Evolving Musical Harmonisation

Somnuk Phon-Amnuaisuk, Andrew Tuson, and Geraint Wiggins
Department of Artificial Intelligence, University of Edinburgh
80 South Bridge, Edinburgh EH1 1HN, Scotland, UK.
Email: {somnukpa,andrewt,geraint}@dai.ed.ac.uk

Abstract

We describe a series of experiments in generating traditional musical harmony using Genetic Algorithms. We argue that the encoding of explicit domain knowledge in these algorithms is helpful in producing a reasonable result, and discuss some problems which are specific to the musical domain. We conclude that a GA with no notion of the overall structure of a piece of music, even when given explicit domain knowledge, is unlikely to solve the harmonisation problem well, due to the lack of meta-level control of the reasoning process.

1 Introduction

In recent years, the use of evolutionary techniques such as Genetic Algorithms (GAs) has generated significant interest in the artificial intelligence and computer science communities. This has been reflected in a number of publications in the computer music world, some of which will be discussed later. However, as GA research proceeds, it is becoming clear that the operation of a GA need not be enormously different from that of a knowledge-based system (KBS) (Tuson (1998)). Indeed, Wolpert and Macready (1997) have suggested that for an evolutionary method to be really effective, domain-specific knowledge is not just desirable, but strictly necessary. In addition, practitioners such as Michalewicz (1992) and Davis (1991), amongst others, have advocated the use of domain-specific representations and operators.

In this paper, we explore two aspects of the application of GAs to music:

1. the use of knowledge-rich structures and procedures within the algorithm itself, as opposed to the more traditional use of GA components which are not problem-specific;
2. the strict use of objective methods, in the sense that any reasoning encoded in the GA should be stated explicitly, rather than being implicit in the expressed opinion of a human user.

These criteria are important because we are interested in simulating and understanding aspects of human behaviour, and not just in achieving a musical result. We wish to be able to examine the internal behaviour of our methods, and attempt to form some notion of *why* the answer we achieve is produced. In particular, we wish to compare the behaviour of our harmonisation system with human behaviour, and attempt to explain any discrepancies.

This paper is structured as follows. We present brief statements on the issues of implicit *vs.* explicit knowledge representation and interaction *vs.* non-interaction in GAs. We then outline existing applications of GAs in computer music. We present a case study of a knowledge-rich musical GA, including a discussion of some significant problems, and then draw conclusions about the implications of the work for musical GAs in general.

2 Explicit and Implicit Knowledge Representation

Early GA research supposed that the best way to represent the solution space of any GA problem was with a set of binary strings, as in, for example, Goldberg (1989). These strings were manipulated by operators which were independent of the domain over which the GA was defined. The *schema theorem* of Holland (1975) was assumed to demonstrate that this use of low cardinality representations was the most efficient way to run a GA. More recently, Fogel and Ghozeil (1997), amongst others, have shown this to be incorrect.

An alternative approach is to use a solution space represented by sets of other structures which directly represent the problem in an *explicit* fashion. This has a number of potential advantages, including:

1. it is possible to see directly and without intermediate interpretation what is the current state of the solution space;

2. it is possible to engineer *knowledge-rich* operators which act in some way which is *useful* in leading the search to an acceptable or optimal solution.

In particular, in the work presented here, it has been helpful to introduce *directed mutations* (Corne et al. (1994)) which preferentially change the elements of the solution space in ways which are musically meaningful. This allows the search to make rapid leaps towards acceptable solutions which would not be possible with an implicit representation.

A useful consequence of using explicit knowledge representation in the potential solutions of a GA is that it becomes possible to restrict the search by ruling out unacceptable solutions, simply by making them unrepresentable. This, of course, was always possible in the implicit representation style, but the building of implicit assumptions into experiment or theory is always scientifically suspect – therefore we argue that the explicit approach here is superior.

Finally, one particular issue which is obfuscated by the use of implicit representations is that of whether GAs are a symbolic or a sub-symbolic method. Many researchers seem to suppose that they are sub-symbolic. We suggest that our experiments, among many others by other researchers, clearly indicate that this is not the case, and that GA systems are indeed symbolic, and are the better for being so.

3 Interactive GAs

The *Interactive GA* (IGA) approach, taken by many GA applications in the musical domain, uses a human listener as a means to evaluate the fitness of chromosomes. This approach is flawed from our point of view, for the following reasons.

