An AI Tool for the Analysis and Generation of Melodies

Conference or Workshop Item

An AI tool for the analysis and generation of melodies

Matt Smith and Simon Holland

Department of Computing, The Open University, Milton Keynes, England MK7 6AA
email: matt@uk.ac.abdn.csd / simon@uk.ac.open

ABSTRACT

Artificial Intelligence (AI) offers music theorists and cognitive musicologists the means to express theories unambiguously, through the implementation of computational models. Any inconsistencies in a theory will be discovered during the encoding process, and, since the computer has no musical intuitions or experience, any **obvious** or assumed knowledge must be made explicit by the knowledge engineer.

Our eventual goal is the development of an intelligent computer environment for students learning melody composition. This paper describes the development of one tool which will be used within the tutoring system: a program for the analysis and generation of melodies, based upon Eugene Narmour's (1990) *Implication-Realisation Model* - a hypothetical theory for melody analysis.

The tool is being developed using the declarative, logic-based language Prolog. In addition to encoding Narmour's analytical theory, we have designed extensions to the computational model which allow the generation of melodies using constraint-based techniques. This means that with a single computer model a user will be able to analyse a given melody, modify a melody in musically 'sensible' ways (explained below) by changing notes at various hierarchical levels, generate new melodies and finish partially completed melodies.

The paper is presented in four sections: the first describes those aspects of Narmour's theory necessary to understand the computational model; the second section describes the design of the analytical model; the third outlines how the computational model can be used to test Narmour's theory; a final section describes the additions to the model to allow generation of melodies using constraint techniques.

KEYWORDS

Artificial Intelligence; Music Analysis; Music Composition; Melody; Symbolic Programming; Constraints.

1. NARMOUR'S THEORY

Narmour's theory is based on two hypotheses which predict the moment-by-moment, low level expectations (implications) a listener has of the properties of future melodic events (such as contour, interval size and duration). In addition Narmour considers the effect of **style** learned by a listener, as a top-down effect on expectations. The two hypotheses are called **"continuation"** and **"reversal"**. The hypothesis of continuation, based on the Gestalt school of psychology, states that under certain circumstances (usually when a listener hears a small interval, Narmour defines when an interval is considered small) the listener will expect the next interval to have similar properties to that of the generating interval (e.g. ascending contour followed by ascending contour, small interval followed by small interval). The hypothesis of reversal is based upon Narmour's own musical intuitions (and can be found in other's theories, for example Meyer 1956). Reversal is when a listener expects melodic parameters to change; such expectations are usually initiated by a large interval and in such situations the listener normally expects the large interval to be follow by small, and contour to change direction. Narmour describes a number of **"structures"** (resulting from the hypotheses of continuation and reversal), these structures group together consecutive notes in a melody which realise and deny the same expectations. For example, one such structure is **"prospective intervallic process"**, symbolised by the letters **IP** - all notes in such a structure realise expectations of similar, small intervals (hence the 'I'), but deny expectations of an unchanging contour. The two hypotheses lead to two classes of structures : *continuation* structures are either **"Iterative"** (where pitches or intervals are repeated) or **"Processive"** (where contour and interval size are continued). There is only one category of *reversal* structures.

A key concept of the implication-realisation model is that of **"closure"**. Narmour uses the term in a general sense, where closure varies in degree of strength, occurring throughout a melody, as well as at points when a number of melodic parameters combine in a manner indicating a summarisation.

> (page 45, Narmour 1990)
> Closure refers *"... to the various ways musical parameters interact to create melodic 'chunks', perceptual groupings whose beginning and ending nodes exhibit varying degrees of stability..."*
>
> (page 102, Narmour 1990)
> *"By closure I refer to syntactic events whereby the terminating, blunting, inhibiting, or weakening of melodic implication occurs"*

Closure causes structures to terminate, and if closure on a note is strong enough the note is promoted to the next hierarchical level of the analysis. Narmour's theory also takes into account how stylistic learning can effect a listener's expectations, from the top down (as opposed to the low level, bottom-up hypotheses of continuation and reversal). Narmour considers style from two points of view : the extra-opus style a listener builds up by listening to a corpus of melodies in a tonal tradition (e.g. Western Tonal Music), and the intra-opus style of each melody, where repeated figures and their variations are quickly recognised (and so expectations effected) at later points in a melody. Structures in Narmour's theory can be **"prospective"**, where the expectations are matched with forthcoming notes, or **"retrospective"** where the listener suddenly comes to understand some past notes and revises the current structure. These retrospective structures are analogous to **"garden path"** sentences in natural language (e.g. Markus, 1980).

