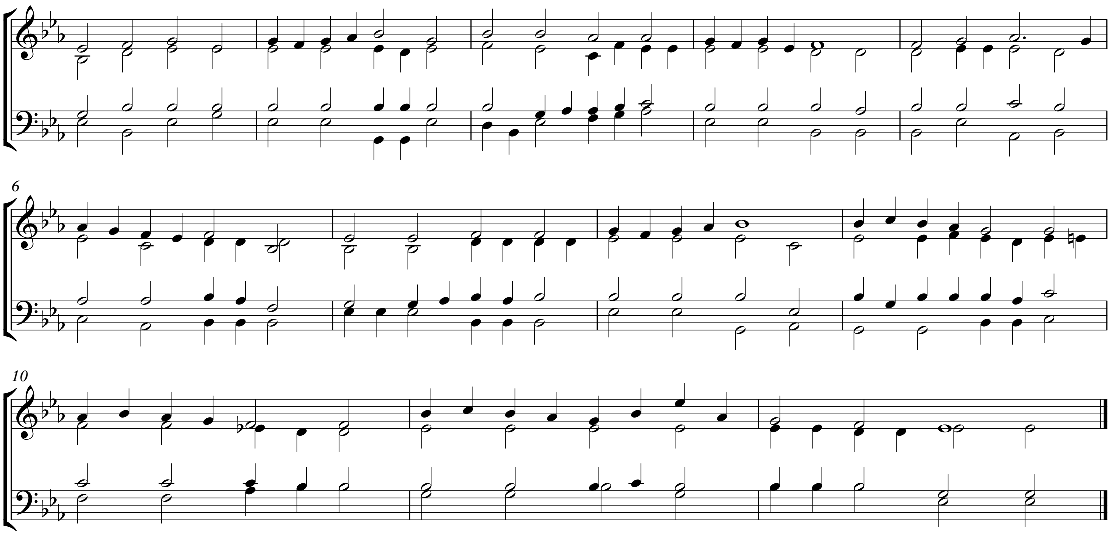


Figure F.2: Harmonisation of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33) automatically generated by the best version 2 model with all probability threshold parameters set to 1, using corpus 'A+B'.

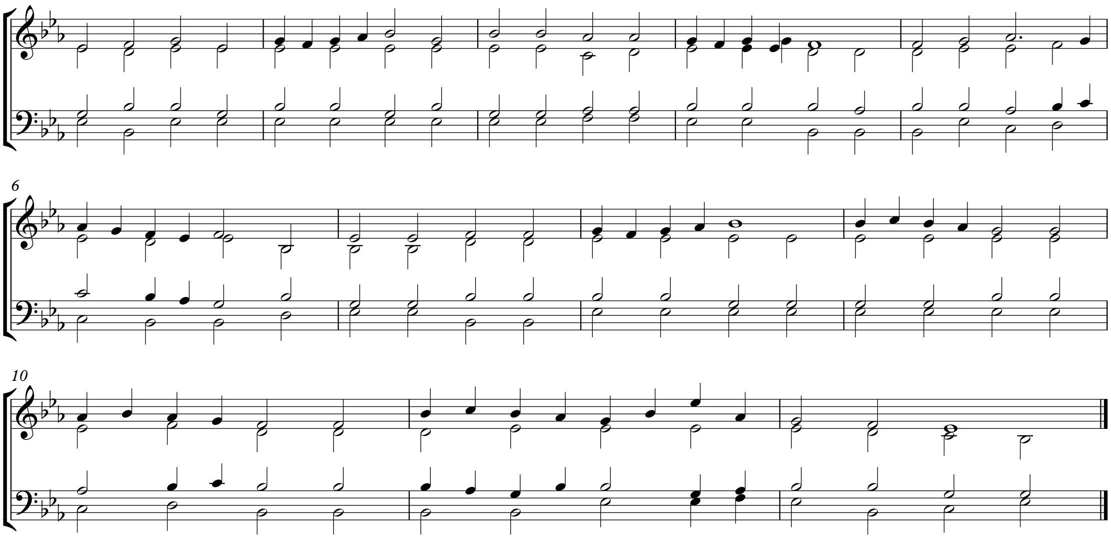


Figure F.3: Harmonisation of hymn tune *Grafton* (Vaughan Williams 1933, hymn no. 33) automatically generated by the best version 3 model with all probability threshold parameters set to 1, using corpus 'A+B'.

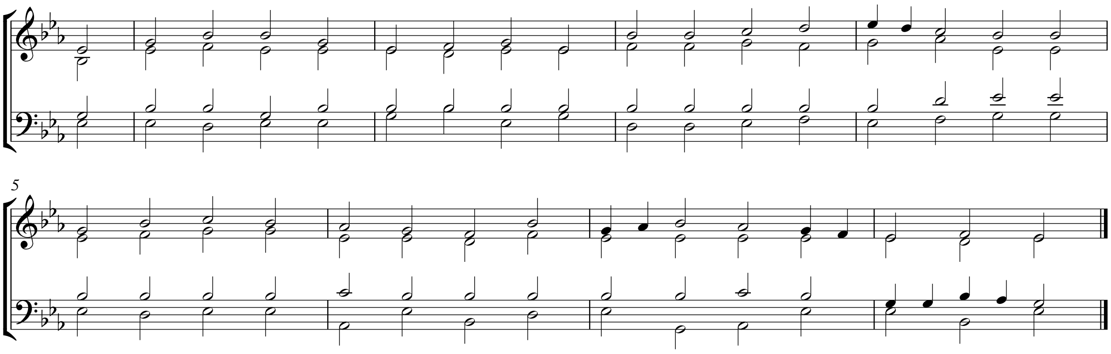


Figure F.4: Harmonisation of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36) automatically generated by the best version 1 model with all probability threshold parameters set to 1, using corpus 'A+B'.

