Substructure Discovery of Symbolic Musical Information using the Minimum Description Length Principle

Tang-Chun Li
Faculty of Music, McGill University
555 Sherbrooke Street West, Montreal, H3A 1E3, Canada
litc@music.mcgill.ca

Abstract
This paper presents a model for discovery and organization of recurring patterns and substructures in a music piece based on the Minimum Description Length principle. It describes a new system of how to encode, match, and organize a large symbolic musical sequence for patterns and substructures searching in machine recognition of music and music database.

1. Introduction and Related Works

Music content-based retrieval and machine recognition of music share the same difficulty in organizing and matching musical substructures in a systematic way. Some current content-based retrieval systems are limited to only certain types of music and assume availability of many meta-events types of information. (Chou et. al., 1996) Recent research on machine recognition of music shows that patterns are one of the key elements to the understanding of music. (Simon and Sumner, 1968; McAdams, 1989; Rowe, 1993) Previous research emphasized on induction and recognition of short melodic fragments or motives. (Rowe, 1993; Stammen & Pennycook, 1993; Bakhmutova et. al., 1997) Although these are useful techniques for comparing new input with stored segments, it is not very efficient for representation, organization, and comparison of works in a database. This is because structural organization at the level of the whole piece was not addressed. This paper describes an alternative top-down matching method that reduces the storage size and matching time. Particularly, we emphasize on the preprocessing of input sequence to discover substructures. Later we will show that this preprocessing is also a sequence matching process. The result is a compressed representation that is easy to search at various levels.

This on-going research project is based on earlier work on music pattern processing and melodic fragments matching of Pennycook et. al. (1993), Rowe (1993), Stammen & Pennycook (1993), Rowe & Li (1994) and others. In this system, we only deal with symbolic music data that is equivalent to the *note* concept. An ideal input data format would be the direct translation of a notated score into a MIDI stream played in real-time. This means three things: (1) meta-events information such as tempo, time signature, key signature and others is not required; (2) this system should be able to deal with real-time situation, currently not implemented; (3) normalization including quantization and time shifting is required for real life data.

2. Music Communication and Minimum Description Length Principle

Many theories about music describe a reduction process of internal representation during the listening or analyzing stage. Examples are the Western tonal music theory, the Schenkerian analysis, Lerdahl and Jackendoff (1983), and Bingand (1993). In our view, this reduction process can be viewed as a music communication problem. Music communication can be considered as an information communication between a sender and a receiver. The goal is to communicate the structural/syntactic and possibly the semantic information. Assuming only auditory information is available, the complexity of the notes to notes details usually is too high for a listener to cognize. The listener has to engage a parsing system to create an internal structure that is within his maximum manageable complexity or abandons the communication. The *best communication* then is defined as the one that can describe the observed sequences with the shortest length. The above definition logically follows several independent works in serial pattern research (Simon, 1972; Leeuwenberg, 1971; van der Helm et. al. 1991) and in computational science (Rissanen, 1989; Li & Vitanyi, 1997). In his study of complexity of serial pattern processing, Simon concludes that the complexity of a sequence is the code length. Meanwhile, Leeuwenberg adopts the minimum principle (Hochberg & McAlister, 1953) which states that the preferred interpretation of a pattern is the simplest description of that pattern. The minimum principle is consistent with the Gestalt Psychology.

Our system adopts Rissanen's theory (1989) of the Minimum Description Length (MDL) principle. The MDL principle says that the best theory to explain a data set is the theory that has the shortest description. This shortest description includes the length of the theory that explains the observed data, plus the length to encode the data according to the theory.

Note that when we apply the MDL principle to a musical sequence, there are two levels of encoding. The first level, or the *basic encodin*g, encodes the musical sequence into a preferable processing format for the MDL principle. The basic encoding is determined by the musical knowledge equipped within the system. A system with domain knowledge will perform better than a system without. The second level is the recursive substructure discovery due to the MDL principle directly. Music knowledge can also be used to assist the MDL matching to induce high-level substructures. In this paper, we focus on the second level of encoding and assume that we have the same musical knowledge for all input pieces.

