Schenkerian Analysis by Computer: A Proof of Concept

This is a pre-print of an article to be published in *Journal of New Music Research*, vol.39, no.3. Details of the definitive version may be found at www.informaworld.com/jnmr.

Alan Marsden,
Lancaster Institute for the Contemporary Arts, Lancaster University, UK
A.Marsden@lancaster.ac.uk

Abstract

A system for automatically deriving a Schenkerian reduction of an extract of tonal music is described. Schenkerian theory is formalised in a quasi-grammatical manner, expressing a reduction as a binary-tree structure. Computer software which operates in the manner of a chart parser using this grammar has been implemented, capable of deriving a matrix of reduction possibilities, in polynomial time, from a representation of the score. A full reduction of the extract can be discovered by selecting a tree from this matrix. The number of possible valid reductions for even short extracts is found to be extremely large, so criteria are required to distinguish good reductions from bad ones. To find such criteria, themes from five Mozart piano sonatas are analysed and samples of 'good' reductions (defined by reference to pre-existing analyses of these themes) are compared with randomly sampled reductions. Nine criteria are thereby derived, which can be applied in the process of parsing and selecting a reduction. The results are promising, but the process is still too computationally expensive—only extracts of a few bars in length can be reduced—and more extensive testing is required before the system can be properly claimed to perform automatic Schenkerian analysis.

1. Introduction

1.1 Rationale

Since the first flurry of interest in formal grammars and their potential application to music, beginning in the 1960s and most famously manifested in Bernstein's Norton Lectures of 1973 (Bernstein, 1976), some have looked to Schenkerian theory as providing a kind of grammar of music and have sought to implement it in a computer program after the fashion of computational linguistics. A decade or so later, Lerdahl and Jackendoff presented a similarly reductional theory explicitly grounded in generative linguistics (Lerdahl & Jackendoff, 1983), which has also attracted attempts at computational implementation, most recently and most successfully in the work of Hamanaka, Hirata & Tojo (2006, 2007).

The potential benefits of a program to derive Schenkerian analyses or reduction trees in the style of Lerdahl and Jackendoff would be enormous. Such an analysis adumbrates much of the information carried in other kinds of structural analyses. For example, though few harmonic functions are explicitly indicated in a Schenkerian analysis, it is a simple matter to read at least the main harmonies from a graph. Segmentations are also inherent in the structure of a reduction, represented in Schenker's graphs principally by the use of different sizes of notes, beams and slurs, or in analyses in the style of Lerdahl & Jackendoff in the branching of the tree structure. A system which derived Schenkerian analyses or reduction trees would therefore also deliver a significant quantity of other useful information about a piece of music. (Further benefits of generative reductional representation of music are described in (Marsden, 2005). The benefits for the representation of pattern in particular are described in (Marsden, 2001).)

1.2 Main contribution

This paper gives a proof of concept of a mechanism for deriving Schenkerian analyses, and the kind of reductional representations described in (Marsden, 2005), from score data (more specifically, quantised MIDI-like data). The system described is too inefficient to provide an effective music-analysis tool—only short sections of simple music can be analysed. Furthermore, it is premature to claim that the analyses derived are 'correct' or useful. However, the work described here does provide grounds for optimism that further development might produce a tool which is both useful and reliable.

This work also provides a means for empirical investigation of the principles of Schenkerian analysis. Schenkerian theory (in common with many other music theories) is expressed essentially in the form of rules by which music may be elaborated, rather than describing either a mechanism by which a piece may be reduced or criteria by which one may choose between the various possible reductions which conform to the rules. There has been some discussion of what such criteria might be (Plum, 1988; Schachter, 1990), and guidance is given to students in making Schenkerian analyses (e.g., Pankhurst, 2008). However, there is no general agreement among music theorists about how a reduction should be made, nor any clear idea about how one would find out. The computational implementation described here allows the empirical investigation of both the adequacy of a set of rules and the criteria for choosing one analysis over another, and therefore promises advances in the theory of music analysis with a stronger empirical basis than heretofore.

1.3 Outline

Section 2 of this paper gives an essential summary of Schenkerian reduction and reviews related prior systematic work. Section 3 gives a formalisation of quasi-Schenkerian tonal reduction with sufficient precision for computational implementation. Section 4 discusses the problem of the explosion of the solution space and describes a reduction algorithm to cope with this problem. Section 5 describes exploratory empirical work, using short extracts of Mozart piano sonatas and pre-existing analyses of those extracts, to discover a 'goodness' metric by which to select a preferred reduction from among the many possibilities. Section 6 discusses the results with respect to this small sample, which show moderate success, and considers the prospects for further work.

As stated above, this is a proof of concept rather than a claim to have solved the problem of Schenkerian analysis by computer. The mechanism described here will not simply scale up to cover more than a few bars of music. While there are many published Schenkerian analyses which might be a source of test material, most cover entire pieces of music and small test examples are not readily available. Two substantial steps therefore remain to be taken before the problem of Schenkerian analysis by computer can be claimed to have been properly solved: thorough testing, and development of a mechanism which can be applied to realistic spans of music.

2. Systematic Reduction of Tonal Music

2.1 Schenkerian analysis

While reduction, in the broad sense of 'information reduction', is a part of any analysis, when applied to music the term generally has the quite specific meaning of deriving from a piece of music a structure of notes which contains the main structural outline of that music without its more 'ornamental' features. Thus the result of reducing a piece of music is itself a piece of music (at least in the sense that it is a structure of notes) which contains fewer notes than the original. The reduction process can then be applied recursively, producing ever simpler and simpler musical structures, until an irreducible basis is reached. Although the fundamental idea of such reduction is found in many places in music theory, it takes its most developed form in Schenkerian analysis, a technique of music analysis which has its origins in the publications and teaching of Heinrich Schenker (expressed most succinctly in *Der Freie Satz* (Schenker, 1935)). In the English-speaking world, Schenker's ideas were spread principally by his pupils who emigrated to the USA. By the 1970s Schenkerian theory had become the orthodoxy of tonal music taught at American universities. (A recent manifestation of this is (Cadwallader, 2007).) While powerful opposing theories have been proposed, there can be no denying the significant influence of Schenkerian analysis, and the value that a large body of music scholars has placed on it.

There is much more to Schenkerian analysis than just the removal of non-essential notes. Furthermore, Schenker himself did not aim to present a scientifically systematic theory of musical reduction (though he does make reference to musical 'laws'). The work described here is thus not a 'computerisation' of Schenkerian theory *in toto*, but rather an attempt to forge a systematic basis for the kind of reduction which is at the core of that theory.

2.2 Lerdahl & Jackendoff

As mentioned above, Lerdahl and Jackendoff (1983) have proposed a theory which also makes use of the same essential concept of reduction. There are important points of contact with Schenkerian analysis, but also four significant differences.

1. Lerdahl & Jackendoff express their theory in a much more explicitly systematic fashion, presenting it as a set of rules which generate well defined structures. In the case of reductions these structures are explicitly trees.
2. Their reductions are based principally on melody-plus-chord rather than on the full contrapuntal texture, as in Schenker's reductions.
3. Derivation of their reductions depends on grouping and metrical structures which are reductional in a different sense.

4. They identify two different kinds of reductional structure, ‘time-span’ and ‘prolongational’, which are, essentially, the result of bottom-up and top-down derivation respectively.

As will be seen below, I use the concept of binary trees found in the theory of Lerdahl & Jackendoff, but I regard the other three significant differences as disadvantages in comparison with the conceptual simplicity of Schenkerian analysis which finds just one kind of structure based on one kind of information in a piece of music. Some of the more general arguments and conclusions of this paper, however, apply just as much to analysis in the style of Lerdahl & Jackendoff as they do to Schenkerian analysis, and would apply to any similar reductional analysis.

2.3 Recursive functions

Earlier work on implementation of Schenkerian theory in computer software has drawn on similarities between Schenkerian theory and the concept of recursion in mathematics and computer science. As pointed out above, since the result of a reduction (or, inversely, an elaboration) is, like the original, a structure of notes, reduction (or elaboration) may be applied recursively. The earliest publication which relates Schenkerian theory to computing in this manner is (Kassler, 1967), which expresses the middle-to-background reductions of Schenkerian theory as a set of operations on matrix representations of pitch structures. Smoliar and others instead represented music in a list structure (the classic vehicle for recursion) and implemented a set of Schenkerian elaboration functions in LISP (Frankel, Rosenschein & Smoliar, 1976, 1978; Smoliar, 1980). The structure of a piece of music could thereby be expressed as a nested sequence of function calls which, when evaluated, would generate the list structure of notes corresponding to the score.

Others took as their starting point formal grammars rather than computing concepts, but the results are not significantly different. One of the most Schenkerian among these projects was by Baroni and colleagues (Baroni, 1982; Baroni, Dalmonte & Jacoboni, 1992), though it applied only to melodies. Once again, the software generated music rather than analysing it, but one attraction of using formal grammars is the possibility of using a parsing mechanism proven to be effective for a class of grammars as a means of analysis. Interest in musical grammars has therefore often focused on the application of different kinds of grammar to music (Steedman, 1984, 1996; Barbar, Desainte-Catherine & Miniussi, 1993; Desainte-Catherine & Barbar, 1994; Terrat, 2005). However, such parsing has not often been shown to be effective in analysis of pieces. One notable exception is (Pachet, 2000). Another is the Schenkerian component of Ebcioğlu’s chorale-harmonisation system (1987, 1988, 1990) which contains a grammar and heuristics for deriving Schenker-like analyses of the melody and bass lines of a chorale.

2.4 Automatic reduction

Frankel, Rosenschein and Smoliar were hopeful of extending their LISP-based work to make a system capable of automatic analysis: ‘we anticipate that on the basis of our results thus far, we should be able to formulate an experimental grammar to be used for automated analysis.’ (1976, p.30) In a later publication, however, they are more cautious: ‘it is questionable whether a program which produces Schenkerian analyses may be designed without a peripheral "world model" of musical perception'. (1978, p.134) Two years later Smoliar presented his software explicitly as an aid to the human analyst (Smoliar, 1980) with no implication of future automation.

Kassler has worked steadily on software to derive analyses according to his matrix-manipulation model of Schenkerian theory, resulting in software which is able to derive an analysis from a three-voice middleground, i.e., not directly from the score but from an already partially reduced structure (1975, 1977, 1988). This is impressive work which deserves to be better known, though unfortunately Kassler himself has not been able to develop it further (personal communication, 5 May 2005).

Musical grammars have often been concerned with chords rather than notes, partly because chords form single sequences of symbols rather than the multiple parallel sequences of notes which constitute most pieces of music. Grammars which deal with notes have tended, like Baroni's, to represent only single lines. Ebcioğlu's grammar, for example, handled the bass and melody of a chorale harmonisation separately. Furthermore, its author made clear that he did not attempt to faithfully implement Schenker's theory, but rather only aspects of it, principally linear progressions (Ebcioğlu 1987, p.88). The result was a system reported to be capable of deriving 'good hierarchical voice-leading analyses' of chorale melodies, but not of bass lines (Ebcioğlu 1987, p.79).

Mavromatis & Brown (2004) also took a grammar-based approach, but their grammar dealt with multi-voice structures rather than one-dimensional sequences of either chord symbols or notes. They demonstrated the theoretical possibility of expressing Schenkerian theory in a context-free grammar, a kind of grammar which has a particularly simple and effective parsing mechanism. While this suggested that the grammar could form the basis for automatic derivation of Schenkerian analyses, this promise has not been fulfilled because the number of re-write rules required is preventatively large (Mavromatis, personal communication, 6 March 2007). Possibly using a context-free grammar merely moves complexity from the parsing process into the grammar itself.

As will become clear below, complexity is the real stumbling-block to progress towards Schenkerian analysis by computer. A simple analysis system leads to many, many possible analyses, of which many are either incorrect or not as good as others. Although it has not been reported, I suspect that it was this complexity which prevented Smoliar and Kassler from achieving their objectives in the 1980s. Ebcioğlu overcame the problem to some degree through the use of explicit heuristics to select analysis steps which are more likely to produce good analyses. Gilbert & Conklin (2007) took as their starting point a grammar of melody with similarities to that of Baroni, but made it explicitly probabilistic so that analysis steps were more likely to lead to good analyses. A Hidden Markov Model learned probabilities based on a set of exemplars, resulting in software which was able to derive reductions from melodies with a moderate degree of success.

The software ATTA, by Hamanaka, Hirata & Tojo (2006), derived a reduction in the style of Lerdahl & Jackendoff from a melody. The problem of multitudes of possible analyses was here handled through the user adjusting parameters for each melody to arrive at an acceptable reduction (a process reported to take an expert about 10 minutes for each melody, 2006, p.271). A development of the system which uses a feedback loop from higher levels of reduction to adjust parameters automatically has been shown to be able to produce better results than no parameter adjustment, but the results are still quite distant from 'good' analyses (2007).

The most recent research explicitly directed as computational Schenkerian analysis has been by Kirlin & Utgoff. They describe a representation system and theoretical framework for the implementation of Schenkerian analysis, conceived as search through a space of directed acyclic graphs (with some similarities to the work described here). Their response to the problem of the size of the search space is essentially to work on 'pre-processing' to guide search. They report software which is able to pick out a candidate *Ursatz* (fundamental structure) from a piece of piano music (Kirlin & Utgoff, 2008). Working from the other end of the problem, Kirlin has recently implemented software which derives a foreground from a score with a notable degree of success (Kirlin, 2009).

In summary, for more than three decades there have been attempts to implement Schenkerian reduction (or similar) in computer software. Kassler's work demonstrated that this was possible for restricted kinds of music (middlegrounds), but, as demonstrated in the work of Mavromatis & Brown, there are serious impediments to effective automatic reduction. The relative success of Ebcioğlu, Gilbert & Conklin and Hamanaka, Hirata & Tojo in dealing with melodies, however, suggests that techniques for overcoming these impediments can be found for full musical textures. In the remainder of this paper, I give a proof of concept of such a technique.

3. Formalisation of Reduction

3.1 Musical primitives

A Schenkerian analysis generally consists of a set of 'graphs' presented in a kind of music notation, at several levels, often with an accompanying textual commentary and sometimes notes and comments written on the actual graphs. Here I will ignore these textual comments and deal only with the music notation of the graphs. Each graph spans the entire length of the piece, and each represents a level of reduction between the 'surface' of the piece (the notes of the score) and the *Ursatz*, which, according to the analysis, is the particular model used in this piece of one of the three forms of fundamental structure which Schenker believed to underlie every piece of proper tonal music.

The graphs most importantly consist of notes, slurs, barlines, clefs and key signatures. Barlines, if present at all, appear only in the lowest level graphs, but the use of bar numbers and the alignment of graphs with each other can imply where the time-points corresponding to barlines occur in higher-level graphs also. Other symbols on the graphs will not be considered here. While an analyst might make reference to factors such as articulation and dynamics in the course of making an analysis, these factors do not have any explicit role in the theory. One factor which is clearly of considerable importance is which instrument plays which note, but its importance seems to have been so self-evident for

Schenker that even that factor deserves no explicit role in his theory. Here, I simply side-step the issue by considering only keyboard music where this factor is irrelevant. Thus, for the present purposes, the surface of a piece of music will be considered to be a collection of notes defined by their pitch and rhythm, within a given framework of key and metre.