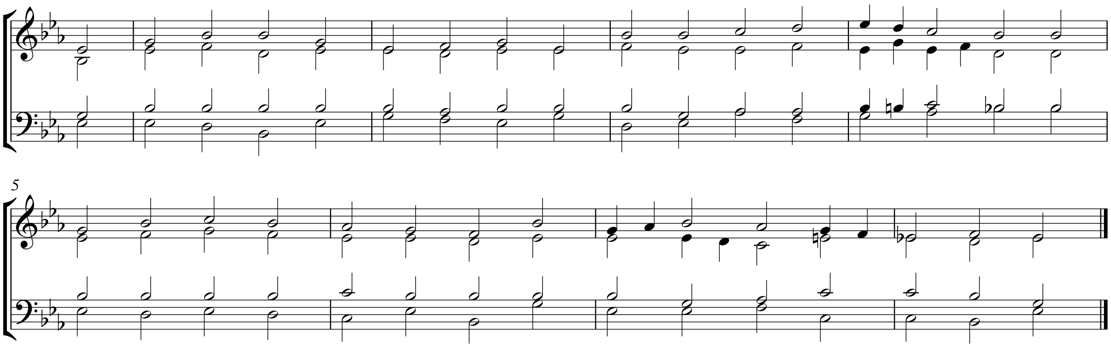


Figure F.5: Harmonisation of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36) automatically generated by the best version 2 model with all probability threshold parameters set to 1, using corpus 'A+B'.

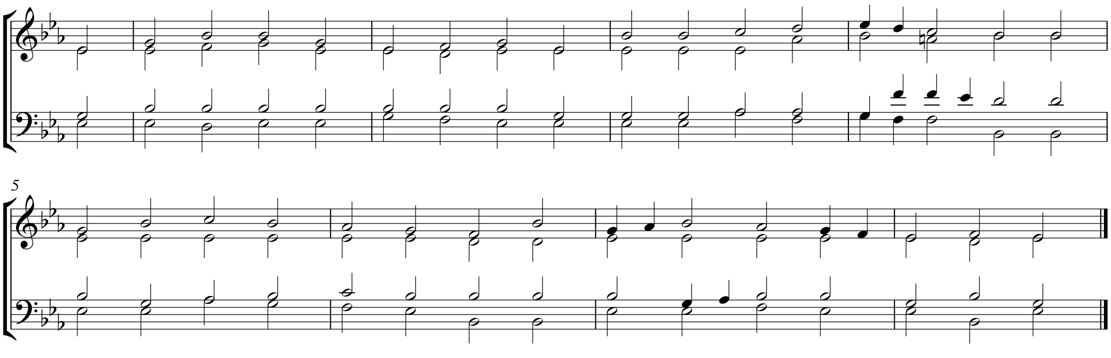


Figure F.6: Harmonisation of hymn tune *Das walt' Gott Vater* (Vaughan Williams 1933, hymn no. 36) automatically generated by the best version 3 model with all probability threshold parameters set to 1, using corpus 'A+B'.

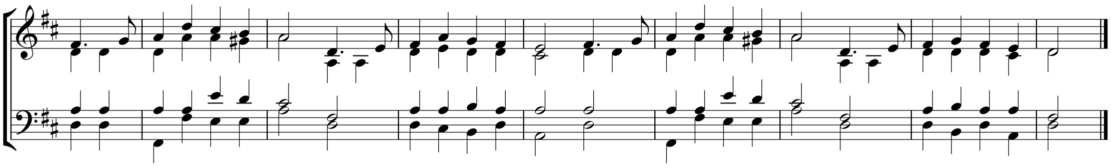


Figure F.7: Harmonisation of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37) automatically generated by the best version 1 model with all probability threshold parameters set to 1, using corpus ‘A+B’.

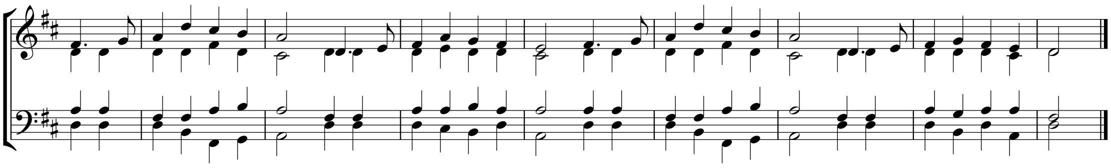


Figure F.8: Harmonisation of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37) automatically generated by the best version 2 model with all probability threshold parameters set to 1, using corpus ‘A+B’.