3. System Overview

The following assumptions are adopted for the system's operations: (1) The input music itself is a congruent entity in which the progression should be consistent within itself. (2) Input sequence consists at least the notes information. Meta-events information is assumed to be unavailable. (3) All values of any parameters used in the basic encoding are within a fixed range. For example, all intervals fall into the range of {-24, 24}.

There are three types of databases for patterns and segments separately. All symbols used at the level of the basic encoding are simple (or terminal) symbols. Any new symbol corresponding to a pattern is a compound (or nonterminal) symbol. Patterns or segments in the *basic database* consist of only simple symbols. In the *mixed database* they consist of both simple and compound symbols. In the *symbol database* they consist of only compound symbols. Patterns in databases are linked to their direct superset patterns or segments.

The system operates in three stages. In the *normalization and basic encoding* stage, the input sequence is normalized, segmented, separated, and encoded into a vector or a matrix of symbols. Normalization refers to the quantization of note duration and time shifting to synchronize with time unit grid. Related research is discussed in (Desain and Honing, 1994). Here we adopt the approach in (Pennycook et. al., 1993). Segmentation at this stage is based on gaps of rest(s). A further refinement is in next stage. Pattern matching and searching starts sequences for one instrument first. When many channels are squeezed into one track (such as MIDI file Format 0), a separation task is required. Finally, each segment is encoded as a string of unique symbols. These symbols correspond to the intervals and other tokens in the segment.

In the next *segment induction* stage, each segment is divided into finer segments by applying a revised version of the Segmenter algorithm implemented in (Pennycook et. al., 1993). This program also follows the Grouping Preference Rules in (Lerdahl and Jackendoff, 1983).

The last stage is the *preprocess-match* stage. While preprocessing the input sequence, the program also searches for all potential patterns. It can split into two steps: (1) match and substitute with the pattern and segment databases; (2) scan segment for internal patterns at last. The program first matches patterns in the basic database, then stores or substitutes them as new or previously stored symbol. When an existing pattern is found, it then matches with the direct superset patterns/segments until either a match is found or all patterns/segments are exhausted. If a pattern is found, then this new segment is replaced by the symbol of the pattern. If a segment is found, then this segment is registered as a pattern and assigned a symbol. After matching with the database, the program scans the input segment for its internal patterns. Eventually, if the program finds nothing, the segment is registered as a segment in one of the three segment databases.

When the preprocessings for all instrumental lines (in horizontal direction) are done, then a vertical grouping and substructure discovery is applied. Vertical grouping is determined by synchronized root notes of segments in all lines. Groups created by the vertical grouping are called basic groups. Each basic group is considered an atomic unit. Member segments from different lines in a basic group are overlapping with each other. For this reason, matching among basic groups is to compare them in entirety. The final output is a maximally compressed network (or tilted tree) of symbols.

After the preprocessing of all pieces in the database, any two musical works in the database can be compared in a top-down manner. Lower level symbols are compared only when necessary.

4. Minimum Description Length Encoding of a Music Segment

The system encodes the following basic information: channel, pitch, velocity, duration, start time, and end time. All other information is inferred from the input data. Meta-events information is not used.

Principles of encoding: (1) P1. Prefer no information lost during the encoding unless P2. (2) P2. When one object is identified *cognitively* as the same as another object even though they are not entirely the same, we can substitute one with the other.

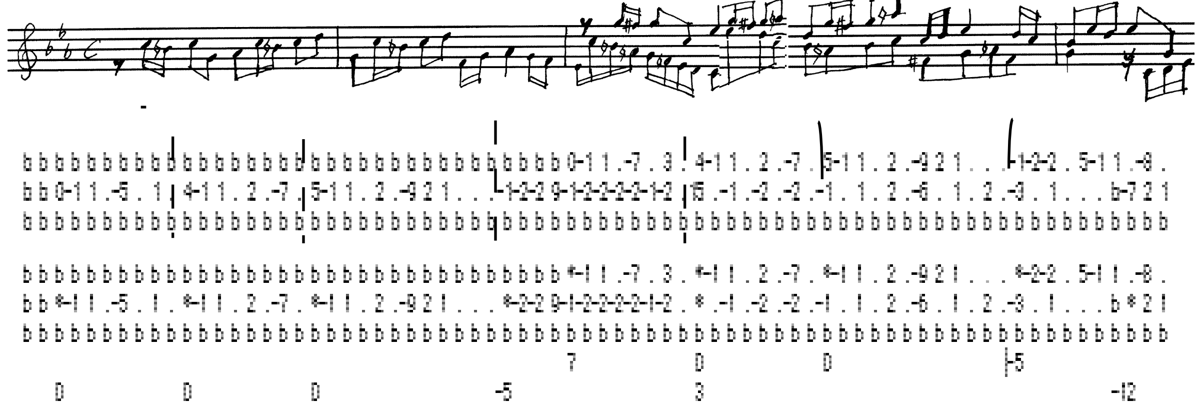
Since the encoded sequence will be used to represent the symbolic sequence of a musical input such as a MIDI file, information should be preserved as much as possible so that a cognitively equivalent version of the original can be restored. For example, a live performance piece in MIDI stream will lose some of the timing information after the basic encoding process, but the restored one should be equivalent to the version notated in the score.