It might be thought that representing pitch in terms which differentiate different 'spellings' (i.e., which distinguish C♯ from D♭) would be appropriate for an application such as this dealing with tonal music. However, it is only when a pitch has a role within a musical structure that one can properly distinguish between two notes sounding the same but having different spellings, and these structures are to be derived from the surface representation rather than to be expressed within it. Furthermore, I know of no tonal structure in which two pitches which sound the same but are spelled differently are simultaneously present. (Structures in which they are successively present are found in many 18th- and 19th-century pieces.) Thus, since it is best to keep things simple until they need to be made more complex, I will assume that pitches can be represented within the twelve-tone scale, and in each derived tonal structure it will be possible to unequivocally determine how a pitch should be spelled, should that be required. Thus pitches can be represented in a twelve-tone scale and considered equivalent to integers, and it is simplest to use the existing MIDI standard pitch codes for this purpose. The common concept of the pitch class, which ignores the octave in which a pitch occurs, will also be used. Pitch classes will be represented simply as integers modulo 12. The tonal structure of a piece depends on a key which, strictly speaking, should be derived in the course of analysis. To simplify early development, however, the key is taken to be explicitly stated in advance of analysis; at a future stage of development, the key should be inferred by the analysis process.

Schenker gives no significance to absolute duration: whether a note lasts for a second or a minute has no explicit role in his theory. Schenker writes little about relative durations, either, but they clearly are important in the making of Schenkerian analyses. Thus durations are represented only in relative terms, for which rational numbers with no absolute significance can be used. Rhythm has an aspect of stress as well as duration, inherent in the strong and weak beats of metre, an aspect of the structure which, like key, should properly be derived in the course of analysis. However, as in the case of key, the metre is currently given in advance as an offset from the start of the piece to the first strong beat, and a pattern giving the lengths of strong and weak beats in the same terms as the rational numbers of the durations of notes. Again, at a future stage of development the metre should be inferred by the analysis process.

Following this reasoning, the surface of a piece of music could be represented as a collection of notes, each represented by three numbers: an integer for the pitch, a rational number for the duration, and another rational number to represent the time at which the note starts relative to the start of the piece. However, for reasons which will become clear later, it is useful to be able to treat a piece as a succession of events each consisting of any number of sounding notes (i.e., a single note or a chord), or none (i.e., a rest), and at which each note might start (or start again) or might be a continuation of a note in the previous event (a tied note). These events will be called 'segments' (corresponding to the 'time spans' of Lerdahl & Jackendoff). (In some music-theoretic texts such a construct is called a ‘simultaneity’, but sometimes the word is also used for instants which have no duration or an indeterminate duration, so the word ‘segment’ is used here to avoid any potential confusion.) In this representation, individual notes do not need to have durations: the duration can be attached to the segment and all notes within the segment will have that duration. Furthermore, starting times do not need to be represented, because they are inherent in the sum of the durations of the preceding segments in the sequence. On the other hand, it is now necessary to add to each note a property to indicate whether it is tied to a preceding note or is a newly started note. Furthermore, a single note in a score might have to be represented by several notes across several segments, with a tie on the second and subsequent notes.

The surface of a piece is thus represented as a sequence of segments. A segment has a duration (represented by a rational number) and a set of notes (possibly empty). Each note has a pitch (represented by an integer) and a state of being tied or not tied to a preceding note. (An illustration is given in Figures 1 and 2.) Note that there is no explicit representation of voices: the notes of one segment follow the notes of another with no implication that any note in the first is in the same voice as any note in the second. While the voices of a piece are generally evident to the listener and to someone reading the score, they are not generally explicitly represented in the score in keyboard music. Furthermore, there are many cases in Schenkerian analysis where elaborations involve notes from more than one voice, or where one voice at the surface becomes more than one voice at some level of reduction. In general, the analysis system presented here employs no explicit concept of voice, except to give a particular role in some situations to the lowest or highest notes of segments.

Since we are dealing with keyboard music, this representation can be thought of in terms also of activity on the keyboard, rather like a piano roll. Each segment corresponds to a state of the keyboard—whether each note is depressed or not—and lasts for a specific relative duration. At the beginning of each state, each depressed note can have been held down (corresponding to a tied note) or struck (corresponding to a non-tied note). This detail would not be required in a strict representation of the changing state of the keyboard, since every occurrence of a struck note must be preceded by a state of the keyboard in which the note is not depressed. However, these states might have an extremely short duration and have no independent structural role in the music. In the same way, the minute differences in timing between the beginnings of notes which are played ‘together’ (but not actually together) are ignored. The representation thus does not correspond to a strict representation of the changing physical state of the keyboard but rather to an idealised representation which abstracts from this the musically significant changes of state. One consequence of this correspondence is that the analysis procedure described below could, subject to this idealisation (which is not trivial), be applied to MIDI representations of pieces of music.

3.2 Binary trees

Lerdahl & Jackendoff explicitly represent their reductions in tree structures. Schenker, working within a different and earlier intellectual tradition, did not, but the work of Kassler and Smoliar demonstrates that a conversion of his graphs to tree structures is possible (discussed in detail in Marsden, 2005). To simplify, a reductional structure will here be considered to be a *binary* tree. Two aspects of Schenkerian theory do not fit simply into this simple structure: elaborations which produce sequences of notes and elaborations which depend on context outside the time span being elaborated. These introduce some additional complications, discussed below, but it appears that the benefit of the simplicity of the binary tree structure is worth the cost of accommodating these complications.

Each node of a reductional tree is a segment (as defined above) (again following Lerdahl & Jackendoff). The duration of a segment which is not a terminal (i.e., not a 'leaf' of the tree) is equal to the sum of the durations of the two segments below it in the tree. In this I follow Komar (1971) who adds treatment of rhythm to Schenkerian theory by regarding elaboration as introducing additional notes within a specific time span. Thus at each branching point a longer (more background) span is divided into shorter (more foreground) spans.

As mentioned above, the highest level of a Schenkerian analysis is an *Ursatz*, one of a class of three prototype structures which Schenker believed to underlie every great piece of music. Each of these begins and ends with tonic harmony in root position, has root-position dominant harmony immediately preceding the final tonic, and has a top line (the *Urlinie*) which descends by step from the tonic, fifth or third of the scale to the tonic an octave, fifth or third below respectively. However, it makes for a simpler formal structure if reduction is considered to continue beyond the *Ursatz* until a single segment is reached, making a single tree. Indeed Schenker himself considered the three forms of the *Ursatz* to be derived from the static tonic triad, the 'chord in nature'. As formalised here, therefore, a legitimate analysis will be a binary tree of segments, of which the highest (the 'root') has root-position tonic harmony, and which contains a sequence of segments forming an *Ursatz*. This sequence must span the entire tree, i.e., every other segment in the tree must be either an ancestor or a descendent of a segment in the sequence.

3.3 Atomic elaborations

The notes of segments which are not the highest-level segment (the 'root') can be derived by applying specific 'atomic elaborations' to the notes of the segment above in the tree. Atomic elaborations correspond to the simplest techniques of diminution described by Schenker in *Der Freie Satz*, some of which are familiar from the theory of ornamentation: appoggiaturas, suspensions, neighbour notes, and the like. Each elaboration refers to a single note in the segment above, which will be called the 'parent', and produces two notes (or a note and a rest), called the 'children', one in each of the segments below. The precise vocabulary of possible elaborations should be a topic of research, based on a close reading of theoretical texts such as Schenker's and empirical research on actual analyses, and it will vary according to the repertoire of music to be analysed. However, for this proof of concept the simple set of elaborations defined in Table 1 will be used. Schenker's more complex techniques of diminution can generally be formed by combinations of these atomic elaborations; that all of them can be so formed is yet to be proven.

Name	**parent**		**1st child**		**2nd child**		**pre-context**	**post-context**	**consonant pitches**	**inheritance**
type	*pitch*	*tie*	*pitch*	*tie*	*pitch*	*tie*	*pitch*	*pitch*	*set of pitch class*	*inheritance*
consonant skip 1	x	t	x	t	y	no	$\varnothing$	$\varnothing$	$x + y$	either
consonant skip 2	y	no	x	t	y	no	$\varnothing$	$\varnothing$	$x + y$	either
appoggiatura	y	no	$x = y\pm1$ or $y\pm2$	no	y	no	$\varnothing$	$\varnothing$	y	2nd
anticipation	x	t	x	t	y	no	$\varnothing$	y	x	1st
neighbour note	x	t	x	t	$y = z\pm1$ or $z\pm2$	no	$\varnothing$	z	x	1st
suspension	y	no	$x = y\pm1$ or $y\pm2$	yes	y	no	x	$\varnothing$	y	2nd
repetition	x	t	x	t	x	no	$\varnothing$	$\varnothing$	x	either
hold	x	t	x	t	x	yes	$\varnothing$	$\varnothing$	x	either
shortening	x	t	x	t	*rest*	no	$\varnothing$	$\varnothing$	x	either
delay	x	no	*rest*	no	x	no	$\varnothing$	$\varnothing$	x	either
interruption*	x	t	x	t	y	no	$\varnothing$	$\varnothing$	I	IV

Table 1. Definitions of 'atomic elaborations'. *Additional conditions also apply to the 'interruption' elaboration: the harmony of the second segment must be V or V^7.

The definitions in Table 1 are probably too restrictive in some details. For example, the definition of ‘repetition’, ‘hold’, ‘shortening’ and ‘delay’ in Table 1 (elaborations which in music theory seem so obvious as not to warrant definition) require the notes to be consonant. In actual music, however, it is not difficult to find dissonant notes which are repeated or followed or preceded by a rest. However, allowing looser definitions considerably increases the complexity of the reduction process. Since, as will become clear below, the computational complexity is already severe, this proof of concept takes this restrictive definition for these elaborations, which is adequate in most cases.

Other definitions are probably too loose. The ‘consonant skip’ elaboration generates a leap from one consonant note to another, and exists in two forms, one in which the parent note has the pitch of the first note and one in which the parent note has the pitch of the second. This elaboration is not one referred to by Schenker, but not every case in which he appears to use an elaboration generating a leap from one consonant note to another conforms to his definition of ‘arpeggiation’ (the term he uses more commonly). Forte & Gilbert make extensive use of the term ‘consonant skip’ but do not give a clear definition. Most occurrences of ‘consonant skips’ in the literature occur as a leap of a third or fourth from one note to the next above or below in the prevailing harmony, but sometimes leaps of larger intervals are labelled as consonant skips. Therefore a non-restrictive definition of consonant skip has been used here which accommodates a leap to any note of the same harmony, regardless of the interval. Deriving acceptable analyses from the test pieces used here was not possible when this definition was restricted to intervals of a third or fourth, but it remains worryingly loose to allow a skip to any note of the harmony with no constraint.

Even looser is the elaboration I call ‘interruption’. This is a kind of elaboration Schenker explicitly describes, but only in specific situations. It occurs when there is a move from the tonic to the dominant, with the *Urlinie* descending to the second degree of the scale, followed by a caesura, a resumption of the tonic and a return to the first note of the *Urlinie*. The pattern thus only occurs for Schenker at the higher levels of reduction. However, something like it evidently occurs also at lower levels, where there is a move to the dominant which does not proceed smoothly to notes which follow. Again, an elaboration like this appears necessary to allow derivation of acceptable analyses from pieces such as the test pieces used here, but its use should probably be more constrained.

Some common techniques of diminution can produce more than one new child note: passing notes and arpeggiations. The simplest passing note is a single note placed between two others a third apart, but passing notes can also exist as two notes placed between notes a fourth apart, which implies the need for ternary branching in the tree. One possibility is to introduce an additional kind of ‘note’ to represent an intermediate stage in a multiple-passing-note elaboration, and represent such passing-note elaborations in more than one binary branching. A similar technique can be used to represent multiple-note arpeggiations. Introducing this additional kind of ‘note’, however, considerably increases the computational complexity of the reduction process. As for the restrictive definition of ‘repetition’, ‘hold’, ‘shortening’ and ‘delay’ elaborations, this proof of concept will take the expedient, but probably ultimately incorrect, approach of assuming that every passing-note elaboration can be represented as a set of one of more neighbour-note elaborations applied recursively. Multiple-note arpeggiations will be represented by recursive consonant-skip elaborations.

In the formal definition given below, the term ‘atomic elaboration’ is used for elaborations such as those described here which have a single note as parent and a pair of notes (or a note and a rest) as children. The term ‘elaboration’ is used to refer to a ternary relationship between *segments* whereby a parent segment is elaborated to become a pair of child segments. Such an elaboration must be made up of a valid set of atomic elaborations.

Every segment is considered to contain a notional rest, even if no rest is notated in the score. Thus there is always a rest available to be a child of a ‘shortening’ or ‘delay’ elaboration. This allows configurations such as ‘unfoldings’, where a single voice adumbrates more than one voice in ‘pseudo-polyphony’, to arise naturally by multiple ‘shortening’ and ‘delay’ elaborations.

3.4 Context notes

The simplest atomic elaborations, such as ‘repetition’ and ‘consonant skip’, require no reference outside of the span of the note to be elaborated. Others, like anticipations and neighbour notes, can exist only in the presence of an appropriate following note. Suspensions require a particular preceding note. These will be referred to as ‘post-context’ and ‘pre-context’ notes respectively. The requirements, if any, for each atomic elaboration are specified in Table 1.

This aspect of elaborations considerably complicates reduction in that one branch of a tree cannot be derived (or generated) without reference to another. It also complicates the correspondence between the tree structure representing the reduction of a piece and segmentations of that piece. (Lerdahl recognises an equivalent difficulty and accommodates it within his theory by a modification of the grouping rules to allow ‘polyphonic grouping’, i.e. grouping in different parts which is ‘out of phase’ (Lerdahl, 2001, pp.32-34).)

Most importantly, two segments can only legitimately be children of a parent segment if all of the elaborations along the ‘trailing edge’ of the tree below the first child (i.e. the elaborations of that child, and of its second child, and of the second child of that child, etc.) find their required post-context pitches in the second child or the segments along the ‘leading edge’ of the tree below that second child, and *vice versa* for the pre-context pitches of the elaborations along the ‘leading edge’ of the tree below the second child. Furthermore, an elaboration cannot find its required post-context pitch in a segment which is at a higher level than an elaboration which finds its required pre-context pitch in a parent of that elaboration. These constraints can be equivalently expressed as follows: it must be possible to progressively elaborate from the root segment by applying one elaboration at a time to any currently non-elaborated segment in a sequence such that at every step the required pre-context notes of every atomic elaboration making up the elaboration of that segment are present in the immediately preceding unelaborated segment and the required post-context notes of every atomic elaboration are present in the immediately following unelaborated segment.

The formal definition below, and the implemented reduction process, make use of the concept of a ‘valid sequence’ of segments. This is a pair of segments which conforms to the constraint on contexts described above. In the base case, every pair of consecutive surface segments is a valid sequence (because surface segments are not elaborated). From this the validity of any other pair of consecutive segments can be induced: a pair is valid if either the pair made of the second child of the first segment and the second segment is valid or the pair made of the first segment and the first child of the second segment is valid.