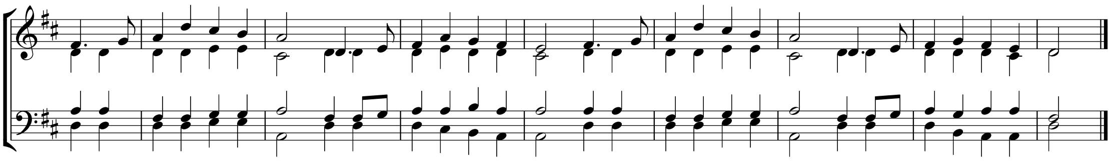


Figure F.9: Harmonisation of hymn tune *Innocents* (Vaughan Williams 1933, hymn no. 37) automatically generated by the best version 3 model with all probability threshold parameters set to 1, using corpus ‘A+B’.

Appendix G

Versions 8 and 9 As Graphical Models

Viewpoint models in versions 1 to 7 comprise a number of N-gram models within the PPM framework or some development of it. In versions 8 and 9, however, N-grams are no longer used; instead, more complex context/prediction configurations are employed which can best be understood as more general dynamic Bayesian network models (see §2.2.5.1). Examples are taken from some very preliminary work using only scale degrees and root position harmonic function symbols; the best performing of the 1st- to 5th-order probabilistic models appearing in the back-off sequence for the bass note prediction subtask (given soprano and harmonic function symbols) are represented here as dynamic Bayesian networks and (for comparison) Markov random fields. A different 5th-order model for this subtask, which incorporates future context, is also represented in the same way. The graphs cover an arbitrary six chords. The dynamic Bayesian network representation of the 1st-order model looks like a 0th-order N-gram model with one additional piece of context (alternatively, it could be seen as a 0th-order hidden Markov model). The Markov random field representation contains twelve disjoint maximal cliques; see Figure G.1. For sets of random variables **s**, **b** and **h** (soprano, bass and harmony respectively), the dynamic Bayesian network joint distribution is:

$$p(\mathbf{s}, \mathbf{b}, \mathbf{h}) = \prod_{n=1}^{6} p(s_n)p(h_n)p(b_n|h_n).$$

The 2nd-order dynamic Bayesian network representation looks like a 0th-order N-gram model with two additional pieces of context, while the Markov random field has six disjoint maximal cliques; see Figure G.2. The dynamic Bayesian network joint distribution is:

$$p(\mathbf{s}, \mathbf{b}, \mathbf{h}) = \prod_{n=1}^{6} p(s_n)p(h_n)p(b_n|s_n, h_n).$$

The 3rd-order dynamic Bayesian network looks like a 1st-order N-gram model with

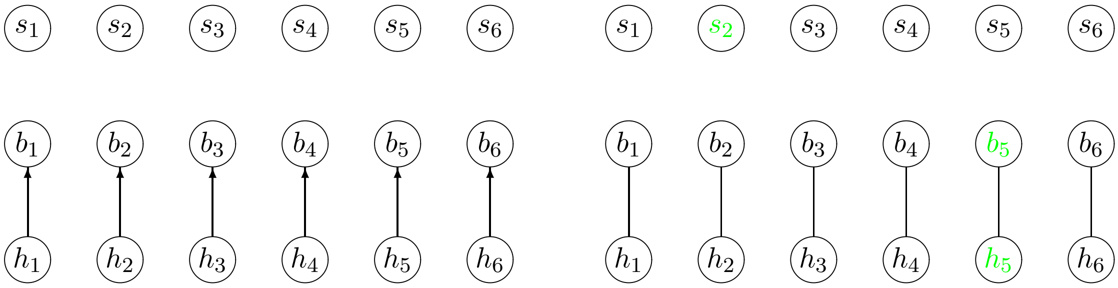

s_1 s_2 s_3 s_4 s_5 s_6
b_1 b_2 b_3 b_4 b_5 b_6
h_1 h_2 h_3 h_4 h_5 h_6

Figure G.1: Dynamic Bayesian network (left) and Markov random field representations of a 1st-order model. Example maximal cliques are shown in green.

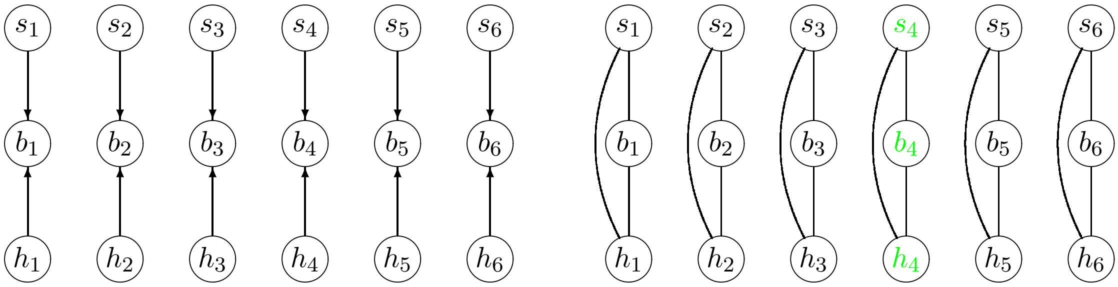

s_1 s_2 s_3 s_4 s_5 s_6
b_1 b_2 b_3 b_4 b_5 b_6
h_1 h_2 h_3 h_4 h_5 h_6

Figure G.2: Dynamic Bayesian network (left) and Markov random field representations of a 2nd-order model. An example maximal clique is shown in green.