Encoding rules for a monophonic line: (1) R1. An alphabet of unique symbols is used to encode all values in all parameters. Each symbol has a constant cost, u bits in length. These symbols have precedence relations. However, only those that are used to encode melodic intervals and velocity parameters will use their precedence relations. We can think of them as positive and negative integers. (2) R2. The start note of the piece has a relative pitch 0 with its original velocity. The root note of a segment has a relative pitch equal to the interval between itself and the start note. Only root notes of segments have pitch information. (3) R3. All segments are encoded as a vector of intervals except the root note, which use a special root symbol. (4) R4. Velocity vector is encoded by velocity differences. Directional information of dynamic change is embedded in the difference values. (5) R5. Timing information is preserved in the vector as the index.

For the encoding of a polyphonic line, replace all vectors as matrices. (6) R6. The lowest notes at each time unit form a bass line. This bass line is stored as an interval vector. Music notes that are vertically above any notes in the bass line are represented by its interval from the bass note. If there is no note above, fill in a rest symbol.

Note that in our system the velocity information is only used to assist the decision making of the pattern matching of the melodic patterns. Interval and duration serve the main roles.

Example: Fugue 2 in C minor, BWV 847: (1) score; (2) basic encoding with segmentation info; (3) input string for the pattern matching and substructure discovery.



According to the MDL principle, the theory that best supports a data set is the one that minimizes the Description Length, $DL(DB,P) = L(P) + L(DB \mid P)$, or, simply $L(DB)$, where P is the patterns discovered, DB is all the music sequences in the system, $L(P)$ is the number of bits required to encode all the patterns, and $L(DB \mid P)$ is the number of bits needed to encode all the sequences. In brief, as long as the basic encoding scheme is set, the best theory should the one that picks the patterns those minimize the length. Next we will prove that our system always minimize the description length when a new input sequence has been added into our database.

Definition: A local minimum is defined as the pattern set that matched with the segment and minimizes the description length of the $DB \cup s$, where s is the segment. The occurrences of the new pattern can be associated with the previous pattern(s) as: (1) *independent*, (2) *linked component,* (3) *subpattern*, (4) *superset*. We call these the *relations* between patterns. Let T be a set of sequences; $F(T,s)$ is the function to evaluate the description cost after the processing of s.

Theorem 1. Any newfound pattern will either decrease or maintain the current description length. If the pattern is longer than two or occurs more two times, the description length will decrease.

Proof. Let l be the total length of sequences. X is a pattern of length k and occurs m times in the sequences, including the local segment. The encoding length of the sequences before the pattern processing is $L = lu$. The new coding length is $L' = (l - mk)u + mu + ku \;= (l + 1 + (1-k)(m-1)) * u$.
Since by definition, $k \geq 2$ and $m \geq 2$, so $L' \leq lu = L$. When $k > 3$ or $m > 3$, $L' < L$. ###

Lemma 1. The coding length evaluation function F is order independent for independent patterns.
Lemma 2. The coding length evaluation function F is order independent for linked related patterns.
Lemma 3. The coding length evaluation function F is order independent for subset-superset related patterns.

Theorem 2. Many different pattern sets are available for interpreting the input sequence *s*. Let P_S be the set of new patterns found in *s* and *DB* that will lead to the shortest description for *(DB, s)*. Then P_S is the union of the patterns that reach the local minimum in each segment or substructure.