3.5 Harmonic constraints and combinations of elaborations

Elaborations generally require certain notes to be consonant. (The others need not be dissonant, but often are, or at least are dissonant with respect to the time span of the elaborated note.) When simultaneous elaborations occur, the consonances they imply must be consistent. This is defined to mean that the pitch classes of the consonant notes must be members of an acceptable harmony. Properly, a complex definition referring to keys and progressions is required for harmonies, but for the present purposes it is sufficient to define harmonies as sets of pitch classes. Every segment has associated with it a harmony, which is a (possibly empty) set of pitch classes. This set indicates those pitch classes which are consonant within that segment. Note that it is not necessary for all the pitch classes in the set of pitch classes to be actually present in the notes of that segment. All surface segments have an empty harmony.

Eight kinds of acceptable harmony are defined: four triads—minor, major, diminished and augmented—and four sevenths—dominant, minor, diminished, and half diminished. (Harmonies should be added to or removed from this list according to the repertoire under examination; as stated previously, the objective at this stage is to keep definitions simple.) The harmony of a segment is acceptable if it is equal to or a subset of one of these eight. In the reduction process described below, a distinction is made between ‘simple’ harmonies—minor, major and diminished triads, and the dominant seventh of the current key—and ‘complex’ harmonies—augmented triads, minor, diminished and half-diminished sevenths, and dominant sevenths in other keys. This is intended to encompass the concept that some harmonies are more consonant than others, and to reflect in part Schenker’s theory concerning seventh chords. While Schenker made clear that he did not regard seventh chords as consonant harmonies like triads (see Brown, 2005, p.58), he did also allow the possibility of a dominant seventh occurring even at the first level of elaboration of the Ursatz (see, for example, Figure 19 of *Der freie Satz* (Schenker, 1935, supplement)). I therefore include the dominant seventh among the ‘simple’ harmonies (an approach recommended in Pankhurst, 2008, pp.22–25).

In a reductional structure, the harmony of the parent segment must be equal to, or a superset of, the harmony of at least one of the child segments. Which child this must be is characterised as ‘inheritance’: ‘1st’ indicates that the harmony of the first child is equal to or a subset of the harmony of the parent, ‘2nd’ that the harmony of the second child is equal to or a subset of the harmony of the parent, while ‘either’ indicates that either or both may be equal or a subset. A special inheritance of ‘IV’ is used in the case of ‘interruption’ elaborations to specify that the harmonies of the children must be equal to or subsets of I and V^7 respectively. At later levels of reduction, an inheritance of 'both', requiring the harmonies of both children to be equal to or subsets of the parent harmony, is also possible. (Schenker does not state this requirement for inheritance of harmony in his theory, but it is evident from his analyses. It is explicit in the theory of Lerdahl & Jackendoff (1983, pp.152–155).)

Again, although not a recognised component of Schenkerian theory, it is useful to make a distinction in the harmonic behaviour of children between progressions and non-progressions. (This is similar to the 'progressions' and 'prolongations' of Lerdahl & Jackendoff (1983, pp.181–182).) A progression occurs when the harmonies of one of the children is not equal to or a subset of the harmony of the parent. In this case all the notes of each of the two child segments must be consonant with their respective harmonies. In the case of a non-progression, when the parent harmony is equal to or a superset of the harmony of both child segments, it is possible for dissonant notes to occur.

There seems no reason to insist that a parent note cannot be elaborated by more than one atomic elaboration. On the contrary, there appear to be instances in actual analyses where a single note is elaborated simultaneously in more than one way. Similarly, there seems no reason to insist that a single note cannot be the child of more than one atomic elaboration. On the other hand, the logic of the tree structure of segments requires that every note in a segment which is not a leaf must be a parent of at least one atomic elaboration. Furthermore, every note in every segment which is not the root must be a child of at least one atomic elaboration.

Schenker does admit 'implied notes' which occur at higher levels but do not correspond to any note with the same pitch at the surface, and also the possibility that a note might occur at a higher level in a different octave from that at which it occurs at the surface, by 'register transfer'. However, these are subtleties which are not accommodated in this proof of concept, and in the formalisation used here every atomic elaboration has a parent note which has the same pitch as one of its children. This means that the two children can be designated as a 'main note' and a 'subsidiary note'. While, as indicated above, a single note can be a child of more than one atomic elaboration, it cannot be a subsidiary note in one but a main note in another.

3.6 Summary

To summarise, the formal theory presented here uses the following concepts

pitch: integer
duration: rational
pitch class: integer modulo 12
tie: boolean
note: pitch × tie
rest
harmony: set of pitch class
inheritance: '1st', '2nd', 'either', 'both' or 'IV'.
segment: (set of note + rest) × duration × harmony
surface: sequence of segments

analysis: binary tree of segments

and the following relations (‘_’ stands for any unspecified argument, $\varnothing$ for the empty set, I for the tonic triad, and V7 for the dominant seventh):

class(p, pc): pitch × pitch class

$pc = p$ modulo 12; the operator **class(p)** can be defined to yield the pitch class pc such that class(p, pc).

consonant(p, h): pitch × harmony

consonant(class(p) $\cup$ h)

consonant(h): harmony

The harmony must be equal to or a subset of one of a certain set of harmonies, defined by the tonal language. In the system described here this set is all major, minor, diminished and augmented triads, and all dominant, minor, diminished and half-diminished sevenths.

atomic_elaboration(n, $n1$, $n2$, $p1$, $p2$, h, i):

note × (note $\cup$ rest) × (note $\cup$ rest) × (pitch $\cup$ $\varnothing$) × (pitch $\cup$ $\varnothing$) × harmony × inheritance

The arguments must conform to one of the elaboration patterns defined by the tonal language, such as those defined in Table 1. The arguments are, respectively, the parent note, the first child note, the second child note, the pre-context pitch (if any) required to be present in some preceding segment, the post-context pitch (if any) required to be present in some following segment, the set of pitch classes which must be consonant (the harmony), and the inheritance.

elaboration(s, $s1$, $s2$, E): segment × segment × segment × set of atomic_elaboration

Let h, $h1$, $h2$ be the harmonies of s, $s1$, $s2$ respectively.

All the following must be true:

valid_sequence($s1$, $s2$).

consonant(h).

$\{n: (n, \_, \_, \_, \_, \_, \_) \in E\}$ = notes of s.

$\{n1: (\_, n1, \_, \_, \_, \_, \_) \in E\}$ – rest = notes of $s1$.

$\{n2: (\_, \_, n2, \_, \_, \_, \_) \in E\}$ – rest = notes of $s2$.

Let $P1 = \{p1: (\_, \_, \_, p1, \_, \_, \_) \in E\}$.

If $P1 \neq \{\varnothing\}$, there must exist a segment $s3$ such that valid_sequence($s3$, s) and $\forall p1 \in P1$ ($\exists n3$ component of $s3$ ($p1$ = pitch of $n3$)).

Let $P2 = \{p2: (\_, \_, \_, \_, p2, \_, \_) \in E\}$.

If $P2 \neq \{\varnothing\}$, there must exist a segment $s4$ such that valid_sequence(s, $s4$) and $\forall p2 \in P2$ ($\exists n4$ component of $s4$ ($p2$ = pitch of $n4$ and $n4$ is not tied)).

Let $h4 = \cup\{h3: (\_, \_, \_, \_, \_, h3, \_) \in E\}$ and $I = \{i: (\_, \_, \_, \_, \_, \_, i) \in E\}$.

Either $(h1 \cup h2 \cup h4) \subseteq h$

or let $h1a = h1 \cup \{$class(pitch of $n1$): $n1$ member of $s1\}$ and $h2a = h2 \cup \{$class(pitch of $n2$): $n2$ member of $s2\}$.

consonant($h1a$) and consonant($h2a$) and

either $I \subseteq \{$1st, either$\}$ and $(h1a \cup h4) \subseteq h$

or $I \subseteq \{$2nd, either$\}$ and $(h2a \cup h4) \subseteq h$

or I = IV and $h1a \subseteq$ I and $h2a \subseteq$ V7 and h = I

```
valid_sequence($s1$, $s2$): segment $\times$ segment
    Either $s2$ follows $s1$ in the surface
    or let $E1$ be a set of atomic elaborations such that elaboration($s1$, _, $s12$, $E1$), and
        valid_sequence($s12$, $s2$) and
        let $P2 = \{p2\text{: } (\_, \_, \_, \_, p2, \_, \_) \in E1\}$.
            Either $P2 = \{\emptyset\}$ or $\forall p2 \in P2$ ($\exists n2$ component of $s2$ ($p2 =$ pitch of $n2$)).
    or let $E2$ be a set of atomic elaborations such that elaboration($s2$, $s21$, _, $E2$), and
        valid_sequence($s1$, $s21$) and
        let $P1 = \{p1\text{: } (\_, \_, \_, p1, \_, \_, \_) \in E2\}$.
            Either $P1 = \{\emptyset\}$ or $\forall p1 \in P1$ ($\exists n1$ component of $s1$ ($p1 =$ pitch of $n1$)).
```

As noted above, formally every segment contains a rest, even if no rest is notated in the score at the corresponding point. (The only place where a rest must be notated is where a segment contains no notes.) The piece to be analysed must be represented as a sequence of segments whose harmonies are all ∅ (the empty set). To derive a reduction, one needs to find an analysis (a tree of segments) such that the sequence of leaves of the tree is equal to the surface to be analysed, and for every segment s of the analysis and its children $s1$ and $s2$, a valid relation elaboration(s, $s1$, $s2$, _) holds.

3.7 Example of formal reduction

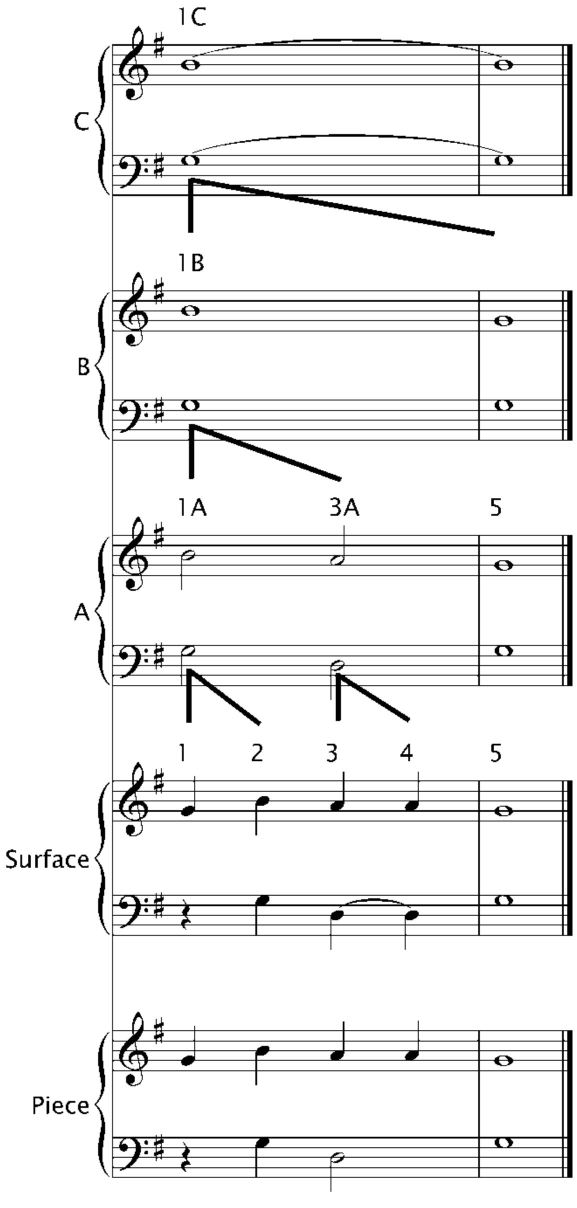

Figure 1. Final bars of ‘Au claire de la lune’

Surface:

[1, {<67, no>}, ∅], [1, {<71, no>, <55, no>}, ∅], [1, {<69, no>, <50, no>}, ∅],
[1, {<69, no>, <50, yes>}, ∅], [4, {<67, no>, <55, no>}, ∅]

Reduction:

```
[8, {<71, no>, <55, no>}, {2, 7, 11}]
    [4, {<71, no>, <55, no>}, {2, 7, 11}]
        [2, {<71, no>, <55, no>}, {7, 11}]
            [1, {<67, no>}, ∅]
            [1, {<71, no>, <55, no>}, ∅]
            {consonant skip 2, delay}
        [2, {<69, no>, <50, no>}, {2, 9}]
            [1, {<69, no>, <50, no>}, ∅]
            [1, {<69, no>, <50, yes>}, ∅]
            {repetition, hold}
        {neighbour note, consonant skip 1}
    [4, {<67, no>, <55, no>}, ∅]
    {consonant skip 1, repetition}
```

Figure 2. The surface and reduction of Figure 1 in numerical notation. Each segment is notated as [duration, notes, harmony], and each note as <pitch, tied>.

The analysis of an example taken from the end of the French folk song 'Au claire de la lune' is illustrated in Figure 1. As described above, the piece must first be split into segments, as in the line labelled 'Surface', so the minim (half note) D3 is divided into two crotchets (quarter notes), the second tied to the first. Figure 2 gives the same information (but this time containing information on harmonies) in a numerical form.

The first stage of reduction is to derive segment 1A from segments 1 & 2. These two segments form a valid sequence, since both are surface segments. There is just one note in segment 1, but every segment also includes a notional rest, so two atomic elaborations can form segment 1A from segments 1 & 2:

consonant_skip_2(<71, no>, <67, no>, <71, no>, ∅, ∅, {7, 11}, either)
delay(<55, no>, rest, <55, no>, ∅, ∅, {7}, either)

The combined inheritance of these two atomic elaborations is 'either', so the harmony of segment 1A must be a superset of the harmony of either segment 1 or segment 2, and of the union of the required harmony of the atomic elaborations, i.e., {7, 11}. The harmony of segment 1A is therefore {7, 11} (i.e., pitch classes G and B must be consonant), which is a valid harmony. The duration of segment 1A is the sum of the durations of segments 1 & 2, i.e., 2.

Segment 3A similarly derives from segments 3 & 4 by the following two atomic elaborations:

repetition(<69, no>, <69, no>, <69, no>, ∅, ∅, {9}, either)
hold(<50, no>, <50, no>, <50, yes>, ∅, ∅, {2}, either)

The harmony of segment 3A is {2, 10} (i.e., pitch classes D and A must be consonant).

Segments 1A & 3A make a valid sequence: the set of atomic elaborations which make segment 1A has no required post-context pitches and the set which makes segment 3A has no required pre-context pitches; the second child of segment 1A, segment 2, is a surface segment and so makes a valid sequence with segment 3, also a surface segment; therefore segment 1A makes a valid sequence with segment 3, and since this is the first child of segment 3A, segment 1A makes a valid sequence with segment 3A. Furthermore, they can be reduced to segment 1B by the following set of atomic elaborations:

neighbour_note(<71, no>, <71, no>, <69, no>, ∅, 67, {11}, 1st)
consonant_skip_1(<55, no>, <55, no>, <50, no>, ∅, ∅, {2, 7}, either)

The harmony of segment 1B must therefore be the union of the required harmony of these two atomic elaborations, and of the harmony of segment 1A (by virtue of the '1st' inheritance of the first atomic elaboration). Its harmony is therefore {2, 7, 11} (i.e., pitch classes G, B and D must be consonant).