2. THE COMPUTATIONAL MODEL

The analysis program has been written as a type of parser, and although it is not based upon a strict grammar, like a typical natural language program, it does exhibit some of the behaviour of such parsers. The analysis program looks at each note in the melody in onset time order, and given a new note it analyses it in a number of steps :

- **(a)** check note attributes (pitch, onset time, duration),
- **(b)** calculate attributes of interval between new note and previous (interval size in semitones, contour),
- **(c)** check whether in the current context the interval is considered large or small according to the implication-realisation model (small intervals initiate continuation expectations, while large ones initiate reversal),
- **(d)** check if stylistic knowledge matches with the current events and if so, generate expectations according to style knowledge,
- **(e)** if there is an existing structure see if the new note causes closure or whether the structure should be extended to include the new note, or possibly whether the old structure should be abandoned and a retrospective one created; if there was no existing structure then initiate one,
- **(f)** run an intra-opus style learning program,
- **(g)** if closure has occurred, see if it is strong enough to promote the new note to the next higher hierarchical level. If promotion then go through steps **(b)**-**(g)** for the notes and structures at the next hierarchical level.

As the program moves through the melody it will build up structures at a number of hierarchical levels, and will exhibit the re-structuring of past notes when it encounters situations of retrospective structures. Although a more efficient parser could be written, working on a whole melody in one pass, it would have the limitations of not being able to show the intermediate structures before retrospective structures are identified, and could only work on complete melodies. The note-by-note parser described in steps **(a)**-**(g)** above has neither of these limitations.

To date we have implemented a parser which embodies a number of aspects of Narmour's theory (i.e the two hypotheses, the structures and closure). Currently stylistic modules are being designed, which will perform the tasks represented by steps **(d)** and **(f)** above.

3. A PROGRAMME FOR TESTING NARMOUR'S THEORY

There are a number of ways in which computational models of musical theories can be used :

- **(a)** to create an unambiguous (formal) description of the theory,
- **(b)** to find any inconsistencies in the theory (e.g. when *"hand-coded"* examples don't match output of computational model),
- **(c)** to identify any assumed knowledge, and force the model developer to make explicit all aspects of the theory (e.g. one must formally describe harmonic and metric *"strengths"* if the theory refers to such measures),
- **(d)** to provide an easy and quick method for seeing the results of changes to the theory (i.e. change the computer model and re-run experiments, then compare results),
- **(e)** to make realistic the application of the theory to a large number of experiments (e.g. for an analytical model to analyse a large corpus of musical examples), previously out of the question due to the sheer time for human application of the theory,
- **(f)** to make possible communication of the theory with little effort (i.e. by being able to simply send copies of a computer program).

With Narmour's theory in mind, a number of steps for the testing of our computational model (and the theory) suggest themselves :

- **(1)** check program analyses against Narmour's published examples (if mismatches, then check program for mistakes, if program okay then check theory for incompleteness),
- **(2)** for any given structure (or chain of structures) of a set number of notes and pitch range, all possible melody fragments matching the structure(s) can be generated; this can bring to light incomplete definitions in the theory which have been overlooked due to their occurrence only in unlikely sequences of notes,
- **(3)** generate new analyses for many melodies, look for unmusical analyses and check theory (although 'unmusical' is of course subjective, the analyses could be checked by a number of music theorists),
- **(4)** given analyses for a corpus of melodies, patterns can be searched for (e.g. certain structures only occurring at certain hierarchical levels), perhaps leading to a useful method for stylistic analysis.

4. THE CONSTRAINT-BASED GENERATOR

Melodies can be generated by applying harmonic and metric constraints to notes occurring in particular levels of the hierarchy. Such constraints might be very restricting at high levels, and weakened for each lower level. For example, harmonic constraints could take the form of movement through Lerdahl's (1988) *"Pitch Spaces"*, a set of spaces which contain pitch classes of increasing harmonic importance. There is also the built-in constraint that a note appearing in a particular hierarchical level must generate sufficient closure in the next lower level, to justify its place in the hierarchy. The constraint satisfaction process is driven by the logic programming language Prolog. Using Prolog's backtracking and automatic instantiation (a kind of reversible assignment) of variables, the same rules and facts that test for certain properties of a melodic sequence can also be used to assign properties to variables. In such a way the program to analyse a melody and generate an implication-realisation model hierarchy can be used (with extensions) to generate notes for the musical surface of a melody that satisfy the structures and constraints at higher hierarchical levels. The extensions will take the form of heuristics for the appropriate application of constraints at given hierarchical levels. These constraints and the heuristics for their application are, in effect, compiled stylistic knowledge. A further possible extension would be to generalise about stylistic constraints in terms of Narmour's theory, rather than compiled extensions to his theory. The motivation to extend Narmour's model for melody generation is the development of a tool for students with little music training who wish to learn about melody composition. The computer environment will guide students in the application of constraints.