two additional pieces of context, while the Markov random field has six overlapping maximal cliques; see Figure G.3. The dynamic Bayesian network joint distribution is:

$$p(\mathbf{s}, \mathbf{b}, \mathbf{h}) = p(s_1)p(h_1)p(b_1|s_1, h_1) \prod_{n=2}^{6} p(s_n)p(h_n)p(b_n|b_{n-1}, s_n, h_n).$$

The 4th-order dynamic Bayesian network looks like a 1st-order N-gram model with three additional pieces of context, while the Markov random field has six overlapping

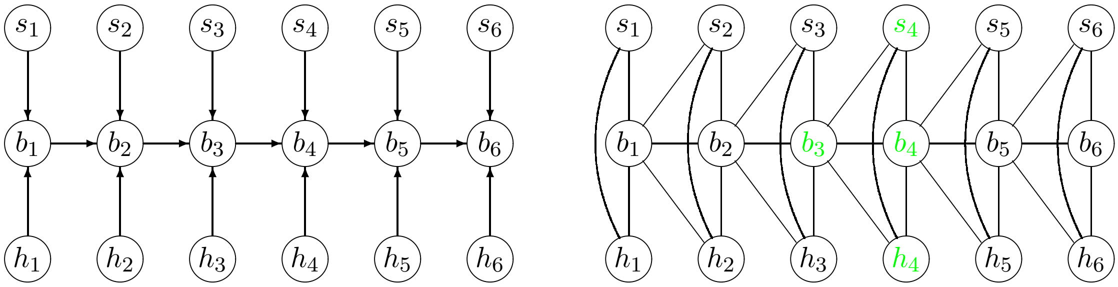

s_1 s_2 s_3 s_4 s_5 s_6
b_1 b_2 b_3 b_4 b_5 b_6
h_1 h_2 h_3 h_4 h_5 h_6

Figure G.3: Dynamic Bayesian network (left) and Markov random field representations of a 3rd-order model. An example maximal clique is shown in green.

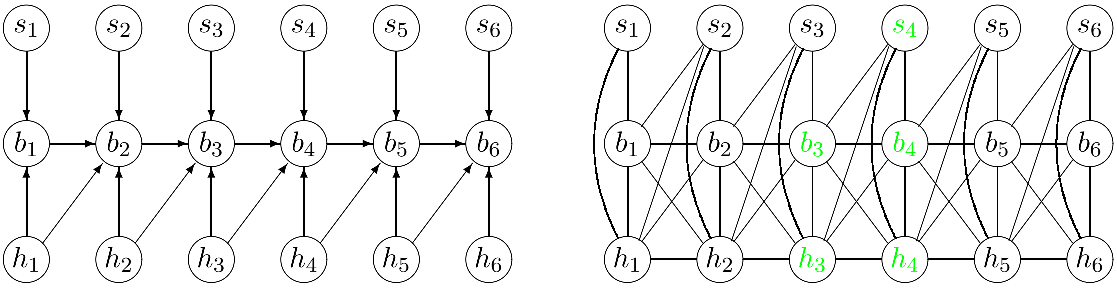


Figure G.4: Dynamic Bayesian network (left) and Markov random field representations of a 4th-order model. An example maximal clique is shown in green.

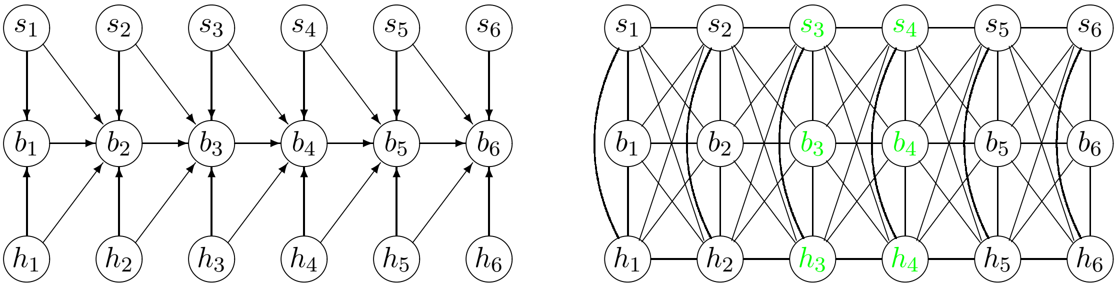


Figure G.5: Dynamic Bayesian network (left) and Markov random field representations of a 5th-order model. An example maximal clique is shown in green.

maximal cliques; see Figure G.4. The dynamic Bayesian network joint distribution is:

$$p(\mathbf{s}, \mathbf{b}, \mathbf{h}) = p(s_1)p(h_1)p(b_1|s_1, h_1) \prod_{n=2}^{6} p(s_n)p(h_n)p(b_n|b_{n-1}, h_{n-1}, s_n, h_n).$$