1. It is subjective, because it relies on individual preferences – we want an objective measure of what is going on in our system, so we can properly judge its performance;

2. Human listeners tend to become more open to a given piece of music on repeated hearings, and are prone to other inconsistencies based on mood, attention span, and so on;
3. An IGA does not allow us to study the fitness function itself, to determine how faithful it is to our chosen task – so using an IGA would be removing a major object of interest in this study.

In the experiments described here, music-theoretical knowledge is used to construct a fitness function in objective logical terms, which allows us to examine the behaviour of the system more scientifically than would an IGA. The GA can then be used to test a theory of musical behaviour by observing the behaviour produced.

4 Existing Work on GAs in Music

GAs have been used in music generation elsewhere. Examples include Horner and Goldberg (1991) who used a GA for thematic bridging; Biles (1994), who used an IGA for Jazz improvisation; Jacob (1995), who devised a composing system using an IGA; and McIntyre (1994) and Horner and Ayers (1995). McIntyre used a GA to generate a four part harmonisation of an input melody, focusing on Baroque harmony, while Horner and Ayers focussed on the harmonisation of chord progressions using GAs.

A main aim of our harmonisation project is to investigate the potential of a knowledge-rich GA and its performance in the musical domain. So our search is not artificially limited as in McIntyre's system (which only used a C major scale); nor is there problem abstraction as in Horner and Ayres' system, (which uses the GA to generate parts, given a chord progression, which is a significantly simpler task). Our work aims to harmonise input melodies with no explicit cues as to the required harmony, and does not limit itself to a specific key or scale; and it works at the level of individual voices, with all the extra constraints this entails.

Finally, for a more complete summary of GA work in music, the reader is directed to Burton and Vladimirova (1997).

5 A GA to Harmonise Chorale Melodies

In this section, we present the results of a study on the use of GAs in generating four-part homophonic tonal harmony for user-specified melodies. The domain-specific (*i.e.,* musical) knowledge in this system is implemented in three parts of the GA. These are described in turn, and followed by an overview of the GA configuration used.

5.1 Chromosome Representation

Generally speaking, keys and chords are the main concepts in harmonisation of western tonal music. Harmonisation rules are expressed in terms of relationships between triads, and between degrees of scale within a key signature (*e.g.,* tonic-dominant, etc) but not the absolute pitch. Therefore, in this implementation, musical information (*e.g.,* pitch, interval, time, duration) is represented after *normalisation* with respect to key – that is, absolute pitch information is abstracted out. Then, pitch is expressed in terms of scale degree. To express all twelve semitones, the standard five accidentals are used. Different octaves are distinguished by an associated integer. Finally, time intervals are represented as integers. The representation conforms to the CHARM specification of Wiggins et al. (1989).

As noted earlier, a knowledge-rich, explicit, and directly meaningful representation is used in our chromosome representation (compare with the binary strings more commonly used in GAs). This representation may be thought of as a matrix, which consists of five strings of equal, fixed length. The top four strings contain soprano, alto, tenor and bass parts, with the fifth describing the durations of the chords. The user inputs the soprano information (assumed to be the melody); the GA will then harmonise the input soprano, homophonically, with a further three voices in conjunction with the musical domain knowledge encoded in its operators and fitness function.

This approach is illustrated in Figure 1, using the completed harmonisation for the first two bars of "O Come, All Ye Faithful"; see Figure 4 for the score rendition of the corresponding output.

chromosome length

Soprano	[0,0,3]	[0,0,3]	[4,0,2]	[0,0,3]	[1,0,3]	[4,0,2]
Alto	[2,0,2]	[2,0,2]	[2,0,2]	[2,0,2]	[4,0,2]	[1,0,2]
Tenor	[4,0,1]	[4,0,1]	[2,0,1]	[0,0,2]	[7,0,1]	[1,0,1]
Bass	[2,0,1]	[0,0,1]	[0,0,1]	[4,0,1]	[4,0,1]	[7,0,0]
Duration	1	2	1	1	2	2

Figure 1: Schematic Diagram of a Four-Voice Harmony Chromosome

In the figure, we can see the matrix arrangement produced by the direct representation of time (left to right along the structure) and vocal part and note duration (top to bottom down the structure). For the purposes of breeding operators, we view the five horizontal layers as inseparable.