Finally, segment 1C is formed from segments 1B and 5. That these two segments make a valid sequence is slightly more complicated, because one of the atomic elaborations involved in the derivation of segment 1B requires a post-context pitch: the pitch 67 (G4), to which the neighbour note resolves, must be found. This pitch is present in segment 5, and so the segments form a valid sequence. Segment 1C can then be derived by the following two atomic elaborations:

consonant_skip_1(<71, no>, <71, no>, <67, no>, ∅, ∅, {7, 11}, either)
repetition(<55, no>, <55, no>, <55, no>, ∅, ∅, {7}, either)

The segments at level A form an *Ursatz*: the three segments have top pitches in the sequence 71, 69, 67 (B4, A4, G4) forming a step-wise descent to the tonic. The first and last segments have tonic harmony and the tonic pitch in the bass. (In fact the harmony of the last segment is the empty set, which means that strictly any pitch can be consonant, but this admits the possibility of tonic harmony, which is all that is required.) This reduction therefore forms a valid analysis of this tiny piece.

Note that other sets of atomic elaborations can produce the same reduction, and there are other possible valid reductions of this piece. Music theorists admit the possibility that there can be more than one valid Schenkerian analysis of a piece of music. A great deal of the literature discusses alternative analyses, with a strong implication that some analyses are better than other valid but weaker analyses. As will become clear below, a major difficulty in the implementation of Schenkerian analysis is to choose between many alternative analyses.

4. A Reduction Algorithm

4.1 Deriving a reduction

From inspection of Table 1, it is clear that, given any pair of notes *n1* and *n2*, it is possible to determine the possible parent notes and elaborations which could generate this pair of notes, i.e., to find all members of the relation atomic_elaboration(*n*, *n1*, *n2*, *p1*, *p2*, *h*, *i*) which have *n1* and *n2* as arguments. Using this information, for any pair of segments *s1* and *s2*, it is therefore possible to determine all possible parent segments *s* such that elaboration(*s*, *s1*, *s2*, _). A procedure for this is as follows. First we must ensure that *s1* and *s2* form a valid sequence. We can determine whether or not this is the case from a record of the intermediate segments and pre- and post-context requirements of elaborations by which *s1* and *s2* are derived from the surface. Then, taking each possible pair of notes and rests from *s1* and *s2*, and each combination of the atomic elaborations which can apply to these pairs, we can determine valid combinations of atomic elaborations by taking the union of their harmonies and their inheritances and ensuring that these are valid harmonies, and we can form the resulting segments from the set of resultant parent notes and the resultant harmonies.

The only tricky part of this is the requirement for possible context notes *p1* and *p2* in atomic elaborations, since these do not need to occur at the surface of the piece but might instead arise in reductions from the surface. (In the piece illustrated in Figures 1 and 2, for example, the G4, which is the required context for the 'neighbour note' elaboration by which segment 1B is derived, need not have occurred at the beginning of the second bar of the surface. There might have been some elaboration of this note which places a different note or a rest at the beginning of that bar. The required context note would then have occurred not at the surface but at level A, in a new segment 5A.) However, the only pre-context note required in the set of elaborations defined in Table 1 is for suspensions, and this note (called a 'preparation' in music theory) must be tied to the first of the child notes, and therefore it must be present at the surface, since no note can intervene between the note tied from and the note tied to. (In fact, this is an oversimplification, and in some repertoires suspensions do occur with a 'remote' preparation which, on the surface, does not occur immediately before the suspension and is not tied to it. However, such cases could also be considered, when using the set of elaborations of Table 1, to be appoggiaturas, which require no pre-context.) Thus it is only post-contexts which need to be sought in levels above the surface. If reduction proceeds 'backwards' through a piece, from the end to the beginning, any required post-context notes will, at each point, have already been found in reductions of the following section of music.

A naive reduction procedure, then, might select a pair of as-yet unreduced segments near the end of the piece and determine a set of atomic elaborations and a parent segment which could produce this pair as children. Then it would recursively select another pair of unreduced segments, possibly including the newly created parent segment, and determine its set of atomic elaborations and parent. If for any chosen pair a set of atomic elaborations and parent segment cannot be found, the procedure could try another pair instead. If all possible pairs have been tried, the procedure could backtrack, undoing the last reduction, and try a different reduction, either for the same pair of for a different pair. At each step, except one involving backtracking, the number of segments still to be reduced is one less. Provided the backtracking procedure tries every possibility and does not try a possibility which has been tried before, the procedure will eventually find a complete reduction if one exists, resulting in just a single segment at the highest level.

This is essentially the same argument as used by Kassler (1976) to demonstrate that automatic Schenkerian reduction is a logical possibility. However, the naive procedure suffers from two faults. Firstly, there is no guarantee that an analysis will be found in a reasonable time. It is entirely possible that the procedure will spend so long finding a valid analysis of the piece as to be useless. Secondly, experiments have shown that the atomic elaborations defined here can produce analyses which are valid in the terms of this theory, but which most analysts would regard as unacceptable. Furthermore, it is possible (indeed likely) that different definitions of elaborations, unless the definition is vastly more complex, are likely to also produce unacceptable analyses. The 'rules' of Schenkerian analysis seem to be too loose to determine acceptable analyses by themselves alone. (Temperley makes a similar point based on some initial experiments. (Temperley, 2007, pp.175-176)) Other factors must come into play to distinguish acceptable from unacceptable analyses or, better, good from bad. A path towards solution of these two problems is the main contribution of this paper.

4.2 Explosion of the solution space

For all but the smallest pieces, the size of the possible solution space—the number of possible analyses—is extremely large, and grows extremely rapidly with larger pieces, for two reasons. Firstly, the number of binary trees with the same number of leaves but with different branching increases very rapidly with increases in the number of leaves. If all possible branchings are allowed, the number of different binary trees for *n* leaves is given by the 'Catalan number' C_n: $(2n)!/(n+1)!n!$. There are 14 trees with five leaves, but 4862 with ten. As we will see below, for actual pieces of music, even taking into account restrictions on branching, the number of possible trees is well over a trillion.

Secondly, the notes from segments reduced together can be paired differently. It is not necessarily the case that the top note of the first segment will be paired with the top note of the second, etc. The top note of the first segment might be paired with a rest, for example, and its second highest note be paired with the top note of the second segment. To determine the total number of different possible analyses, the number of different pairings in the reduction of one pair needs to be multiplied by the number of reductions in the preceding pair, etc., with the result that the number of possible analyses rises exponentially as the length of the music to be analysed increases.

4.3 Matrix of partial solutions

A common technique to overcome such combinatorial explosion is to use dynamic programming. This is a procedure which constructs an array of local solutions together with some scoring metric. It is necessary that each local solution and score be derivable from the information stored in the immediately neighbouring local solutions on which it is based. Once the array is complete, the optimal global solution is constructed by tracing back the path through the best scoring local solutions to the origin. The technique typically transforms a problem of exponential complexity to one of polynomial complexity.

The technique used here is of this kind, and essentially uses the CKY algorithm (Jurafsky & Martin, 2009, pp.470–477) to act as a kind of chart parser (Jurafsky & Martin, 2009, pp.482–484). This constructs a 'chart' of partial parses, like a dynamic-programming array, on the way to making a complete parse of a sentence. An empty triangular matrix of 'cells' is formed with the segments of the surface in the lowest row. The second row of cells will hold segments which will each span two surface segments, the third row segments spanning three surface segments, etc., until the highest cell contains segments which span the entire piece. Essentially using the bottom-up CKY algorithm, this array is filled from the bottom right with all the possible segments which result from reduction of each of the pairs of segments which cover the span of each cell.

Once the matrix is filled, complete analyses can be extracted from the matrix by selecting one of the top-level segments, and then recursively selecting pairs of possible child segments until the surface is reached. The selection process is rather complicated because is it necessary to ensure fulfilment of the context constraints on the elaborations by which parents are derived from children. Thus selecting a child in one branch of the analysis can have the effect of invalidating certain selections in an adjacent branch. If records are kept of the context constraints for each parent–child relationship, elaborations rendered invalid in another branch can be removed, restricting the choice of children in that branch. However, the constraints can have remote consequences also, and the currently implemented procedure does occasionally reach a point where no valid children can be selected. In such cases selection must either backtrack or start again. The analysis selection procedure is thus, in the worst case, of factorial complexity, and so I cannot claim to have implemented a Schenkerian-analysis procedure of polynomial complexity. However, in practice the constraints are rather loose, and so backtracking is not common and the worst case rarely arises.

As the matrix is filled, three additional pieces of information are attached to derived segments:

1. 'Goodness' score. This allows later selection, in dynamic-programming fashion, of a best-scoring analysis. Section 5 describes a first attempt to establish a suitable metric for goodness.
2. *Ursatz* possibilities. This gives the parts of an *Ursatz* which a segment can supply (whether as single upper and bass notes or as a sequence of such notes in its children and other descendents). This allows selection of top-level segments and subsequent children to be restricted to only those which yield an analysis containing an *Ursatz*.
3. The multiplicity of sub-trees. This is an upper bound on the number of different possible sub-trees there are below the segment concerned. This information is used in selecting a sample of analyses and is described in more detail in Section 5.2.

Once the matrix is filled, it is pruned in a top-down process which removes segments below the top level which have no parent. At the same time, the record of *Ursatz* possibilities is also pruned to remove records of possibilities which turn out to be impossible to realise because the necessary corresponding parts of an *Ursatz* can never arise from neighbouring segments.

4.4 Computational complexity

The time complexity of the CKY algorithm is generally described to be of order O(n^3). The size of the matrix, and so the number of cells to be filled, is clearly related to the square of the number of surface segments. The number of pairs of lower-level cells to be considered when filling a cell increases linearly with the height of the row of the cell, related in turn to the number of surface segments, yielding an order where the square of the number of segments (for the number of cells) is multiplied by the number of surface segments once more (for the number of pairs to be considered at each cell), and the quoted order of O(n^3). A further factor to be considered, however, is the number of alternative segments stored in each cell. At the surface there is only one segment for each cell, but at higher levels there can be many alternative reductions. In the absence of information on which to make a better inference, we might assume that the number of alternative segments stored in a cell increases linearly with the height of the row of the cell. The maximum number of alternative pairs of segments to be considered for reduction for each pair of cells is the product of the maximum number of alternative segments in each cell. This maximum number of segments is related to the number of surface segments, so the maximum number of pairs is related to the square of the number of surface segments, and the overall time complexity should be taken to be O(n^5) [O(n^3 * n^2)]. On the other hand, there is a ceiling on the number of possible alternative segments in a cell, because there is a finite limit to the number of possible combinations of notes which can be played simultaneously on a keyboard, so this factor can strictly be considered as constant and the time complexity taken to be the theoretical O(n^3). In practice, the time complexity is likely to be between n^3 and n^5. Other factors do not change this order of complexity: determining the presence or absence of a required context, and whether or not a pair of segments forms a valid sequence can both be done in constant time with appropriate record-keeping; the time taken by top-down pruning is related only to the size of the matrix.

In space, the base theoretical complexity is O(n^2), determined by the size of the matrix to be filled. However, once again, the number of alternative segments to be stored in each cell should be considered, which would increase this complexity to O(n^3). Furthermore, the requirement to record the validity of each pair of possible segments increases this yet further to O(n^4). As for time, because of constraints on the number of alternative segments, a realistic estimate of space complexity is likely to be between n^2 and n^4.

These quantities are better than the factorial order outlined in section 4.2, but still too large to yield a useful algorithm. Section 6 briefly considers the development of different algorithms of lower complexity.

4.5 Limits

Initial results showed that unrestricted filling of the matrix of cells was practical only for extracts of music just a handful of segments long. Thus certain limits have been imposed to rule out reductions believed to be almost always ‘bad’ and keep the processing requirements of time and space within practical bounds, as follows.

1. Segments are limited to no more than four notes on the grounds that Schenkerian graphs usually show no more than four simultaneous lines.
2. ‘Complex’ harmonies (i.e., seventh harmonies other than the dominant seventh) are permitted only in contexts where there is no possible reduction which results in a simple harmony.
3. Pairing of very short with very long segments is restricted, especially if the short segment comes first. (It will be seen that empirical investigations confirmed that these were indeed apparent factors in the goodness and badness of reductions.) If the short segment comes first, it must have no less than a third of the duration of the longer one. If the short segment comes second, it must have no less than an eighth of the duration of the longer one.
4. The complexity of division of a time span is limited. If the ratio between two segments forming a sequence valid for reduction is expressed as a rational number

in its simplest form, the sum of the numerator and denominator must not exceed nine.

5. There is a limit on the syncopation inherent in derived segments. Each time point at the surface is associated with a given level of beat. The beginning of a bar is level 1, the next strongest beat or beats (if the division is into threes) within the bar are at level 2, etc., with each time division introducing a weaker level of beat. (The mechanism is explained more fully in Section 5.5 below.) A segment is syncopated if there exists a time point within its span which has a stronger level of beat than the level at the start of the segment. The limit on syncopation used here is that there can be no time point which is more than one level stronger than the start of the segment. (The empirical investigations reported below confirm that syncopation should, on balance, be avoided.)

6. Pairings of notes between segments which produce crossing voices are not permitted. I.e., if note *n1* in the first segment is paired with note *n2* in the second, then no note below (above) *n1* can be paired with a note above (below) *n2*. A similar restriction prevents crossing over a note paired with a rest: if *n1* is paired with a rest, then no note below (above) *n1* can be paired with a note in the other segment which is higher (lower) than *n1*.

7. The number of voices which can join at or split from a single note is restricted. A note *n1* can be paired with no more than two notes in the other segment. If it is paired with a rest, it cannot be paired with any other note.