The 5th-order dynamic Bayesian network looks like a 1st-order N-gram model with four additional pieces of context, while the Markov random field has five overlapping maximal cliques; see Figure G.5. The dynamic Bayesian network joint distribution is:

$$p(\mathbf{s}, \mathbf{b}, \mathbf{h}) = p(s_1)p(h_1)p(b_1|s_1, h_1) \prod_{n=2}^{6} p(s_n)p(h_n)p(b_n|b_{n-1}, s_{n-1}, h_{n-1}, s_n, h_n).$$

For an example containing future context, the second-best of the 5th-order bass note prediction models is shown in Figure G.6. The dynamic Bayesian network looks like a 1st-order N-gram model with four additional pieces of context, while the Markov random field has six overlapping maximal cliques. The dynamic Bayesian network joint distribution is:

$$p(\mathbf{s}, \mathbf{b}, \mathbf{h}) = p(b_1|s_1, h_1, s_2)p(b_6|b_5, h_5, s_6, h_6)\prod_{n=2}^{5}p(b_n|b_{n-1}, h_{n-1}, s_n, h_n, s_{n+1})\prod_{n=1}^{6}p(s_n)p(h_n).$$

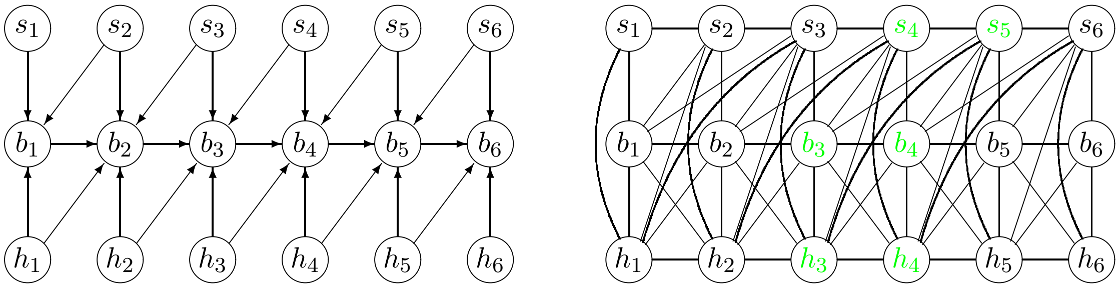

s_1 s_2 s_3 s_4 s_5 s_6
b_1 b_2 b_3 b_4 b_5 b_6
h_1 h_2 h_3 h_4 h_5 h_6
s_1 s_2 s_3 s_4 s_5 s_6
b_1 b_2 b_3 b_4 b_5 b_6
h_1 h_2 h_3 h_4 h_5 h_6

Figure G.6: Dynamic Bayesian network (left) and Markov random field representations of a different 5th-order model, incorporating future context. An example maximal clique is shown in green.

It should be noted that although the graphical representations are an excellent way of visualising the complex models developed here, there are no immediate plans to employ graphical model algorithms in future work. For the foreseeable future, it is intended that the statistical models will continue to be constructed and used in much the same way as in the current research, for purposes of comparison. The use of graphical model algorithms remains an option for work further into the future.

Bibliography

Aha, D. W. and Bankert, R. L. (1996). A comparative evaluation of sequential feature selection algorithms. In D. Fisher and H. J. Lenz, editors, *Learning from Data: AI and Statistics V*, pages 199–206. Springer, New York.

Allan, M. (2002). Harmonising chorales in the style of Johann Sebastian Bach. Master's thesis, School of Informatics, University of Edinburgh.

Allan, M. and Williams, C. K. I. (2005). Harmonising chorales by probabilistic inference. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 17. MIT Press.

Amabile, T. M. (1996). *Creativity in Context.* Westview Press, Boulder, Colorado.

Andreae, J. (1977). *Thinking with the Teachable Machine.* Academic Press.

Assayag, G., Dubnov, S., and Delerue, O. (1999). Guessing the composer's mind: Applying universal prediction to musical style. In *Proceedings of the 1999 International Computer Music Conference*, pages 496–499, San Fransisco. ICMA.

Bach, J. S. (1938). *St. Matthew Passion.* Novello Publishing Limited, London.

Bach, J. S. (1998). Chorale harmonisations in machine-readable (MIDI and text) files. http://i11www.ira.uka.de/ftp/pub/neuro/dominik/midifiles/bach.zip.

Bach, J. S. (2000). Chorale harmonisations in the **kern format. http://www.musedata.org/encodings/bach/bg/chorals/.

Baum, L. E. and Petrie, T. (1966). Statistical inference for probabilistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, 37.

Bellgard, M. I. and Tsang, C. P. (1994). Harmonizing music the boltzmann way. *Connection Science*, 6(2–3):281–297.

Bergeron, M. and Conklin, D. (2011). Subsumption of vertical viewpoint patterns. In C. Agon, M. Andreatta, G. Assayag, E. Amiot, J. Bresson, and J. Mandereau, editors, *Mathematics and Computation in Music: Third International Conference, MCM 2011, Paris, France, June 2011, Proceedings*, volume 6726 of *Lecture Notes in Artificial Intelligence*, pages 1–12, Berlin Heidelberg. Springer-Verlag.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning.* Springer Science+Business Media, LLC.