5.2 Reproduction Operators

The following crossover and mutation operators are used in this implementation, described here in musical terms. The reader less familiar with musical jargon may find solace in Taylor (1996).

Splice: One point crossover between two chromosomes – selects a crossover point between successive notes of the melody and corresponding chords.

Perturb: Mutate by allowing alto, tenor and bass to move up or down by one semitone or tone. The selection of the various possible mutations is random.

Swap: Mutate by swapping two randomly picked voices between alto, tenor or bass. This gives the effect of changing the chord between different open and closed positions, and of changing inversions.

Rechord: Mutate to a different chord type. This mutation generates a new chord from the melody data. A chord is built with the soprano note as root, 3rd or 5th. Doubling (necessary for a four note chord) can be in any position.

PhraseStart: Mutate the beginning of each phrase to start with tonic root position on a down beat.

PhraseEnd: Mutate the end of each phrase to end with a chord in root position.

5.3 Fitness Function

The fitness function judges the fitness of each chromosome according to the following criteria derived from music theory. Within individual voices (as opposed to between voices), we prefer stepwise progression over large leaps, and we keep the voice within its proper range. We penalise progression to dissonant chords, and we avoid leaps of major and minor 7ths, of augmented and diminished intervals, and of intervals larger than one octave.

Between voices, we apply the following criteria: we avoid parallel unison, parallel perfect 5ths, and parallel octaves; we forbid progression from diminished 5th to perfect 5th (though the converse is allowed); we avoid hidden unison; we forbid crossing voices; and we forbid hidden 5th and octave in the outer voices, when soprano is not progressing stepwise.

Solutions are penalised for note doubling and omission, in the primary major and minor triads: doubling of the root (tonic) is preferred, while doubling of the 3rd is penalised; doubling of the 3rd is forbidden in a dominant chord; if it is necessary to omit a voice, omit the fifth only, except in 1st inversion; in inverted chords, doubling of the bass is preferred; and we penalise doubling of tones which give a strong harmonic tendency, such as leading notes.

In this implementation, the system does not have enough knowledge to plan for large scale harmonic progression. The fitness function determines only the plausible harmonic movement between two adjacent chords. The fitness function prefers (in decreasing order of preference): descending 5th movement; progression towards the tonic; retrogression; and repetition.

5.4 GA Configuration

In most of these experiments, exceptions being mentioned explicitly here, a generational GA, in the style of Davis (1991), with a panmitic (unstructured) population model was used. Strings were initialised by randomly picking chords out of a list suggested by soprano pitches. Finally, a population size of 50 was used with binary tournament selection.

6 Results and Analysis

Figure 2 shows a harmonisation by our system of the first eight notes of "Joy to the world". The output is not perfect, but it is surprisingly good given the limited rules built into the system. All the output of the system was marked by Dr. John Kitchen, a lecturer in the Faculty of Music at the University of Edinburgh, according to the criteria he uses for 1st year undergraduate students' harmony. This example scored 5 out of 10 – a clear pass. While other examples were less successful (most earning around the 30% mark), according to the marker, this was mostly due to the lack of coherent large-scale musical progression, which was not represented in the fitness function. Interestingly, the system was judged by the marker to be better than student harmonisers at getting the basic rules right.

Some experimentation was carried out with various GA parameter settings. As expected, the weights of the various penalties applied in the fitness function have a significant

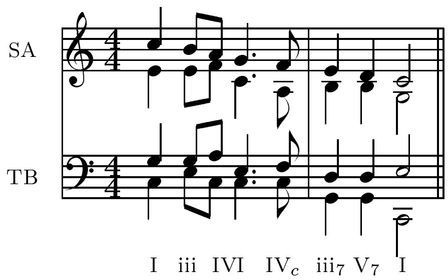

Figure 2: Harmonisation of the First Line of *Joy to the World*

effect on the solution. Other parameters, such as crossover rate, mutation rate, and different selection schemes appear to affect the time taken for the population to converge, and do little for the solution quality. This is due to the fact that it is the fitness function which primarily defines the knowledge in the system pertaining to what does or does not constitute a good pice of music, while the other parameters define the search strategy.

What is most significant – and problematic – is that, with the current evaluation functions and reproduction operators, the GA still cannot satisfy all the constraints within 300 generations. Figure 3 illustrates a typical fitness profile of the best solution in each generation. The data is from the first phrase of the hymn "O come, all ye faithful", which is twenty four notes long. The figure shows the distribution of penalties along the chromosome – in other words, the score of each harmonic movement or chord in the piece – as the generations proceed. The higher the contour, the less acceptable the chord at that point.