4.6 Example of automatic reduction

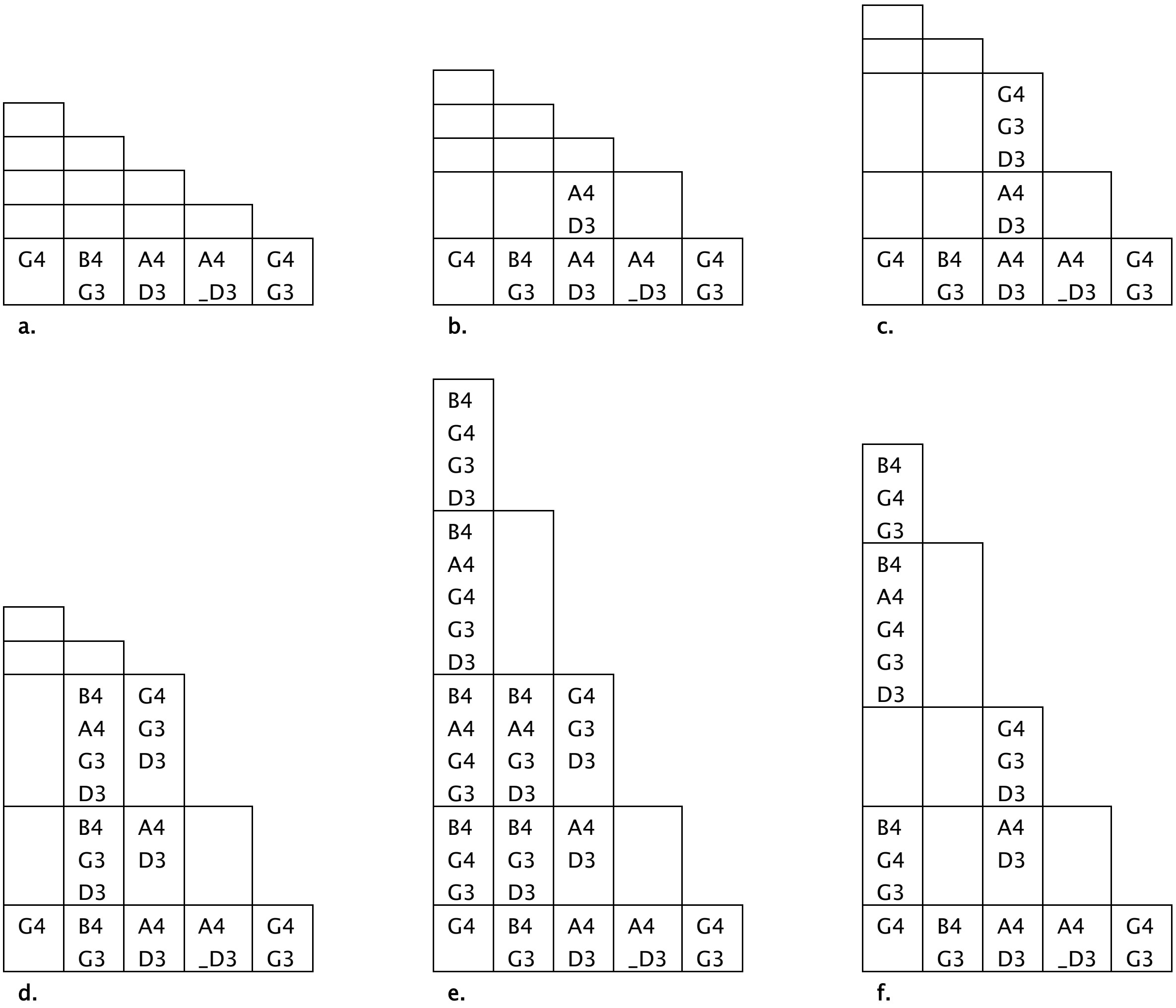

Figure 3. Illustration of reduction procedure.

The reduction procedure is illustrated in Figure 3. The music here is the same as in Figures 1 and 2. The bottom row of the charts **a–f** represents the segments of the surface of the piece. Each cell in each second row from the bottom contains segments which span two segments of the surface (the one below and the one below and to the right). Each cell in each third row contains segments which span three segments of the surface, etc.

Chart **a** represents the initial state, when only the surface segments exist. The first step is to reduce the last pair of segments to find segments to fill the last cell of the second row from the bottom. However, the resulting segments would exceed the syncopation limit. (The time point at their start is two levels weaker than the level at the start of the next bar, which would be included in the span of the resulting segments.) Therefore no segments are derived.

At the next step, the possible segments resulting from reducing the third and fourth surface segments (A4/D3 with A4/_D3) are placed in the second-to-last cell of the row second from the bottom. Only one segment is possible: one consisting of the notes A4 and D3, as shown in chart **b**.

Then the segments for the cell above, which covers the last three segments of the surface, are derived. The segments in this cell could be derived from two pairs of cells: the third cell of the surface with the last cell of the second-from-bottom row, and the third cell of the second-from-bottom row with the last cell of the surface. However, the last cell of the second-from-bottom row contains no segments, so only the second possibility is considered, which means that only one pair of segments needs to be considered since the third cell of the second-from-bottom row contains only one segment, like the last cell of the surface. This pair of segments (A4/D3 and G4/G3) produce three possible reductions: G4/G3/D3, G4/G3 and G4/D3, represented in chart **c** only by their constituent pitches.

In a similar manner, two cells of the second column are filled with three and four segments respectively, to produce the chart **d**. The top cell of this column remains empty because its segments would once again exceed the syncopation limit. Furthermore, one of the possible segments in the second-from-bottom cell (B4/G3) of this column is subsequently deleted because it turns out not to have any parents in later reductions. (This segment lacks D3, which is necessary to make a reduction with the fourth segment of the surface, since the only note which can form an atomic elaboration with the tied _D3 in that segment is another D3.)

Finally the cells in the first column are filled from the bottom up with seven, fifteen, sixteen and fifteen segments respectively. Once again some of the segments in the third row, first column are subsequently removed because they lack parents, leaving just nine segments in this cell. The result is illustrated in chart **e**.

Chart **f** shows the result of pruning the resulting set of reductions to leave only those which contain an *Ursatz*. Let it be stressed that the cells of the chart properly contain sets of segments, not sets of pitches as Figure 3 might imply. The notation used there is intended to be illustrative only, but readable. For example, it should not be presumed that the cell in the second row, first column of chart **f** contains segments made up of all combinations of the five pitches listed there. There are 31 such combinations. Some are excluded because they do not make valid harmonies or the seventh chords they imply would not be allowed. However, seventeen combinations of these pitches would imply simple harmonies, but the cell in the resulting chart contains just eight cells. (The example software referred to below shows more details, allowing a chart to be constructed, explored and manipulated.)

From the resulting chart **f**, it is possible to select a number of valid analyses, including the one shown in Figures 1 & 2.

5. Empirical Results and a 'Goodness' Metric

5.1 Software

The reduction procedure described above has been implemented in software, using the programming language Java, version 1.6. Example software is available for download at http://lancs.ac.uk/staff/marsdena/schenker. A demonstration applet is also provided. At present, musical input must be in a text file in a simple but specific format. It is intended to allow import from formats such as Humdrum kern (Huron, 1997) and MusicXML (Good, 2001) in future. The software at this stage is intended to facilitate further theoretical research rather than to provide a tool for making Schenkerian analyses.

The results reported below arose from using this software on a computer with 4GB of RAM and an Intel Core2 Duo processor running at 2.66GHz with the Ubuntu operating system 8.04.

5.2 Materials

The materials used in this investigation are four themes from movements in rondo form from piano sonatas by Mozart (the last movements of K.545, K.333, K.570, and the rondo K.494, which was published with the two movements K.533 to form a complete sonata) and two themes from variation movements (the first movement of the sonata K.331 and the last of K.284). All these themes form small but complete and self-contained musical units: in Schenkerian terms, they contain an *Ursatz*. The four rondo themes are ones I have used for many years in teaching Schenkerian analysis to undergraduate students, for the same reason of combination of compactness and completeness. As a consequence, there are pre-existing analyses for these themes in the model answers I have supplied to students. Furthermore, the theme from K.333 is used as an exercise in the textbook by Forte & Gilbert (1982a, p.140), who also provide a model answer (1982b, p.42). Forte & Gilbert and other authors also present analyses of the variation theme from K.331, while my colleague Neil Boynton has kindly allowed me to use an analysis he himself has made of the theme from K.284. These analyses form a 'ground truth' sufficiently sound for this initial investigation. Ideally the ground truth would consist entirely of analyses by someone other than myself—ideally they would be by Schenker—but it is difficult to find complete analyses of short but complete musical examples. At the outset of this investigation I had planned to use also the last-movement themes from K.330 and K.576, which I use in teaching alongside the other four, but the demands of computing time and space from these slightly longer themes were prohibitive.

Indeed, short as they are, only parts of these six themes have been used. The extracts analysed are taken from, or based on, just the second phrase of each of the themes. (All the themes have a first phrase which ends on the dominant and a second which starts like the first phrase but ends on the tonic.) The resulting six phrases are given in Figure 4. The example from K.570 is not exactly the second phrase of the Mozart theme; it omits the anacrusis and a grace note and replaces the penultimate note of the melody line. In the original this note is D5, a note which music theory would call an 'échappée' because it is dissonant with the dominant harmony but does not resolve down to the consonant C5. (Other formulations would call this an 'incomplete neighbour note'.) This kind of elaboration could have been included in the set defined in Table 1, but, because it is only loosely constrained, possible examples could then be found in many places, considerably increasing the already severe complexity of the reduction process. As an expedient to allow some results to be obtained, this note has been replaced by the Bb4 found in example 3, which is instead an ‘anticipation’. Similarly, the example from K.284 omits the anacrusis of two crotchet (quarter-note) A4s, and a middle-voice A3 is missed out in the last bar to ensure that no segment contains more than four notes.

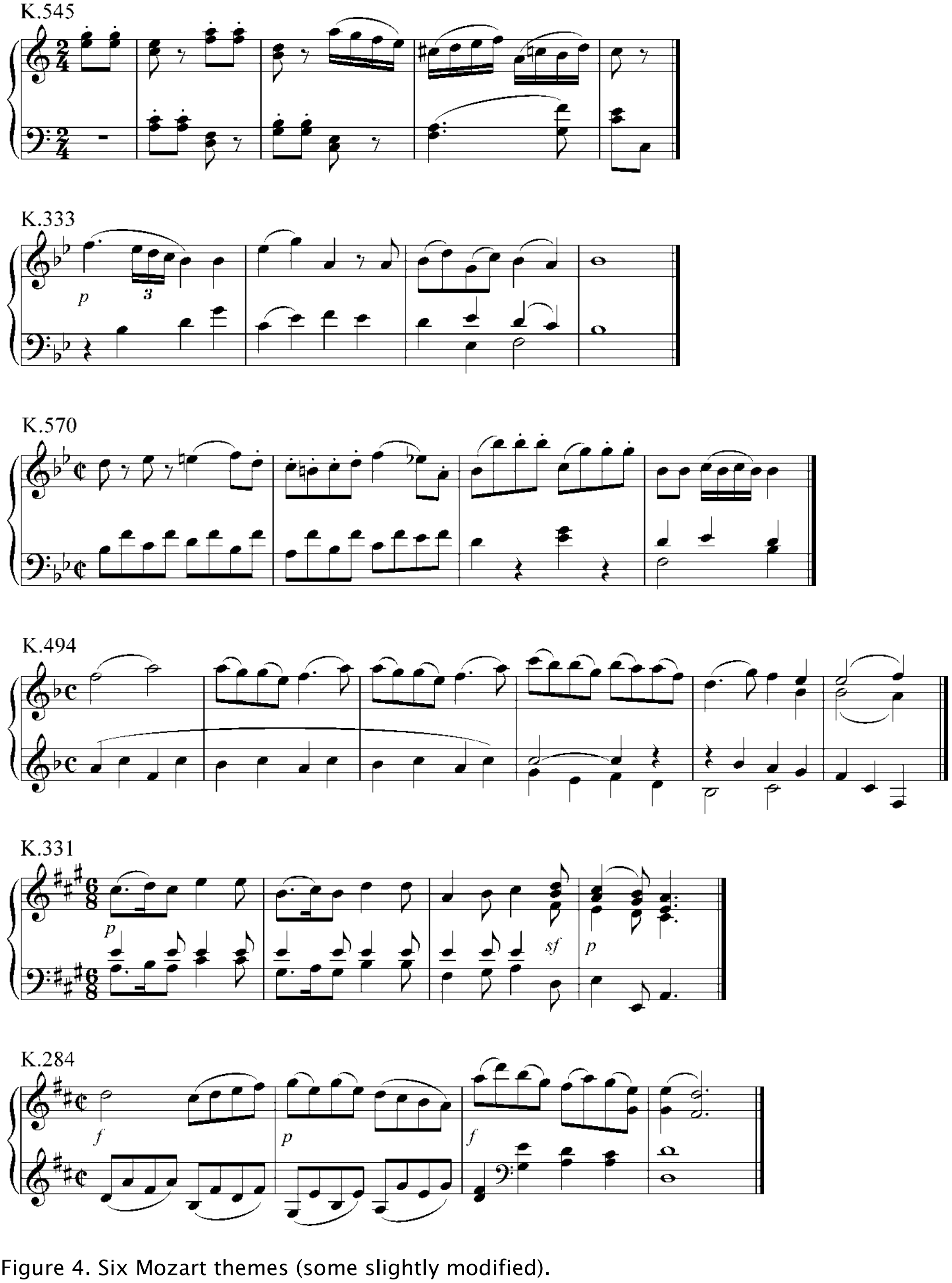

Figure 4. Six Mozart themes (some slightly modified).

Outline reductions were derived from the ‘ground-truth’ analyses of each of the examples. In a number of cases, the analyses could be interpreted in slightly different ways. There were two sources of ambiguity. First, since the rhythm of higher levels is not specified, occasionally the notes of a higher level could align with the surface in more than one way. Second, the analyses sometimes transposed bass notes by an octave to make a neater analysis, and the reduction could contain a note in one of two possible octaves. For each ambiguous case, a reduction was formed encompassing each possible interpretation (though the differences were very small). Thus there were often more possible 'ground-truth' reductions for an example than there were original analyses. As an example, Figure 5 shows the 'ground truth' outline reduction derived from Forte & Gilbert's analysis of the K.333 theme (1982b, p.42). The reduction of this theme derived from my teaching materials differs from this in some lower-level details (the bass crotchets (quarter notes) D4-G4 in the first bar (measure) are omitted and a quaver (eighth note) D5 is included) and in taking C4 rather than F4 as the main bass note of the second bar.

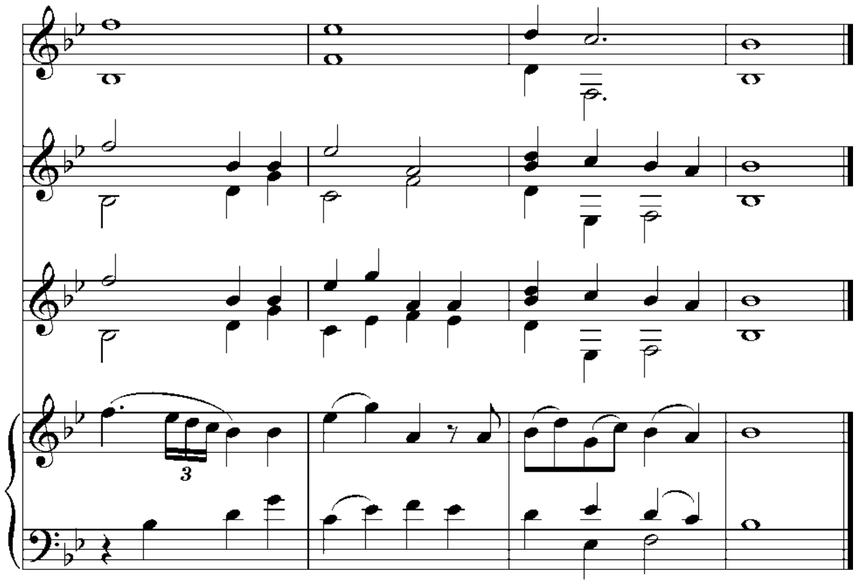

Figure 5. Outline reduction of the K.333 theme derived from Forte & Gilbert (1982b, p.42).

