A MULTILEVEL MELODIC SIMILARITY FRAMEWORK

Chris Walshaw
Faculty of Architecture, Computing &
Humanities, University of Greenwich,
London SE10 9LS.
c.walshaw@gre.ac.uk

ABSTRACT

This paper proposes a multilevel framework for measuring melodic similarity in symbolic musical representations. The basis of the framework is straightforward: initially each tune is normalised and quantised and then recursively coarsened, typically by removing weaker off-beats, until the tune is reduced to a skeleton representation with just one note per bar. Melodic matching can then take place at every level: the multilevel matching implemented here uses the longest common substring algorithm at each level, but in principle a variety of similarity measures could be used. The multilevel framework is also exploited to reduce computational complexity.

1. INTRODUCTION

1.1 Background

This paper discusses a multilevel melodic similarity measure used to identify related tunes and variants found within the abc notation music corpus.

Abc notation is a text-based music notation system particularly popular for transcribing, publishing and sharing folk & traditional music online. Similar systems have been around for a long time but abc notation was formalised and named by the author in 1993. He maintains the website abcnotation.com with links to tutorials, software, tune collections and, in particular, an online tune search engine which indexes around 480,000 abc transcriptions from across the web and currently attracts around half a million visitors a year.

In 2014 the search was further enhanced with the introduction of TuneGraph, an online visual tool for exploring melodic similarity, [1]. TuneGraph uses a melodic similarity measure to derive a proximity graph representing similarities within the abc notation corpus backing the search engine. From this graph a small localised version is extracted for each vertex, aimed at indicating close variants of the underlying tune represented by the vertex. Finally an interactive user interface displays each local graph on that tune's webpage, allowing the user to explore melodic similarities.

The latest version of TuneGraph under test uses a prototype multilevel matching algorithm to construct the proximity graph.

2. MOTIVATION: A CASE STUDY

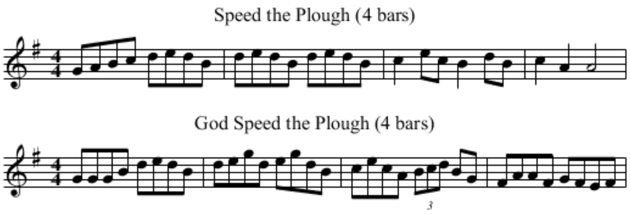

Figure 1. Two tune variants for Speed the Plough.

Figure 1 shows two related versions of the first 4 bars of Speed the Plough, a tune well-known across the British Isles. Typically in tunes such as these, the emphasis is placed on odd numbered notes, and in particular the first note of each beam. The strongest notes of the bar are thus 1 and 5, followed by 3 and 7. To capture this emphasis an approach has been developed, [1], which builds a multilevel (hierarchical) representation of the tunes.

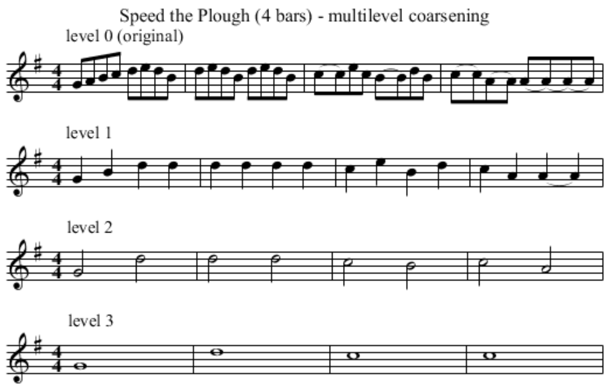

Figure 2. Multilevel coarsening of Speed the Plough

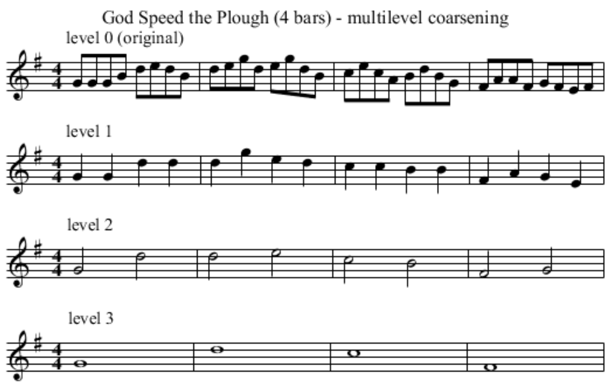

Figure 3. Multilevel coarsening of God Speed the Plough

The multilevel paradigm is simple one which involves recursive coarsening to create a hierarchy of increasingly coarse approximations some original representation. As a general solution strategy, the multilevel framework is widely used in combinatorial optimisation problems and can be extremely effective, both at imparting a global perspective and at accelerating solution techniques, [2].