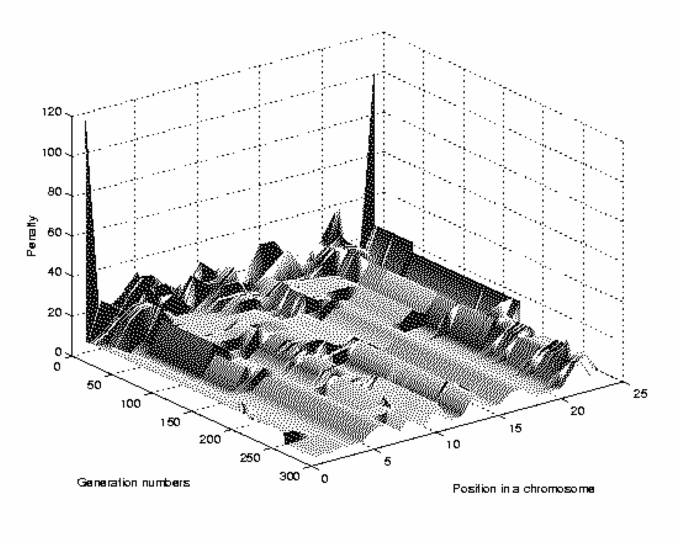

Figure 3: A Typical Fitness Profile Landscape

In the figure, we can see that the distribution of penalties at the beginning is quite random. After a few generations, it starts to shape to a certain pattern. However, the GA could not reduce all the penalties in the fitness function. The musical output associated with the final contour in the diagram is shown in Figure 4.

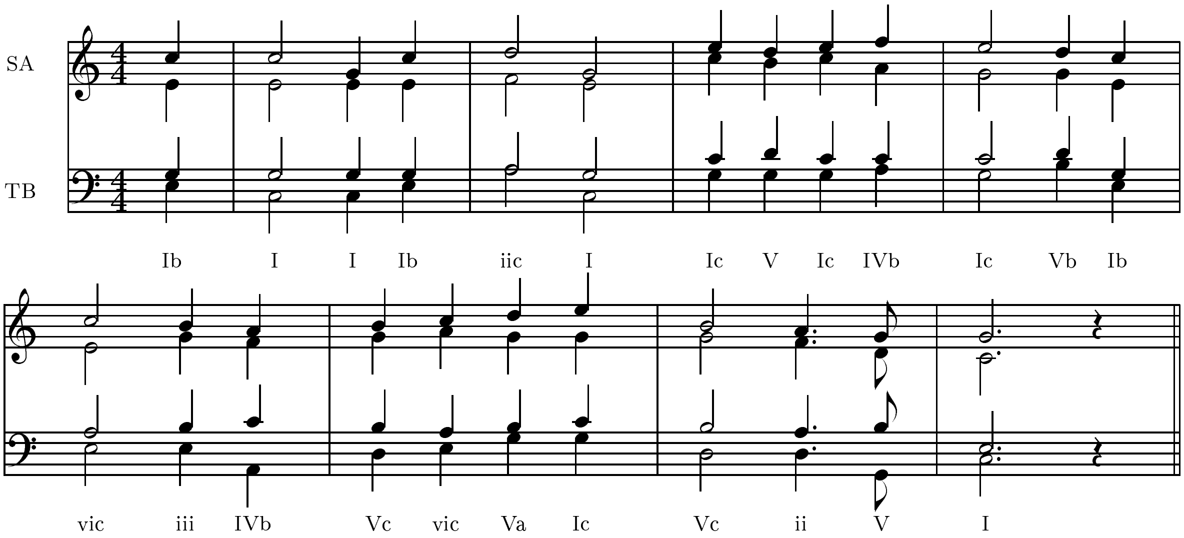

Figure 4: A GA Harmonisation of "O Come, All Ye Faithful"

An experiment with an *island model* (Gordon et al. (1992)) with four population groups was also carried out, to determine whether different population groups might be able to preserve their own salient cultures, and might be able to bring the GA to a more globally acceptable solution. Figure 5 is an example result from the island model. However, though the experiment showed an improvement in search efficiency, the GA still could not reduce all of the penalties.