Biyikoğlu, K. M. (2003). A Markov model for chorale harmonization. In R. Kopiez, A. C. Lehmann, I. Wolther, and C. Wolf, editors, *Proceedings of the 5th Triennial European Society for the Cognitive Sciences of Music (ESCOM) Conference*, pages 81–84.

Boden, M. A. (2004). *The Creative Mind.* Routledge, second edition.

Brooks Jr., F. P., Hopkins Jr., A. L., Neumann, P. G., and Wright, W. V. (1993). An experiment in musical composition. In S. M. Schwanauer and D. A. Levitt, editors, *Machine Models of Music*, pages 23–40. MIT Press.

Bundy, A. (1990). What kind of field is AI? In D. Partridge and Y. Wilks, editors, *The Foundations of Artificial Intelligence*, pages 215–222. Cambridge University Press.

Bundy, A. (1994). What is the difference between real creativity and mere novelty? *Behavioural and Brain Sciences*, 17(3):533–534.

Bunton, S. (1997). Semantically motivated improvements for PPM variants. *The Computer Journal*, 40(2/3):76–93.

Cemgil, A. T. (2006). Bayesian methods for music signal analysis. ISMIR Tutorial, available at `http://www-sigproc.eng.cam.ac.uk/~atc27/papers/cemgil-ismir-tutorial.pdf`.

Chen, S. F. and Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13(4):359–394.

Cleary, J. G. and Teahan, W. J. (1997). Unbounded length contexts for PPM. *The Computer Journal*, 40(2/3):67–75.

Cleary, J. G. and Witten, I. H. (1984). Data compression using adaptive coding and partial string matching. *IEEE Trans Communications*, COM-32(4):396–402.

Clement, B. J. (1998). Learning harmonic progression using Markov models. EECS545 Project, University of Michigan.

Conklin, D. (1990). Prediction and entropy of music. Master's thesis, Department of Computer Science, University of Calgary, Canada.

Conklin, D. (2002). Representation and discovery of vertical patterns in music. In C. Anagnostopoulou, M. Ferrand, and A. Smaill, editors, *Music and Artificial Intelligence: Proc. ICMAI 2002, LNAI 2445*, pages 32–42. Springer-Verlag.

Conklin, D. (2003). Music generation from statistical models. In *Proceedings of the AISB 2003 Symposium on Artificial Intelligence and Creativity in the Arts and Sciences*, pages 30–35, Aberystwyth, Wales.

Conklin, D. and Anagnostopoulou, C. (2001). Representation and discovery of multiple viewpoint patterns. In *Proceedings of the International Computer Music Conference*, pages 479–485. International Computer Music Association.

Conklin, D. and Cleary, J. G. (1988). Modelling and generating music using multiple viewpoints. In *Proceedings of the First Workshop on AI and Music*, pages 125–137. The American Association for Artificial Intelligence.

Conklin, D. and Witten, I. H. (1995). Multiple viewpoint systems for music prediction. *Journal of New Music Research*, 24(1):51–73.

Cope, D. (2001). *Virtual Music.* MIT Press.

Corman, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2001). *Introduction to Algorithms.* MIT Press, second edition.

Cover, T. M. and King, R. C. (1978). A convergent gambling estimate of the entropy of English. *IEEE Transactions on Information Theory*, 24(4):413–421.

Cuddy, L. L. and Lunny, C. A. (1995). Expectancies generated by melodic intervals: Perceptual judgements of continuity. *Perception and Psychophysics*, 57(4):451–462.

Della Pietra, S., Della Pietra, V., and Lafferty, J. (1997). Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38.

Dixon, S. (2000). A lightweight multi-agent musical beat tracking system. In *Proceedings of the Pacific Rim International Conference on Artificial Intelligence (PRICAI)*, pages 778–788, Melbourne, Australia.

Dixon, S. and Cambouropoulos, E. (2000). Beat tracking with musical knowledge. In W. Horn, editor, *Proceedings of the 14th Biennial European Conference on Artificial Intelligence (ECAI)*, pages 626–630, Amsterdam. IOS Press.

Ebcioğlu, K. (1988). An expert system for harmonizing four-part chorales. *Computer Music Journal*, 12(3):43–51.

Fine, S., Singer, Y., and Tishby, N. (1998). The hierarchical hidden Markov model: Analysis and applications. *Machine Learning*, 32(1):41–62.

Friedman, N. and Koller, D. (2003). Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50:95–126.

Garvey, T. D., Lowrance, J. D., and Fischler, M. A. (1981). An inference technique for integrating knowledge from disparate sources. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, pages 319–325, Vancouver, Canada.

Hamburger, H. (1986). Representing, combining and using uncertain estimates. In L. N. Kanal and J. F. Lemmer, editors, *Uncertainty in Artificial Intelligence*, pages 399–414. North-Holland.