As a motivational example, figures 2 and 3 show multilevel coarsened versions of the original tunes, where the weakest notes are recursively replaced by removing them and extending the length of the previous note.

3. IMPLEMENTATION & TESTING

The core of the multilevel implementation is very straightforward. Each tune is initially normalised and then recursively coarsened down to a skeleton representation with just one note per bar. Melodic matching can then take place at every level with heuristics used to optimise and enhance performance.

Normalisation: As part of the normalisation process, each tune is cleaned of grace notes, chords and other ornaments. Generally most tunes under consideration from the abc corpus are single-voiced but if not, only the first voice is used for the matching. Tunes are also quantised to give every note the same length, [3].

Coarsening: The coarsening works by recursively removing "weaker" notes from each tune to give increasingly sparse representations of the melody. Currently this is a subjective choice, and in the prototype implementation the default coarsening strategy considers that the weaker notes are the off-beats or every other note and it is these which are removed (see Figures 2 & 3). Exceptions to the "remove every other note" rule are handled with heuristics, typically for tunes in compound time, [3].

Coarsening progresses until there is one note remaining in each bar; it is possible to go further but experimentation suggests that the bar is a good place to stop.

Multilevel Matching: Once the multilevel representation is constructed the tune at each level is converted to an array of intervals and matching is done **at all levels**.

The prototype version uses the Longest Common SubString (LCSS) algorithm for matching but in principle various methods could be used, e.g. [4] (this is a strength of the multilevel paradigm which is not generally tied to a particular local search strategy).

Because folk and traditional tunes can differ widely at the finest level whilst resembling each other at coarser levels, the similarity values from each level are important and are combined in a weighted sum to give an overall multilevel similarity measure.

For convenience, it is helpful to formulate the matching problem as a minimisation and hence to express the similarity as a difference, D(X,Y). This is easy to do by computing, at each level l,

$$D_l(X,Y) = \min(\text{length}(X_l), \text{length}(Y_l)) - \text{LCSS}_l(X,Y),$$

where $\text{length}(X_l)$ is the length of the array of intervals at level l.

In absolute terms, $D_l(X,Y)$ much more emphasis on the finer levels, simply because there are many more notes. To compensate, contributions from the coarser levels are weighted with the value 2^l (based on the observation that the length of the interval arrays is approximately halved at each successive level), giving

$$D(X,Y) = \Sigma_l \, 2^l \, . \, D_l(X,Y)$$

as the overall multilevel difference measure under test.

Optimisation: The multilevel framework also allows the option to terminate the matching early, at the coarser levels, when it looks unpromising and heuristics are used both to reduce computational complexity and to enhance the matching qualitatively, [3].

Testing: Preliminary testing has indicated that the prototype algorithm works well at matching tunes to produce a proximity graph of tunes and initial results can be found at the abc notation website[1].

However, there are several limitations in the current implementation and, in particular, a number of tune features that it cannot handle fully, [3]. In addition further development work remains to be done, particularly in the light of more rigorous experimentation.

It is also arguable that the utility of the technique may be restricted to music from the folk and traditional genres with a strong on-beat structure (although the general framework could easily be used with different forms of melodic reduction).

However, the technique seems promising and worthy of further investigation.

4. REFERENCES

[1] C. Walshaw, "TuneGraph: an online visual tool for exploring melodic similarity," in *Proc. Digital Research in the Humanities and Arts*, 2015, pp. 55–64.

[2] C. Walshaw, "Multilevel Refinement for Combinatorial Optimisation: Boosting Metaheuristic Performance," in *Hybrid Metaheuristics - An emergent approach for optimization*, C. Blum, Ed. Springer, Berlin, 2008, pp. 261–289.

[3] C. Walshaw, "Multilevel Melodic Matching," in *5th Intl. Workshop on Folk Music Analysis*, 2015, pp. 130–137.

[4] D. Müllensiefen and K. Frieler, "Melodic Similarity: Approaches and Applications," in *Proc. 8th Intl Conf. Music Perception & Cognition*, 2004, pp. 283–289.

[1] http://abcnotation.com/searchHelp#TuneGraph