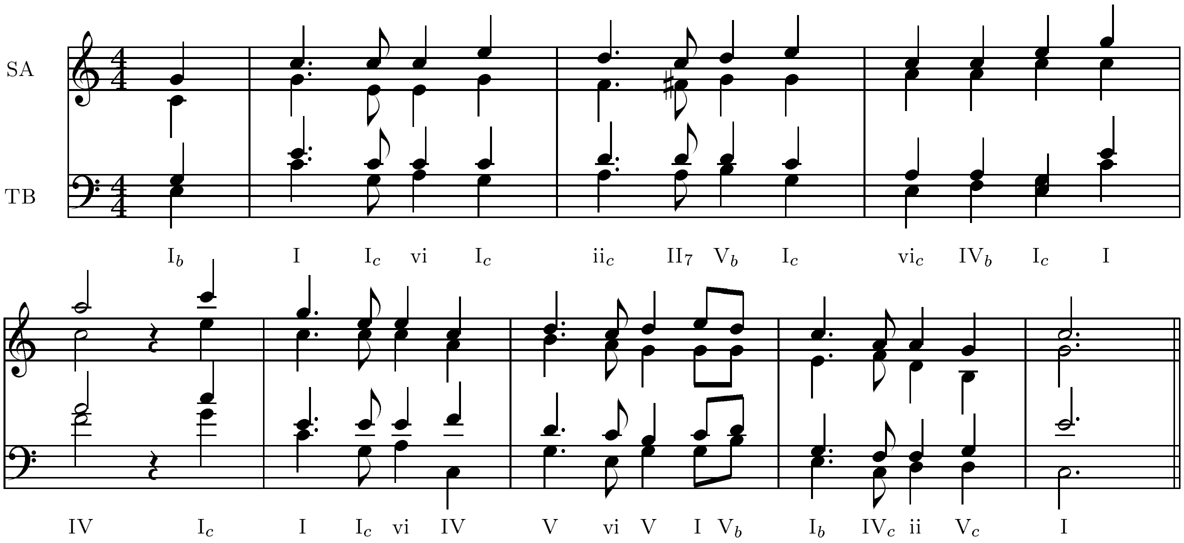

Figure 5: Harmonisation of *Auld Lang Syne*

Why, then, do these problems arise? They arise because the structure of a harmonisation of this kind is very specific, in that individual variations in chord are very strongly context dependent. Therefore, it is often the case that one cannot change any given chord without changing the chords around it. In terms of the GA, this means that reduction of a fitness penalty in one position is likely to increase penalties in other positions, because the movement from one chord to the next is not considered with respect to overall movement in the phrase.

Human composers solve this problem by structuring the construction process itself in an explicit way and designing an overall harmonic framework, and then filling in the most crucial parts first – in short, a least commitment strategy is taken. This kind of structured reasoning is not feasible in a simple GA of the kind used here, and so the results are sub-optimal.

To restate this argument in terms of the search space: we believe that the problem is due to a multimodal fitness landscape, characterised by many local optima in basins of attraction which separated by very high barriers, due to the interactions described above. Before the GA can move from one basin of attraction to another, many chords leading to a fitness penalty need to be changed. Such a simultaneous change is very unlikely to occur.

7 Conclusions

It is quite clear from the experiments here and elsewhere that GAs can be applied successfully in the musical domain – up to a point. It is also clear that the efficacy of the GA approach depends heavily on the amount of knowledge the system possesses. Looking at the output of our systems from an aesthetic viewpoint, the results are still far from ideal: the harmonisation produced by the GA has neither clear plan nor intention. This is not a surprise as the discussion above suggested that we cannot expect large scale structure to arise from the kind of programming inherent in a GA containing (even in this experiment) relatively little domain knowledge.

In summary, therefore, we conclude that while GAs can be surprisingly good at small, constrained musical tasks, their performance, at least in this context of simulating human musical behaviour, is currently limited by two issues.

1. GAs are a stochastic, heuristic search method, so one cannot be sure that an optimal solution will be reached, even if there is one. In particular, in a problem of this kind, they tend to get stuck in local optima in the search space.

2. GAs of the form used here lack structure in their reasoning. On the other hand, composers have developed complex and subtle methods over several centuries involving different techniques for solving the problems highlighted here. No musician would seriously suggest that an author of hymn tunes works in the same way as our GA. Therefore, while we may be able to produce (near) acceptable results with a GA, doing so says little about the working of the compositional mind.