5.3 Method

The research method, in outline, was as follows.

1. The reduction chart was derived for each example, following the procedure described in Section 4.
2. A sample of 1000 analyses was selected from the chart, without regard as to whether or not they contained an *Ursatz*.
3. A second sample of 1000 analyses was selected, this time selecting only analyses which contain an *Ursatz*.
4. A third sample of 1000 analyses was selected, this time selecting only analyses which both contain an *Ursatz* and conform to a 'ground-truth' reduction. A sample was taken for each 'ground-truth' reduction. These analyses were taken to constitute 'good' analyses, since they conform to the ground truth.

5. A 'preferred' analysis was selected, containing an *Ursatz* and conforming to a 'ground-truth' reduction, which yielded the apparent best analysis, according to my experience of Schenkerian analysis and my musical judgement. Figure 6 shows the preferred analysis for the K.333 theme.

6. The results of steps 2–5 were measured according to a set of candidate goodness measures, and the means and standard deviations computed for each set.

7. A candidate 'goodness' metric was derived from those measures which distinguished the ideal analyses (step 5) and the good analyses (step 4) from the sample analyses with *Ursatz* (step 3).

8. To test the 'goodness' metric, for each example the reduction chart was derived once again using the new goodness metric, and the highest-scoring analysis was selected. (The segments derived in the reduction process were identical to those derived in step 1, but they now had meaningful scores attached.) If the procedure and goodness metric were correct, each selected analysis should both conform to a 'ground-truth' reduction and appear reasonable on the basis of musical judgement.

9. As a further test of the 'goodness' metric, step 8 was repeated with pruning. At each step, only the *n* best-scoring segments were kept, with *n*, the 'pruning limit', ranging from 1 to 100. If the goodness metric were effective, a good analysis should result from a low pruning index in each case.

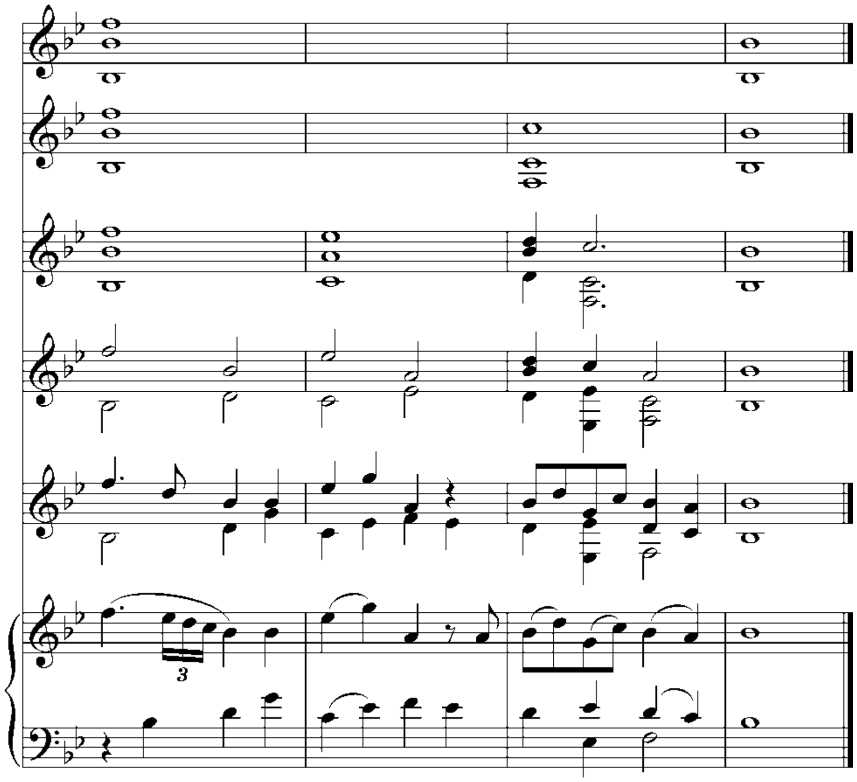

Figure 6. Preferred analysis of the K.333 theme.

The test procedures used here are not really adequate: tests should be made on material different from that used to derive the goodness metric to avoid the error of over-fitting. Ideally, more extensive testing would be carried out on different materials, but to be practical this will have to await collection of a suite of suitable test materials and development of more efficient algorithms. The procedures outlined here have taken about two weeks of processing time to complete.

5.4 Sampling

For this procedure to be at all valid, the samples taken in steps 2-4 should fairly reflect the distribution of their sets. Naively selecting child segments at random will not result in a fair sample, because the number of possible sub-trees below one child might be very different from those below another. Selecting children with equal probability will result in a higher likelihood of selecting analyses which exist among fewer alternatives. Ensuring that each different analysis has an equal likelihood of selection requires weighting the selection of children by the multiplicity of the sub-trees below them. This quantity is recorded as the reduction chart is filled as follows. The multiplicity at a surface segment is 1. The multiplicity at segments above the surface is the sum of the products of the multiplicities of each pair of child segments. This is the total number of possible sub-trees below that segment, ignoring any context constraints, and is therefore an upper bound on the number of valid sub-trees. As selection proceeds, the non-selected segments are removed. Elaborations and segments which have become invalid because their required contexts have been removed are then also removed, and the procedure repeated recursively until all consequential changes have been made. The multiplicity for segments is then recalculated to compute the new (lower) upper bound. Selection is thus not guaranteed to be completely fair, but the difference between the upper bound on multiplicity of sub-trees and the actual number of valid sub-trees is relatively small, and any unfairness which remains in the selection procedure must be the result of variation in this difference. I know of no reason why this variation should be systematic and so believe that the selection is likely to be sufficiently fair.

A by-product of this selection procedure is a means of estimating the total number of valid analyses in each set. The reduction in multiplicity after recalculation following the selection of a pair of child segments can be measured at each stage. Assuming that this reduction does not vary systematically from one part of the chart to another, the average reduction can then be applied to the sum of the multiplicities of the top-level segments to arrive at an estimate of the total number of valid analyses.

Table 2 gives these estimated sizes, the computation times and other data for each of the six examples. The reduction chart for the K.545 theme contained no analysis which conformed to the ground truths, so the sets at steps 4 and 5 were empty. This theme is unusual in that it does not start with a root position tonic (and indeed it is the one students find most difficult!); both my sample answer and that of Forte & Gilbert (1982b, p.44) take a bass C to be implied at the beginning, a step not unusual in Schenkerian analysis, but not one accommodated in the theory as formalised here. Peculiarities in this theme probably also underlie another anomaly: that the estimated number of possible reductions with an

Ursatz is greater than the estimated total number of possible reductions, which is clearly wrong. Something about this theme caused a severe over-estimate of the factor by which the raw multiplicity should be reduced when estimating the total number of possible reductions, a much greater reduction than in the case of the other themes.

As indicated in section 5.2 above, I had originally intended to use two other themes also. However, even when these were simplified by making a composite phrase which joined the opening few bars of the first phrase to the conclusion of the second phrase, the result was too large to deal with. A reduction chart could be derived successfully (though only after hours of processing), but the amount of memory taken was so great as to make any further processing with this chart impractical, especially since the selection procedure is of exponential order of complexity in time. Table 2 includes figures for the time and space taken to derive the reduction chart for each theme. These do conform to the hypothesised computational complexities of between n^3 and n^5 for time and n^2 and n^4 for space, but of course there is insufficient data to confirm this. The number of notes in a piece appears to be a better predictor of time and space complexity than the number of segments, and suggests complexities of $n^{4.5}$ and $n^{3.2}$ for time and space respectively.

Theme	**Number of segments**	**Number of notes**	**Computation time (sec)**	**Computation space (MB)**	**Total number of possible analyses**	**Number of possible analyses with *Ursatz***	**Conformance to 'ground truth' of best-scoring analysis**
K.545	22	58	138	141	6.3e12	9.2e12	–
K.333	19	40	125	88	7.5e11	1.0e11	79%
K.570	31	65	543	277	4.5e22	5.8e20	83%
K.494	34	82	2512	880	1.0e25	1.0e24	90%
K.331	17	55	248	230	1.8e15	8.3e14	91%
K.284	26	67	1524	510	1.3e21	1.2e20	98%

Table 2. Computation results.

The striking thing about the figures in Table 2 is how many possible analyses there appear to be for even short and simple themes. (For comparison, the number of stars in the visible universe is generally estimated to be of the order of 10^{22}.) As Temperley suggests (2007, p.176), the rules of Schenkerian analysis seem to be much too loose to specify an analysis with any degree of usefulness. Clearly other factors are at work in Schenkerian analysis, which are probed in the following sections.

5.5 Candidate goodness measures

Schenkerian analysts have long recognised that alternative analyses of the same piece are possible, and the literature does include some discussion of the factors used to decide which is best among alternative analyses. However, there is no accepted set of principles which distinguish a good analysis from a bad one. Candidate goodness measures could be found in the ‘indices’ proposed by Plum (1988) and in a famous article by Schachter (1990). Other good sources for candidate goodness measures are the preference rules of Lerdahl & Jackendoff (1983), Larson’s ‘musical forces’ (Larson & Vanhandel, 2005), and Schenkerian instructional literature (e.g., Forte & Gilbert, 1982a; Cadwallader & Gagné 2007; Pankhurst, 2009). However, for this initial proof of concept, measures have been designed on a more intuitive basis of what is simple to compute and is presumed to be potentially fruitful. In principle, the same procedures could be applied to test measures derived from these sources.

The fourteen measure used were as follows.

1. **duration ratio**: the duration of the shorter child divided by the duration of the longer.
2. **short-long**: 1 if the duration of the first child is less than the duration of the second; 0 otherwise.
3. **syncopations**: if a beat stronger than the beat at the start of the segment occurs before the end of the segment, the difference in level of strength between the strongest such beat and the beat at the start of the segment; otherwise 0.
4. **harmonic simplicity**: 1 if the harmony is (consonant with) I or V^7; otherwise 0.
5. **root-position**: 1 if the lowest pitch of the segment can be the root of the harmony; otherwise 0.
6. **second-inversion**: 1 if the lowest pitch of the segment can only be the fifth of the harmony; otherwise 0.
7. **harmonic support**: proportion of the surface in the span of a segment which is consonant with the harmony of that segment (measured in time).
8. **pitch support**: proportion of the surface in the span of a segment which contains the pitches of that segment, averaged for each pitch of the segment.
9. **interval** in semitones between the paired notes of the child segments (i.e., the pairs of notes which participate in atomic elaborations), averaged per pair.
10. **voice split/join**: the number of notes in child segments which are paired with more than one note in the other segment.
11. **delay**: the number of atomic elaborations which start with a rest.
12. **shortening**: the number of atomic elaborations which finish with a rest.
13. **post-context from ancestor**: the number of levels of reduction between the lowest common ancestor and the required context (if any).
14. **post-context from surface**: the number of levels of reduction between the surface and the required context (if any).

A value for each of these measures was computed for each segment. Since measurements were only taken for complete analyses, derived by selecting children from the reduction matrix, each segment which was not a surface segment had just one pair of children. However, there might have been more than one possible harmony for a segment arising from more than one possible set of atomic reductions. In such cases, the measure computed was the presumed best among the possible harmonies and sets of atomic reductions. This was the highest for measures 4, 5, 7, and 8, and the lowest for measures 6, 9, 10, 11, 12, 13, and 14. For surface segments, those measures which depend on children (1, 2, 9, 11, 10, 11, 12) took a value of 0. For other segments, the recorded value for each measure was the sum of the recorded values for its children plus the value computed for the segment itself. The measure for an entire analysis was then the recorded measure for the top-level segment divided by the number of segments in the analysis.

The syncopation measure was computed as follows. As mentioned above, a metre for each extract is specified in advance which assigns to each point in time a strength of beat, represented as an integer, with lower numbers representing strong beats. The strength of the beat at the beginning of each bar (measure) is 1 and all beats within the bar are weaker. For each metre, there is a specified pattern of beats, such as 1 3 2 3 for 4/4. The strengths for time points between these specified beats is computed according to the following procedure.

```
Let $b1$ and $b2$ be the beats preceding and following the time point $t$ respectively.
Let $s$ be the strength of the weaker of beats $b1$ and $b2$.
Let $x$ be the interval between $b1$ and $b2$ and $y$ be the interval between $b1$ and $t$, both expressed as positive integers.
Until $y$ is a multiple of $x$,
    divide $x$ by the smallest prime number less than $x$ and set $x$ to the result,
    add 1 to $s$.
The strength of the beat at $t$ is $s$.
```

5.6 Sampling results

Table 3 gives the mean value for each of these measures for the sets of results from each of the themes. For each theme, the row 'All' gives the values for the sample of all possible reductions (step 2 of the procedure outlined above), the row 'With *Ursatz*' from the sample of reductions with an *Ursatz* (step 3), the row 'Conforming' from the samples of reductions conforming to a 'ground truth' (step 4), and the row 'Preferred' from the single most preferred reduction in each case (step 5). The row 'Std. dev.' gives the standard deviations of the sample of measure from all possible reductions (step 2). (The standard deviations from the *Ursatz* samples (step 3) were not significantly different.)