Hild, H., Feulner, J., and Menzel, W. (1992). Harmonet: A neural net for harmonizing chorales in the style of J. S. Bach. In R. P. Lippmann, J. E. Moody, and D. S. Touretzky, editors, *Advances in Neural Information Processing Systems*, volume 4, pages 267–274. Morgan Kaufmann.

Hinton, G. E. and Sejnowski, T. J. (1986). Learning and relearning in boltzmann machines. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing*, volume 1. MIT Press, Cambridge, MA.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems.* The University of Michigan Press.

Hörnel, D. and Ragg, T. (1996). A connectionist model for the evolution of styles of harmonization. In *Proceedings of the 1996 International Conference on Music Perception and Cognition*, Montreal, Canada.

Huron, D. (1997). Humdrum and kern: Selective feature encoding. In E. Selfridge-Field, editor, *Beyond MIDI: The Handbook of Musical Codes*, pages 375–401. MIT Press, Cambridge, MA.

Katz, S. M. (1987). Estimation of probabilities from sparse data for the language model component of a speech recogniser. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 35(3):400–401.

Kirkpatrick, S., Gelatt Jr., C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220:671–680.

Kneser, R. and Ney, H. (1995). Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 181–184.

Krumhansl, C. L. (1995). Effects of musical context on similarity and expectancy. *Systematische Musikwissenschaft*, 3(2):211–250.

Krumhansl, C. L. and Kessler, E. J. (1982). Tracing the dynamic changes in perceived tonal organisation in a spatial representation of musical keys. *Psychological Review*, 89(4):334–368.

Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86.

Lerdahl, F. and Jackendoff, R. (1983). *A Generative Theory of Tonal Music*. MIT Press.

Mackworth, A. K. (1977). Consistency in networks of relations. *Artificial Intelligence*, 8:99–118.

Mann, A., editor (1965). *The Study of Counterpoint from Johann Joseph Fux's Gradus Ad Parnassum*. W. W. Norton, New York.

Manning, C. D. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press.

Manzara, L. C., Witten, I. H., and James, M. (1992). On the entropy of music: An experiment with Bach chorale melodies. *Leonardo*, 2(1):81–88.

Marr, D. (1982). *Vision*. W. H. Freeman, San Fransisco.

McClamrock, R. (1991). Marr's three levels: a re-evaluation. *Minds and Machines*, 1:185–196.

Meredith, D. (1996). *The Logical Structure of an Algorithmic Theory of Tonal Music*. PhD thesis, Faculty of Music, University of Oxford.

Moore, B. (1982). *An Introduction to the Psychology of Hearing*. Academic Press.

Murphy, K. P. (2002). *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, University of California, Berkeley.

Narmour, E. (1990). *The Analysis and Cognition of Basic Melodic Structures: The Implication-realization Model*. University of Chicago Press.

Narmour, E. (1992). *The Analysis and Cognition of Melodic Complexity: The Implication-realization Model*. University of Chicago Press.

Nicholson, S., Knight, G. H., and Dykes Bower, J., editors (1950). *Hymns Ancient & Modern Revised*. William Clowes and Sons, Ltd.

Ovans, R. and Davison, R. (1992). An interactive constraint-based expert assistant for music composition. In *Proceedings: Ninth Canadian Conference on Artificial Intelligence*, pages 76–81.

Pachet, F., Ramalho, G., and Carrive, J. (1996). Representing temporal musical objects and reasoning in the MusES system. *Journal of New Music Research*, 25(3):252–275.

Pachet, F. and Roy, P. (1998). Formulating contraint satisfaction problems on part-whole relations: The case of automatic harmonization. In *13th Biennial European Conference on Artificial Intelligence (ECAI) Workshop: Constraint techniques for artistic applications*, Brighton, UK.

Paiement, J.-F., Eck, D., and Bengio, S. (2005a). Chord representations for probabilistic models. Research Report IDIAP-RR 05-58, IDIAP Research Institute, Switzerland.

Paiement, J.-F., Eck, D., and Bengio, S. (2005b). A probabilistic model for chord progressions. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, pages 312–319.

Paiement, J.-F., Eck, D., and Bengio, S. (2006). Probabilistic melodic harmonization. In *Proceedings of the 19th Canadian Conference on Artificial Intelligence*. Springer.

Pearce, M. T. (2005). *The Construction and Evaluation of Statistical Models of Melodic Structure in Music Perception and Composition*. PhD thesis, Department of Computing, City University, London.

Pearce, M. T., Meredith, D., and Wiggins, G. A. (2002). Motivations and methodologies for automation of the compositional process. *Musicae Scientiae*, 6(2):119–147.

Pearce, M. T., Müllensiefen, D., and Wiggins, G. A. (2008). A comparison of statistical and rule-based models of melodic segmentation. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, pages 89–94, Philadelphia, USA. Drexel University.

Pearce, M. T. and Wiggins, G. A. (2001). Towards a framework for the evaluation of machine compositions. In *Proceedings of the AISB 2001 Symposium on Artificial Intelligence and Creativity in the Arts and Sciences*.

Pearce, M. T. and Wiggins, G. A. (2004). Improved methods for statistical modelling of monophonic music. *Journal of New Music Research*, 33(4):367–385.