Appendix I illustrates the proposed constraint techniques by generating the opening of the melody for Swan Lake (Tchaikovsky 1877). The melody is generated in four stages, starting at **H4** and moving down to **H1** (the musical surface). Harmonic constraints based on Pitch Spaces (Lerdahl 1988) (with key as A minor) are :

H4 : Open Fifth Space (A,E)
H3 : Triadic Space (A,C,E)
H2,H1 : Diatonic Space (A,B,C,D,E,F,G)

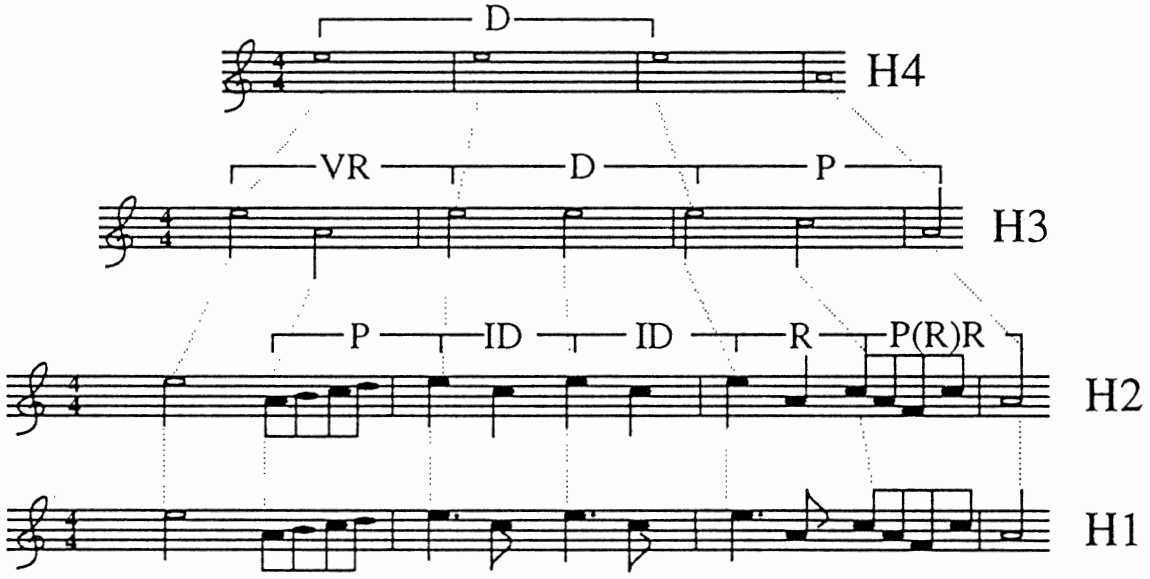
Metric constraints are :
H4 : Bar boundaries (i.e. semibreve, whole note)
H3 : Half bar boundaries (i.e. minim, half note)
H2,H1 : Eighth bar boundaries (i.e. quaver, eighth note)

With these constraints the choice by the novice composer becomes which structure and how many notes. After a generation, the novice can ask for alternatives (again Prolog's backtracking can be used to generated alternative variable instantiations). The step from **H2** to **H1** is to satisfy the inherent constraint that each note to be promoted must create sufficient closure to justify its promotion (durational closure being added by the changes from **H2** to **H1**). Therefore, **H2** can be considered an first attempt by the generator to create a melody fitting the constraints, while **H1** is an alternative (with only durations changed) that satisfies all constraints.

5. SUMMARY

This paper describes a comprehensive, symbolic implementation of Narmour's *"Implication-Realisation model"*. We have described a programme for the testing of Narmour's theory using the computational model. The principled application of hierarchical harmonic and metric constraints allow this low-level, bottom-up theory to form the basis of a high-level tool for the generation of melodies.

6. APPENDIX I : GENERATION OF OPENING OF SWAN LAKE



7. REFERENCES

(Lerdahl, 1988) Fred Lerdahl, "Tonal Pitch Space", *Music Perception*, Vol. 5, No. 3, pp.315-350, Spring 1988.

(Markus, 1980) Mitch Markus, Theory of Syntactic Recognition for Natural Language, MIT Press, Cambridge, MA, 1980.

(Meyer, 1956) Leonard B. Meyer, Emotion and Meaning in Music, University of Chicago Press, Chicago, 1956.

(Narmour, 1990) Eugene Narmour, The Analysis and Cognition of Basic Melodic Structures, University of Chicago Press, Chicago, 1990.

(Tchaikovsky, 1877) Peter Ilyich Tchaikovsky, "Swan Lake", written 1877.