	duration ratio	short–long	syncopation	harmonic simplicity	root–position	second–inversion	harmonic support	pitch support	voice split/join	interval	delay	shortening	post–context from parent	post–context from surface
K.333														
Std. dev.	*0.39*	*0.07*	*0.04*	*0.10*	*0.16*	*0.08*	*0.04*	*0.03*	*0.07*	*0.42*	*0.14*	*0.15*	*0.31*	*0.28*
All	2.17	0.17	0.07	0.76	0.63	0.07	0.69	0.47	0.15	2.05	0.42	0.36	0.31	0.46
With *Ursatz*	2.23	0.17	0.06	0.82	0.75	0.01	0.70	0.46	0.16	1.99	0.42	0.38	0.30	0.54
Conforming	1.70	0.13	0.03	0.88	0.81	0.00	0.74	0.51	0.14	1.76	0.36	0.38	0.19	0.60
Preferred	1.56	0.11	0.03	0.89	0.91	0.00	0.74	0.54	0.06	1.16	0.28	0.28	0.33	0.50
K.570														
Std. dev.	*0.24*	*0.08*	*0.04*	*0.03*	*0.12*	*0.07*	*0.03*	*0.02*	*0.07*	*0.44*	*0.10*	*0.10*	*0.26*	*0.21*
All	2.12	0.28	0.14	0.85	0.68	0.08	0.67	0.50	0.21	2.58	0.36	0.43	0.36	0.25
With *Ursatz*	2.21	0.26	0.14	0.85	0.74	0.04	0.67	0.48	0.17	2.40	0.33	0.40	0.30	0.27
Conforming	1.95	0.27	0.11	0.87	0.66	0.06	0.70	0.48	0.14	2.05	0.32	0.41	0.16	0.24
Preferred	1.40	0.03	0.00	0.90	0.53	0.13	0.71	0.52	0.00	1.17	0.27	0.37	0.14	0.29
K.494														
Std. dev.	*0.24*	*0.07*	*0.04*	*0.07*	*0.13*	*0.06*	*0.03*	*0.02*	*0.06*	*0.32*	*0.10*	*0.09*	*0.24*	*0.21*
All	2.04	0.25	0.11	0.87	0.57	0.05	0.72	0.51	0.19	1.82	0.46	0.39	0.29	0.36
With *Ursatz*	1.99	0.24	0.10	0.89	0.65	0.02	0.73	0.52	0.19	1.77	0.47	0.40	0.22	0.47
Conforming	1.92	0.25	0.08	0.95	0.63	0.00	0.78	0.53	0.15	1.45	0.45	0.41	0.08	0.57
Preferred	1.30	0.03	0.00	0.94	0.55	0.00	0.74	0.58	0.00	1.13	0.21	0.18	0.13	0.63
K.331														
Std. dev.	*0.32*	*0.09*	*0.04*	*0.06*	*0.20*	*0.10*	*0.05*	*0.04*	*0.12*	*0.50*	*0.13*	*0.10*	*0.18*	*0.17*
All	2.83	0.29	0.10	0.92	0.54	0.08	0.66	0.53	0.45	2.86	0.42	0.22	0.21	0.09
With *Ursatz*	2.83	0.28	0.10	0.92	0.66	0.02	0.67	0.53	0.44	2.85	0.42	0.24	0.19	0.08
Conforming	2.11	0.16	0.06	0.95	0.70	0.02	0.70	0.62	0.29	1.98	0.28	0.20	0.42	0.18
Preferred	2.44	0.00	0.00	1.00	0.71	0.00	0.70	0.65	0.06	0.99	0.13	0.19	0.17	0.67
K.284														
Std. dev.	*0.28*	*0.07*	*0.04*	*0.05*	*0.14*	*0.07*	*0.03*	*0.02*	*0.07*	*0.44*	*0.11*	*0.11*	*0.25*	*0.28*
All	2.10	0.24	0.12	0.79	0.71	0.06	0.68	0.46	0.20	2.56	0.45	0.39	0.25	0.36
With *Ursatz*	2.19	0.25	0.12	0.78	0.74	0.04	0.68	0.46	0.20	2.39	0.43	0.36	0.25	0.38
Conforming	1.90	0.19	0.09	0.80	0.84	0.02	0.73	0.49	0.18	2.16	0.41	0.37	0.33	0.34
Preferred	1.08	0.04	0.02	0.80	0.93	0.00	0.72	0.51	0.00	1.48	0.44	0.32	0.75	0.50
Weight	**−2.16**	**−1.93**	**−2.05**	**0.75**			**0.37**	**2.72**	**−2.39**	**−2.35**		**−0.54**		

Table 3. Sampling results.

The objective of the sampling and measuring exercise was to discover factors which distinguish the ‘good’ analyses (those which conform to a ground truth, and the preferred analysis) from other possible analyses with an *Ursatz*. Thus in table 3, every case of a 'Conforming' or 'Preferred' measure which differs from the 'With *Ursatz*' measure by more than one standard deviation is underlined. Columns corresponding to measures for which the difference is always positive or always negative (if it is not negligible) are shaded. These are measure which distinguish 'good' analyses from others in a consistent manner (if at all). In four cases (duration ratio, pitch support, voice split/join, and interval) every one of the preferred reductions is distinguished by a difference greater than one standard deviation, and in the case of duration ratio and pitch support the difference for the sample of 'conforming' reductions is also greater than one standard deviation for the majority of themes. In the case of three other measures, (short-long, syncopation and delay) a majority of the preferred analyses differ by more than one standard deviation from the *Ursatz* sample, but in the case of the delay measure, one also differs (but by only a small amount) in the opposite direction. This measure is therefore not considered a good candidate. Three other measures (harmonic simplicity, harmonic support, and shortening) show consistent differences, but most are small. The bottom row of the table gives the harmonic mean of the differences between the measures for the preferred reduction and the *Ursatz* sample in these nine cases, expressed in standard deviations. (The harmonic mean is used because it results in lower values for distributions with greater variance but the same mean. Thus high values result from consistently large differences.) From this we can propose nine principles for making good analyses, in order of decreasing strength.

1. Select higher level pitches which are more often present in the surface.
2. Avoid splitting and joining of voices.
3. Select reductions with small intervals between notes reduced together.
4. Reduce segments of approximately equal duration together.
5. Avoid reductions which create syncopations at higher levels.
6. Avoid reducing a shorter segment with a following longer segment.
7. Prefer reductions with more tonic and dominant harmony.
8. Avoid reductions where a note is followed by a rest.
9. Prefer reductions where higher level harmonies are more often consonant with the surface.

From the product of the harmonic means and the standard deviations, a weight can be computed to be used in combining each measure into a composite 'goodness metric'. For simplicity, simple linear combination was used (i.e., each measure is multiplied by the appropriate weight and the results added together). Further research should determine whether or not there is a more appropriate method of combination.

5.7 Selection of best-scoring analysis

The scoring metric could now be used in a procedure which selected the best-scoring analysis from a reduction chart in order to complete the process of automatically deriving an analysis from the notes of the surface (step 8 of the process outlined in section 5.3 above). First, reduction charts were recomputed for each example using the goodness metric proposed above. Most segments above the surface of the chart were parents of a number of possible elaborations, sometimes with the same pair of children and sometimes with different pairs. The score recorded for a segment was the best among the various possibilities. The best-scoring analysis could then be selected by selecting the highest-scoring top-level segment which had a complete *Ursatz* among its recorded *Ursatz* possibilities. Children were selected in a recursive procedure which, at each level, selected the highest-scoring pair of children which yielded this *Ursatz*. (It was here that it was vital to have recorded the *Ursatz* possibilities of each segment.) However, in many cases the recorded best score for a segment arose from an elaboration or pair of children which was no longer valid, either because a required context had been removed as a result of selection of segments in another part of the matrix, or because they did not conform to the required *Ursatz*. Thus often no complete analysis existed for the selected top-level segment which yielded the score recorded for that segment, and similarly for segments at lower levels of analysis.

To ensure that the highest-scoring valid analysis with an *Ursatz* was found, a best-first search procedure was used which can be described as follows. The procedure depends on a data structure which can be called a 'partial analysis' which has a score and keeps a record of the segments selected so far and of information for updating the score of that partial analysis. An agenda of partial analyses is maintained which lists the partial analyses derived to date in order of decreasing score. Initially the agenda consists essentially of the partial analyses based on each possible top-level segment. At each step the best-scoring partial analysis is removed from the agenda and extended by selecting children for some non-surface segment. This generally results in several new partial analyses which are inserted into the agenda according to their score. The score is essentially an upper bound on the score of any analysis selected from this partial analysis. When children are selected, this upper bound can often be reduced, so the score for the resulting partial analysis is reduced accordingly. (In fact, the procedure is more complex than this. Selection of children takes place in several steps, first selecting the pair of cells in which to find children, and so on, updating the score at each step. Furthermore, no more than two partial analyses result from each extension: the first selects the best-scoring children, the second records which choices remain. This results in a more efficient procedure, but the effect is the same.) The first analysis found to be the best-scoring partial analysis in the agenda which cannot be further extended is an analysis with the highest possible score. Other analyses with equal scores might exist, but none can have a higher score. (The same procedure can be continued to find these other high-scoring analyses.) This procedure is unfortunately of factorial complexity in the worst case, but the search strategy reduces the likelihood of the worst case arising.

In no case did a top-scoring analysis derived by this procedure exactly match a preferred analysis. On the other hand, the results did generally conform quite well to the ground truths derived from pre-existing analyses done by hand. In attempting to quantify the accuracy of his foreground-deriving software, Kirlin used a typical precision/recall measure, but acknowledged that this was problematic because the information given in the published analyses used as ground truth and in his computed foregrounds was not equivalent (2009, pp.426-427). In particular, because he was concerned only with top and bass voices, the published analyses generally contained more notes than his foregrounds. I had the opposite problem: the computed analyses accounted for all voices and every intermediate reduction, whereas the ‘ground truths’ typically concerned only the top and bass voices and left many intermediate reduction steps implied. Thus it was to be expected that the ground truths would contain fewer notes than the computed analyses. An appropriate test would check the inclusion of notes from the ground truths in the results but not *vice versa*. Thus the measure used here was simply to count the proportion of notes in the ground truth reductions which occurred at the same time and with the same duration in the automatically computed analyses. As pointed out above, there was generally more than one ground truth for each theme; the figure used was the best match, on the basis that each ground truth was equally valid and the system should be considered to be successful if its output matched any of the ground truths.

This was nevertheless quite a severe test, since where there were deviations the automatically derived analysis often showed some similarity with the ground truth (for example by having the same pitches but in a different rhythm) but this similarity was not taken into account in the test. The match with the ground truth was therefore generally better than suggested by the figures given in Table 2, but these figures are nevertheless encouraging. As an example, Figure 7 shows the best-scoring analysis for the K.333 theme, which was the worst match with the ground truth among the best-scoring analyses for the themes tested. (The best-matching ground truth for this theme is shown in Figure 5. A better analysis of this theme, derived using pruning, is shown in Figure 8.)

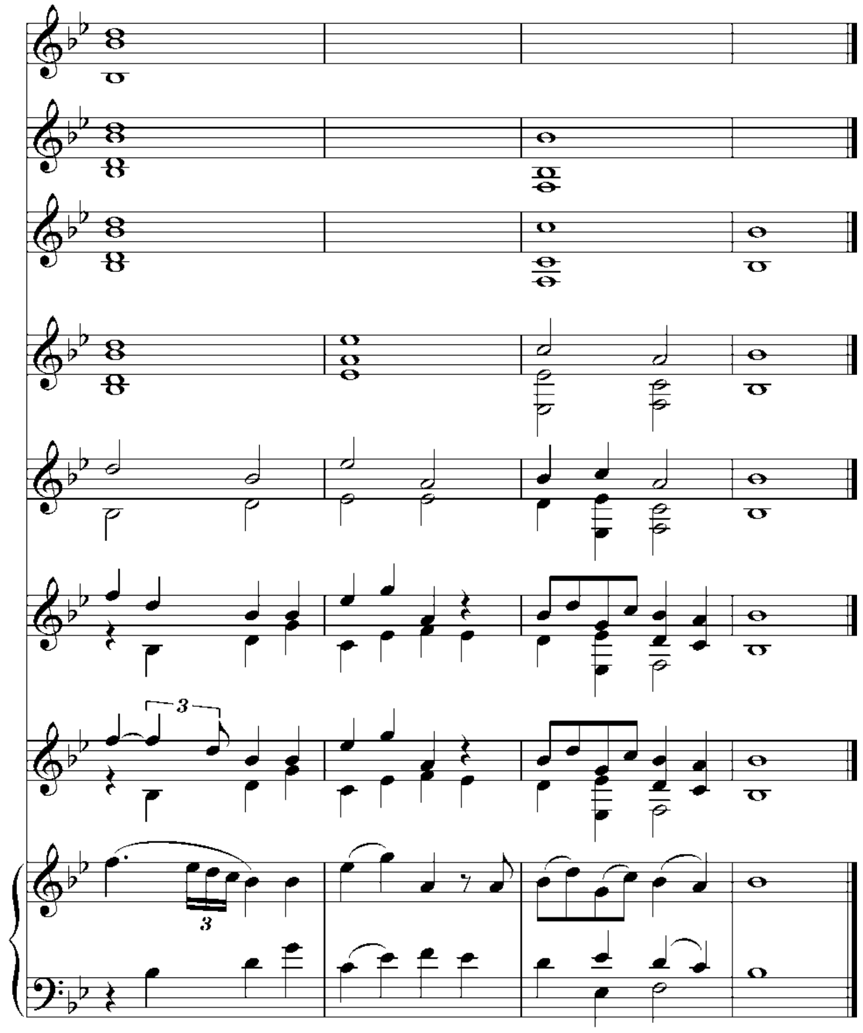

Figure 7. Automatically derived best-scoring analysis of the K.333 theme.

5.8 Pruning

As indicated above, the reduction procedure is extremely expensive in terms of computing time and space. Because the complexity arises from combination, reducing the number of segments to be combined at an early stage has a disproportionate impact on the overall computing cost. It would thus be beneficial to discard low-scoring segments in the course of the reduction procedure if this can be done with confidence that a good analysis is not thereby jeopardised. To test this, the reduction procedure was rerun on the five examples which produced analyses conforming to the ground truth, with one simple modification. This was to limit the number of segments retained in a cell after each reduction step, and discard lower-scoring segments beyond that limit. The limit ranged from 1 to 100, and the resulting best-scoring analyses were tested for conformity with the ground truths in the same manner as described in section 5.7 above. The resulting best scores and corresponding match with the ground truth reductions are shown in Table 4. Figures of 0% for the score and match indicate that the system failed to derive an analysis at all with the given pruning limit. (These tables record only the results where there is a change in score or match as the pruning limit increased. Thus for the K.333 theme, no analyses were derived until the pruning limit reached 10. Then the best score and match remained unchanged until the pruning limit reached 78.)

In no case was an analysis derived by keeping only the best-scoring segment at each reduction step (i.e., with a pruning limit of 1). In every case, the best-scoring analysis derived without any pruning (as in section 5.7 above) was derived with a pruning limit of 79 or less. In four of the five cases, the best score was already reached at a pruning limit of 35. It was only for K.333 that the much larger pruning limit was necessary to achieve the maximum score, and in this case the match with the ground truth is seen to be markedly *worse* than with a much lower pruning limit of 10, even though the score is higher. A higher score (as measured by the metric derived in section 5.6 above) does not therefore necessarily mean a better analysis. For comparison with Figure 7, Figure 8 shows the best-scoring analysis derived with a pruning limit of 10.