Other approaches to using GAs for this kind of task may be more successful. For example, one might take the approach of optimising a set of instructions to plan a harmonisation, rather than actually operating on the musical score directly; this approach is similar to the *indirect* representational approach used by Burke et al. (1995) and others for timetabling and scheduling problems.

An alternative solution in the context of our GA would be to introduce even more knowledge-rich mutations, which would possess knowledge about the entire harmonic structure of each candidate solution, and thus would be able to jump directly across the barriers in the fitness landscape described earlier in a single move. However, it would be hard indeed to make a clear differentiation between either of these approaches and a conventional KBS.

We conclude, therefore, that neighbourhood search methods such as GAs are currently limited in the musical domain by their lack of knowledge about the structure of musical harmony, and methods by which this knowledge can be represented and effectively exploited. It would appear that, if GAs are to both improve in performance, and allow us to gain insights into the compositional mind, dealing with the issues raised here is an urgent task. We end with a suggestion that it is likely that much can be gained by combining a GA somehow with a conventional rule-based system approach for the harmonisation task.

Acknowledgements

Thanks to Dr. John Kitchen for his help in assessing the harmonisation system. Andrew Tuson is supported by EPSRC studentship, reference number 95306458.

References

Biles, J. A. (1994). GenJam: A genetic algorithm for generating jazz solos. In *ICMC Proceedings 1994*. The Computer Music Association.

Burke, E., Elliman, D., and Weare, R. (1995). The Automated Timetabling of University Exams using a Hybrid Genetic Algorithm. In *The AISB Workshop on Evolutionary Computing*.

Burton, A. R. and Vladimirova, T. (1997). Applications of genetic techniques to musical composition. Available by WWW at `http://www.ee.surrey.ac.uk/Personal/A.Burton/work.html`.

Corne, D., Ross, P., and Fang, H.-L. (1994). Fast Practical Evolutionary Timetabling. In Fogarty, T. C., editor, *Selected Papers: AISB Workshop on Evolutionary Computing, Lecture Notes in Computer Science No 865*, pages 250–263. Springer Verlag.

Davis, L., editor (1991). *Handbook of Genetic Algorithms*. New York: Van Nostrand Reinhold.

Fogel, D. B. and Ghozeil, A. (1997). A Note on Representations and Variation Operators. *IEEE Transactions on Evolutionary Computation*, 1(2):159–161.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization & Machine Learning*. Reading: Addison Wesley.

Gordon, V., Whitley, D., and Böhn, A. (1992). Dataflow parallelism in genetic algorithms. In Männer, R. and Manderick, B., editors, *Parallel Problem Solving from Nature 2*, pages 553–42, Amsterdam. Elsevier Science.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. Ann Arbor: The University of Michigan Press.

Horner, A. and Ayers, L. (1995). Harmonisation of musical progression with genetic algorithms. In *ICMC Proceedings 1995*, pages 483–484. The Computer Music Association.

Horner, A. and Goldberg, D. E. (1991). Genetic algorithms and computer-assisted music composition. Technical report, University of Illinois.

Jacob, B. L. (1995). Composing with genetic algorithms. Technical report, University of Michigan.

McIntyre, R. A. (1994). Bach in a box: The evolution of four-part baroque harmony using a genetic algorithm. In *First IEEE Conference on Evolutionary Computation*, pages 852–857.

Michalewicz, Z. (1992). *Genetic algorithms + data structures = evolution programs.* Artificial Intelligence. Springer-Verlag, New York.

Taylor, E. (1996). *The* **AB** *Guide to Music Theory.* The Associated Board of the Royal Schools of Music, London.

Tuson, A. L. (1998). Optimisation with hillclimbing on steroids: A tutorial on neighbourhood search. In *Keynote Papers of the 10th Young OR Conference.*

Wiggins, G., Harris, M., and Smaill, A. (1989). Representing music for analysis and composition. In Balaban, M., Ebcioglu, K., Laske, O., Lischka, C., and Sorisio, L., editors, *Proceedings of the 2nd IJCAI AI/Music Workshop*, pages 63–71, Detroit, Michigan. Also from Edinburgh as DAI Research Paper No. 504.

Wolpert, D. and Macready, W. (1997). No Free Lunch Theorems for Search. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82.