Pearce, M. T. and Wiggins, G. A. (2006). Expectation in melody: The influence of context and learning. *Music Perception*, 23(5):377–405.

Phon-Amnuaisuk, S., Tuson, A., and Wiggins, G. A. (1999). Evolving musical harmonisation. In *Proceedings of the Fourth International Conference on Neural Networks and Genetic Algorithms (ICANNGA'99)*, Slovenia. Springer-Verlag.

Phon-Amnuaisuk, S. and Wiggins, G. A. (1999). The four-part harmonisation problem: A comparison between genetic algorithms and a rule-based system. In *Proceedings of the AISB'99 Symposium on Musical Creativity*, Edinburgh. AISB Symposium.

Pickens, J. and Iliopoulos, C. (2005). Markov random fields and maximum entropy modeling for music information retrieval. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, pages 207–214.

Pinkerton, R. C. (1956). Information theory and melody. *Scientific American*, 194(2):77–86.

Piston, W. (1976). *Harmony.* Victor Gollancz Ltd, London.

Ponsford, D., Wiggins, G. A., and Mellish, C. (1999). Statistical learning of harmonic movement. *Journal of New Music Research*, 28(2):150–177.

Potter, K., Wiggins, G. A., and Pearce, M. T. (2007). Towards greater objectivity in music theory: Information-dynamic analysis of minimalist music. *Musicae Scientiae*, 11(2):295–324.

Pressman, R. S. (2000). *Software Engineering A Practitioner's Approach.* McGraw-Hill Publishing Company, Maidenhead.

Raphael, C. and Stoddard, J. (2003). Harmonic analysis with probabilistic graphical models. In *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR)*, pages 177–181, Baltimore, USA.

Rothstein, J. (1992). *MIDI: A Comprehensive Introduction.* Oxford University Press.

Schellenberg, E. G. (1996). Expectancy in melody: Tests of the implication-realization model. *Cognition*, 58(1):75–125.

Schellenberg, E. G. (1997). Simplifying the implication-realization model of melodic expectancy. *Music Perception*, 14(3):295–318.

Schenker, H. (1979). *Free Composition (Der freie Satz).* Longman, New York.

Shanahan, M. (2005). Consciousness, emotion, and imagination. In *Proceedings of the Society for the Study of Artificial Intelligence and the Simulation of Behaviour (AISB) Workshop: Next Generation Approaches to Machine Consciousness*, pages 26–35.

Sloboda, J. (1985). *The Musical Mind: the Cognitive Psychology of Music.* Oxford Science Press.

Smaill, A., Wiggins, G. A., and Harris, M. (1993). Hierarchical music representation for composition and analysis. *Computers and the Humanities*, 27(1):7–17.

Smoliar, S. W. (1980). A computer aid for Schenkerian analysis. *Computer Music Journal*, 4(2):41–59.

Smyth, P. (1997). Belief networks, hidden Markov models, and Markov random fields: a unifying view. *Pattern Recognition Letters*, 18(11):1261–1268.

Tax, D. M. J., van Breukelen, M., Duin, R. P. W., and Kittler, J. (2000). Combining multiple classifiers by averaging or by multiplying? *Pattern Recognition*, 33(99):1475–1485.

Vassilakis, P. (1999). Chords as spectra, harmony as timbre. *Journal of the Acoustical Society of America*, 106(4/2):2286.

Vaughan Williams, R., editor (1933). *The English Hymnal*. Oxford University Press.

Viterbi, A. J. (1967). Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, 13:260–269.

Volk, A., Wiering, F., and van Kranenburg, P. (2011). Unfolding the potential of computational musicology. In *Proceedings of ICISO*, pages 137–144, the Netherlands.

Weiland, M., Smaill, A., and Nelson, P. (2005). Learning musical pitch structures with hierarchical hidden Markov models. In *Journées d'Informatique Musicale (conference)*, Paris.

Wierstra, D. (2004). A new implementation of hierarchical hidden Markov models. Master's thesis, Utrecht University.

Wiggins, G. A. (1998). The use of constraint systems for musical composition. In *Proceedings of the 13th Biennial European Conference on Artificial Intelligence (ECAI) Workshop: Constraint techniques for artistic applications*, Brighton, UK.

Wiggins, G. A. (2006a). A preliminary framework for description, analysis and comparison of creative systems. *Knowledge-Based Systems*, 19(7):449–458.

Wiggins, G. A. (2006b). Searching for computational creativity. *New Generation Computing*, 24(3):209–222.

Wiggins, G. A., Papadopoulos, G., Phon-Amnuaisuk, S., and Tuson, A. (1999). Evolutionary methods for musical composition. *International Journal of Computing Anticipatory Systems*, 4.

Witten, I. H. and Bell, T. C. (1989). The zero frequency problem: Estimating the probability of novel events in adaptive text compression. Technical Report 89/347/09, Department of Computer Science, The University of Calgary.

Witten, I. H., Manzara, L. C., and Conklin, D. (1994). Comparing human and computational models of music prediction. *Computer Music Journal*, 18(1):70–80.

Ziv, J. and Lempel, A. (1978). Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, 24(5):530–536.