Subsumption of Vertical Viewpoint Patterns

Mathieu Bergeron\textsuperscript{1} and Darrell Conklin\textsuperscript{2}

\textsuperscript{1} CIRMMT, McGill University, Montreal, Canada\hspace{1em}mathieu.bergeron@mail.mcgill.ca
\textsuperscript{2} Department of Computer Science and AI
Universidad del País Vasco, San Sebastián, Spain
IKERBASQUE, Basque Foundation for Science, Bilbao\hspace{1em}conklin@ikerbasque.org

Abstract. This paper formalizes the vertical viewpoint pattern language for polyphonic pattern representation. The semantics of patterns is given in terms of a translation to a relational network form. The language supports pattern subsumption, an essential inference for pattern mining, development, and refinement. Though computed in a way entirely different to relational network matching, this paper proves that subsumption inferences are sound and complete with respect to the underlying relational language.

Keywords: Pattern subsumption, Polyphonic patterns, Music representation, Score slicing, Computational musicology.

1 Motivation

Pattern recognition and discovery are important topics in computational musicology. Patterns in music can be used to describe repetition within pieces and also recurrence within a corpus of pieces. Patterns may refer to, for example, melodic lines, chord sequences, or to more complex polyphonic structures where temporal relations exist between overlapping events.

Considering polyphony, a natural and powerful representation of polyphonic patterns is graph structures, where the nodes represent notes, or more generally features of notes, and edges represent their temporal relations. Graph representations require the computation of subgraph isomorphism to determine pattern to instance relations, and this has constrained their use in computational musicology.

The aim of this paper is to elaborate a new formalism called $\mathcal{VV}$ (vertical viewpoint patterns) for the representation of polyphonic patterns. The representation combines the viewpoint formalism \cite{1} with the idea of segmenting the polyphonic texture into slices \cite{2}. As a slice can contain the continuation of a musical event, a variety of temporal relations are supported. While this representation does not have the full power of graphs, it is suitably expressive for many types of musical patterns, specifically patterns where notes are temporally compact and occur in a fixed number of voices.

The method assumes that patterns have the same number of voices than the musical source from which instances are drawn. This can require the extraction
of fixed voice textures from actual musical data (e.g. considered voice pairs extracted out of the four-voice texture of Bach chorale harmonizations). A similar technique is also required when using Humdrum, a well-known toolkit for pattern matching in symbolic music data. Although Humdrum supports polyphony, it can be difficult to use for even simple patterns. For example, Figure 1 shows some Humdrum data (after preprocessing to obtain explicit event continuations and extra spines with musical features) and a Humdrum pattern that captures the idea of a contrapuntal suspension. The regular expressions in each line of a Humdrum pattern do two things: i) match the beginning of events or the continuation of events (this is the purpose of brackets in the pattern of Fig. 1, a bracket matches the continuation of an event and the absence of a bracket matches the beginning of an event); and ii) match features of those events by matching corresponding values in additional columns (this is the purpose of the tokens ending the lines of the pattern, a dissonance feature is matched on the first line and, on the second line, features encoding the melodic contour of a step down and a consonance are matched).

\[
\begin{array}{cccc}
\text{bass} & \text{tenor} & \text{cont} & \text{qual} \\
D & B & +s & \text{cons} \\
BB & (B) & . & \text{cons} \\
FF# & (B) & . & \text{diss} \\
(FF#) & A# & -s & \text{cons} \\
\end{array}
\]

\[
[a-g A-G] + [- # n]* [^\text{\textcolor{red}{s}}] [\text{\textcolor{red}{c}}] [\text{\textcolor{red}{s}}] [\text{\textcolor{red}{n}}] [^\text{\textcolor{red}{\textcolor{red}{s}}}]
\]

\[
[a-g A-G] + [- # n]* [^\text{\textcolor{red}{s}}] [\text{\textcolor{red}{c}}] [\text{\textcolor{red}{s}}] [\text{\textcolor{red}{n}}] [^\text{\textcolor{red}{\textcolor{red}{s}}}]
\]

Fig. 1. A musical fragment in Humdrum (top) and a suspension pattern (bottom). The musical data is preprocessed to obtain explicit event continuations, denoted by brackets, and to add extra spines containing musical features: cont for melodic contour and qual for harmonic interval.

In addition to the required expertise in regular expressions, more fundamental difficulties with developing Humdrum patterns arise due to their opacity and lack of a denotational semantics. It is in general not possible for a computational musicologist to inspect a pattern and from that deduce what fragments will be matched. Lacking a denotational semantics, it may be intractably difficult to develop and refine patterns. For example, simply adding another melodic feature to a pattern requires both the existence of a spine containing that feature and, to reference the values of that feature, the modification of every line of the pattern in a way that respects the spine ordering.

This paper formally defines the semantics of \( \mathcal{VVP} \) showing how it expresses relations between temporally related events in compact regions of time. The method offers a clear syntax for patterns that are very similar to the patterns that Humdrum can express. In addition, it is possible to formally define the notion of pattern subsumption in \( \mathcal{VVP} \), indicating when the pieces matched by one pattern are always a superset of the pieces matched by another. With a formal