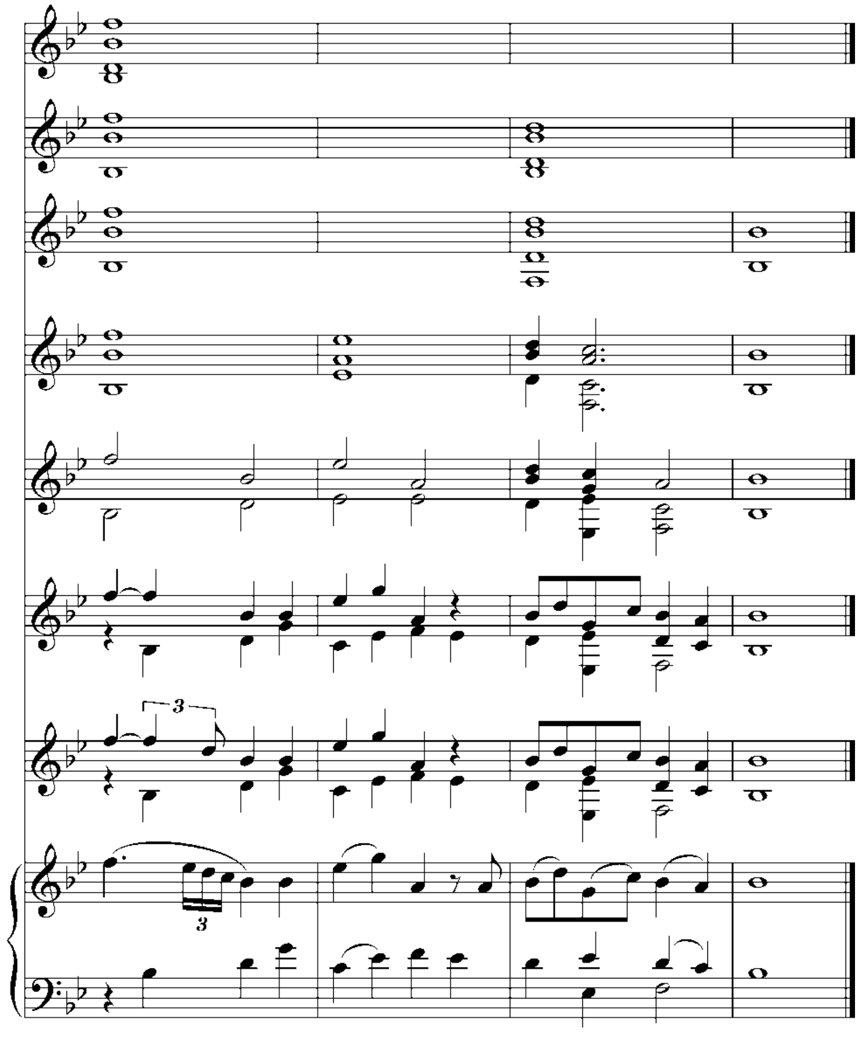

Figure 8. Automatically derived best-scoring analysis of the K.333 theme with a pruning limit of 10.

In two other cases we also see a reduction in the match with the ground truth with the last increases in the pruning limit. In fact, in the case of K.284, there is a perfect match between one of the ground truths for this theme and the best-scoring derived analysis with a pruning limit in the range 20 to 32. Although this decrease in the ‘quality’ of the analyses with less pruning is unexpected, it is actually an encouraging result. It suggests that the factors which determine a good analysis are, to some degree at least, local, since the pruning decision is taken only on the basis of local information. It should be possible, therefore, to find a better pruning mechanism which makes better use of this local information. This also suggests that a reduction procedure based on search (see below) has some chance of success. For the present, it appears that a simple pruning limit of about 30 is likely to produce a good analysis, but cannot guarantee the best analysis.

K.333		
Limit	Score	Match
1	0	0%
10	856	93%
78	879	79%
79	886	79%

K.570		
Limit	Score	Match
1	0	0%
11	1233	66%
13	1263	63%
16	1659	83%
27	1661	83%

K.494		
Limit	Score	Match
1	0	0%
5	1753	59%
6	1781	64%
7	0	0%
8	1599	51%
9	1629	54%
10	0	0%
12	1649	44%
13	1639	44%
15	1648	44%
16	1658	44%
20	1677	59%
21	1693	59%
23	1902	67%
25	1961	90%
28	1962	90%
35	1973	90%

K.331		
Limit	Score	Match
1	0	0%
2	828	84%
3	873	88%
4	932	98%
7	944	98%
9	948	91%

K.284		
Limit	Score	Match
1	0	0%
6	1012	91%
8	1024	91%
9	1025	95%
12	1032	93%
17	1035	93%
20	1169	100%
22	1212	100%
29	1211	100%
33	1213	98%
35	1214	98%

Table 4. Score and match with a ground-truth outline reduction in relation to pruning limit.

6. Discussion and Future Work

6.1 Outcome

The essential outcome of the research described here is a computational procedure capable of deriving Schenkerian-like reductions from the pitch and time information in a score. The reductions do match, to some considerable degree, the analyses that expert analysts make. There are two very significant caveats, however. Firstly, the procedure is extremely expensive in terms of computation time and space, and only short phrases can be processed with realistic resources. The pruning results suggest that savings can be made with further research, but even with a low pruning limit it has not been possible to derive complete analyses from the two other themes which I had planned to use in this project. The reduction chart can be derived in reasonable time, but the procedure to select from it the best-scoring analysis runs out of heap space. As stated above, while the chart-reduction process is of polynomial order, this step is of factorial order in the worst case. A practical analysis system requires a different order of procedure.

Secondly, the quality of analyses produced has not been adequately tested, so the claim that the procedure produces Schenkerian-like reductions must be regarded as provisional. The same sources were used both in the derivation of goodness measures and in testing of those measures, which will tend to over-fitting. Tests on different material are required, but will have to await collection of suitable ground truths. As indicated above, analyses of sufficiently short complete themes are not readily available.

6.2 Greater computational efficiency

Part of the difficulty in testing is the sheer quantity of computation time taken at present. A more efficient procedure is urgently required. As stated above, the main difficulty in the current procedure is the mechanism to select the best-scoring analysis from the reduction chart. I suspect that one problem here is that scores are not associated with *Ursatz* possibilities (i.e., the best score for a segment is recorded, regardless of whether or not the score can arise with a set of children which conform to a particular segment of an *Ursatz*). Thus a great deal of computation time is probably wasted pursuing high-scoring possibilities which will never fit with other parts of the analysis. On the other hand, associating scores with *Ursatz* possibilities will considerably increase the quantity of data to be recorded and so lead to greater inefficiency in another part of the process. This kind of trade-off should be a topic of further research.

The scoring and pruning mechanisms used here have been extremely simple. It is entirely possible that the weights for the different measures should not apply equally in all parts of an analysis (some might weigh more heavily than others at higher levels, for example), and also possible that something other than linear combination gives a better result. The pruning mechanism currently applies only to segments within a single cell, and does not take account of how alternative segments in different cells compete with each other. Considerable efficiency savings are therefore likely in these areas, particularly in view of the hints that local information is more important than global.

Fundamentally, though, I suspect that an entirely different procedure is required, one which has a lower order of complexity as a result of its essential process rather than as a result of incidental savings. Kirlin & Utgoff (2008) have advocated intelligent search as a solution to this problem, and Geraint Wiggins and I have already had some small-scale success in implementing a search procedure (Marsden & Wiggins, 2008). Such a procedure will almost certainly not be guaranteed to produce the 'best' result, but to do so is probably unrealistic for this particular problem. Even human experts produce a range of possible results, and then spend ages and pages of print in discussing their relative merits, so a 'best' result is probably a fantasy.

6.3 Further empirical work

Given more suitable materials, further empirical work to test or refine the results given here would be quite easy. As indicated above, the same procedures could also be used to make empirical tests of the work of some music theorists (Plum, Lerdahl & Jackendoff, Larson, etc.), though to translate their work into empirically testable measures might not be simple.

The really significant empirical work, however, would not depend on published analyses or music theory but on musical *behaviour*. Schenkerian theory has been subject to powerful criticism (e.g., Narmour (1977)) and many doubt that it says anything of use or value about music. Some have attempted to test whether reduction does play some part in listening to or remembering music (e.g., Oura & Hatano, 1991) but there is little work of this kind. Perhaps this is because there has been little to test with, for the same reason that material for this study has been hard to come by: complete analyses of short complete segments are rare. Possibly a mechanism which produces reductions, such as described here, will facilitate such work, allowing the reduction mechanism itself to be refined.

The real test of an automatic reduction procedure, I suspect, will be whether or not it proves to be useful when embedded in other music-processing systems. Will it, for example, allow better searches for music similar to an example? If automatic reduction allows better processing in systems which aim to handle musical data in a musically intelligent manner, then it will have proven its value.

References

Barbar, K., Desainte-Catherine, M., & Miniussi, A. (1993). The semantics of musical hierarchies. *Computer Music Journal*, *17*, 30–37.

Baroni, M. (1983). The concept of musical grammar (translated by S. Maguire with the assistance of W. Drabkin). *Music Analysis*, *2*, 175–208.

Baroni, M., Dalmonte, R., & Jacobini, C. (1992). Theory and analysis of European melody. In A.Marsden & A. Pople (eds.) *Computer Representations and Models in Music*, London: Academic Press, 187–206.

Bernstein, L. (1976). *The Unanswered Question: Six Talks at Harvard*. Cambridge, MA: Harvard University Press.

Brown, M. (2005). *Explaining Tonality: Schenkerian Theory and Beyond*. Rochester NY: University of Rochester Press.

Cadwallader, A., & Gagné, D. (2007). *Analysis of Tonal Music* (2nd edition). Oxford: Oxford University Press.

Desainte-Catherine, M., & Barbar, K. (1994). Using attribute grammars to find solutions to musical equational programs. *ACM SIGPLAN Notices*, *29*, 56-63.

Ebcioğlu, K. (1987). *Report on the Choral Project: An Expert System for Harmonizing Four-part Chorales*. Technical report RC 12628, IBM, Thomas J. Watson Research Centre, Yorktown Heights, New York.

Ebcioğlu, K. (1988). An expert system for harmonizing four-part chorales.*Computer Music Journal*, *12*, 43-51.

Ebcioğlu, K. (1990). An expert system for harmonizing chorales in the style of J.S. Bach. *Journal of Logic Programming*, *8*, 145-185.

Forte, A. & Gilbert, S.E. (1982a). *Introduction to Schenkerian Analysis*. New York: Norton.

Forte, A. & Gilbert, S.E. (1982b). *Instructor's Manual for 'Introduction to Schenkerian Analysis'*. New York: Norton.

Frankel, R.E., Rosenschein, S.J. & Smoliar, S.W. (1976). Schenker's theory of tonal music—its explication through computational processes. *International Journal of Man-Machine Studies*, *10*, 121-138.

Frankel, R.E., Rosenschein, S.J. & Smoliar, S.W. (1978). A LISP-based system for the study of Schenkerian analysis. *Computers and the Humanities*, *10*, 21-32.

Gilbert, E., & Conklin, D. (2007). A probabilistic context-free grammar for melodic reduction. *Proceedings of the International Workshop on Artificial Intelligence and Music, 20th International Joint Conference on Artificial Intelligence (IJCAI)*. Hyderabad, India, 83-94.

Good, M. (2001). MusicXML for notation and analysis. In W.B. Hewlett & E. Selfridge-Field (eds.) *The Virtual Score: Representation, Retrieval, Restoration (Computing in Musicology, 12)*, Cambridge MA: MIT Press, 113-124.

Hamanaka, M., Hirata, K., & Tojo, S. (2006). Implementing "A Generative Theory of Tonal Music". *Journal of New Music Research*, *35*, 249-277.

Hamanaka, M., Hirata, K., & Tojo, S. (2007). FATTA: Full automatic time-span tree analyzer. *Proceedings of the International Computer Music Conference (ICMC)*, Copenhagen, August 2007, 153-156.

Huron, D. (1997). Humdrum and Kern: selective feature encoding. In E. Selfridge-Field (ed.) *Beyond MIDI: The Handbook of Musical Codes*, Cambridge MA: MIT Press, 1997, 375-401.

Jurafsky, D., & Martin, J.H. (2009). *Speech and Natural Language Processing*, 2nd edition. Upper Saddle River, NJ: Pearson.

Kassler, M. (1967). *A Trinity of Essays*. PhD dissertation, Princeton University.

Kassler, M. (1975). *Proving Musical Theorems I: The Middleground of Hienrich Schenker's Theory of Tonality* (Tech. Rep. No. 103). Sydney, Australia: University of Sydney, School of Physics, Basser Department of Computer Science.

Kassler, M. (1976). The decidability of languages that assert music. *Perspectives of New Music*, *14/2-15/1*, 249-251.

Kassler, M. (1977). Explication of the middleground of Schenker's theory of tonality. *Miscellanea Musicologica: Adelaide Studies in Musicology*, *9*, 72-81.

Kassler, M. (1988). APL applied in music theory. *APL Quote Quad*, *18*, 209-214.

Kirlin, P.B & Utgoff, P.E. (2008). A framework for automated Schenkerian analysis. *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Philadelphia, USA, 363-368.

Kirlin, P.B. (2009). Using harmonic and melodic analyses to automate the initial stages of Schenkerian analysis. *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Kobe, Japan, 423-428.

Komar, A.J. (1971). *Theory of Suspensions*. Princeton, NJ: Princeton University Press.

Larson, S. & Vanhandel, L. (2005). Measuring musical forces. *Music Perception*, *23*, 119-136.

Lerdahl, F., & Jackendoff, R. (1983). *A Generative Theory of Tonal Music*. Cambridge, MA: MIT Press.

Lerdahl, F. (2001). *Tonal Pitch Space*. Oxford: Oxford University Press.

Marsden, A. (2001). Representing Melodic Patterns as Networks of Elaborations. *Computers and the Humanities*, *35*, 37-54.

Marsden, A. (2005). Generative Structural Representation of Tonal Music. *Journal of New Music Research*, *34*, 409-428.

Marsden, A. & Wiggins, G.A. (2008). Schenkerian reduction as search. *Proceedings of the fourth Conference on Interdisciplinary Musicology (CIM08)*, Thessaloniki, Greece, http://cim08.web.auth.gr/cim08_papers/Marsden-Wiggins/Marsden-Wiggins.pdf (last accessed 8/12/09).

Mavromatis, P., & Brown, M. (2004). Parsing Context-Free Grammars for Music: A Computational Model of Schenkerian Analysis. *Proceedings of the 8th International Conference on Music Perception and Cognition*, Evanston, USA, 414-415.

Narmour, E. (1977). *Beyond Schenkerism: the need for alternatives in music analysis*. Chicago IL: University of Chicago Press.

Oura, Y., & Hatano, G. (1991). Identifying melodies from reduced pitch patterns. *Psychologica Belgica*, *31*, 217-237.

Pachet, F. (2000). Computer analysis of jazz chord sequences: is *Solar* a blues?. In E. Miranda (ed.) *Readings in Artificial Intelligence and Music*, Amsterdam: Harwood Academic Publishers, 85-113.

Pankhurst, T. (2008). *SchenkerGUIDE*. London: Routledge.

Plum, K-O. (1988). Towards a Methodology for Schenkerian Analysis (translated by William Drabkin). *Music Analysis*, *7*, 143-164.

Schachter, C. (1990). Either/Or. In H. Siegel (ed.), *Schenker Studies*, Cambridge: Cambridge University Press, 165-79. Reprinted in C. Schachter, *Unfoldings,* Oxford: Oxford University Press, 1999, 121-33.

Schenker, H. (1935). *Der frei Satz*. Vienna: Universal Edition. Published in English as *Free Composition*, translated and edited by E. Oster, New York: Longman, 1979.

Smoliar, S.W. (1980). A computer aid for Schenkerian analysis. *Computer Music Journal*, *4*, 41-59.

Steedman, M.J. (1984). A generative grammar for jazz chord sequences. *Music Perception*, *2*, 52-77.

Steedman, M.J. (1996). The blues and the abstract truth: music and mental models. In A. Garnham & J. Oakhill (eds.), *Mental Models in Cognitive Science*, Hove: Psychology Press, 305-318.

Temperley, D. (2007). *Music and Probability*. Cambridge, MA: MIT Press.

Terrat, R.G. (2005). A pregroup grammar for chord sequences. *Journal of New Music Research*, *34*, 355-360.