Because of the limited space, we leave the proof to the readers.

5. Discussions

Substructure Discovery: Regardless its length, any pattern in the system is assigned a symbol. Our assumption is that the amount of symbols is not large. However, when the database is large, the assumption of constant cost of symbol is no longer true. Since symbols required for the basic encoding is much less, a better system is to have a small, constant cost of symbols used for the basic encoding, and a separate symbol set for patterns. Another question we haven't addressed in this paper is the similarity measurement between two large substructures or sequences. Further investigations are required. *Matching and inexact matching:* The current version of the inexact matching is done by the dynamic programming as in the (Stammen and Pennycook, 1993) Currently it only applies to one-dimensional matching. We examine sequences by three methods: self-matching, repeated symbols, and matching with others. *Limitations of the current basic encoding program:* for certain types of input sequence, the current version of basic encoding program does not encode correctly. This is caused by at least two factors: in live performance piece the performer often changes tempo and plays early or later then what is notated in the score; another is the expressional nature of MIDI sequence. It only records the actions of performer, not the actual sound played by the MIDI instruments. Since each sampled instrument has its own envelope, the recorded MIDI sequence might not match with what is notated on the score.

6. Domain Knowledge and Conclusion

In this paper, a pilot system for substructure discovery of symbolic musical data based on the Minimum Description Length principle is discussed. The resulting encoding/representation of musical data can be used for machine recognition of music and content-based retrieval in music database. Specifically, we consider a music communication situation where the goal is to communicate the structural/syntactic and possibly the semantic information. The MDL principle provides a good theory for implementing a system of such a communication nature. We also describe the matching process of the system, its encoding scheme, and the representations of the sequences. Some limitations and problems are also discussed. Finally, although the system is data-driven than rules-driven as in many intelligent music systems, specific domain knowledge could be incorporated into the system for inference. A future version will include more music knowledge to improve both the basic encoding and the normalization.

References

Bakhmutova, I. V., V. D. Gusev, T. N. Titkova. 1997. The search for adaptations in song melodies. *Computer Music Journal,* 21(1), 58-69.

Bigand, E. 1993. Contributions of music to research on human auditory. In *Thinking In Sound: The Cognitive Psychology of Human Audition*, ed. S. McAdams and E. Bigand, Clarendon Press, Paris.

Chou, Ta-Chun, Arbee L.P. Chen, and C.-C. Liu. 1996. Music databases: indexing techniques and implementation. In *Proceedings of the International Workshop on Multi-Media Database*.

Desain, P., & Honing, H. 1994. From foot tapper systems to beat induction models. *Proceedings ICMPC*. Liege: ESCOM.

Hochberg, J. E., and E. McAlister. 1953. A quantitative approach to figural goodness. *Journal of Experimental Psychology,* 46, 361-364.

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

Leeuwenberg, E. L. J. 1971. A perceptual coding language for visual and auditory patterns. *American Journal of Psychology*, 84(3), 307-349.

Li, Ming, and P. Vitanyi. 1997. An Introduction to Kolmogorov Complexity and Its Applications. 2nd. Ed. New York: Springer.

McAdams, S. 1987. Music: a science of the mind? *Contemporary Music Review*, 2, 1-61.

Pennycook, B., D. Stammen, and D. Reynolds. 1993. Toward a Computer Model of a Jazz Improviser. In *Proceedings: 1993 International Computer Music Conference*.

Rissanen, J. 1989. *Stochastic Complexity in Statistical Inquiry*. Singapore: World Scientific.

Rowe, R. 1993. *Interactive Music System*. Cambridge, Mass: MIT Press.

Rowe, R., and T.C. Li. 1994. Pattern processing in music. In *Proceedings: 1994 International*

Stammen, D., and B. Pennycook, 1993, Real-time Recognition of Melodic Fragments Using the Dynamic Timewarp Algorithm. In *Proceedings: 1993 International Computer Music Conference*.

Simon, H. A. 1972. Complexity and the representation of patterned sequences of symbols. *Psychological Review*, 79(5), 369-382.

Simon, H., and Richard Sumner. 1969. Pattern in Music, from *Formal Representation of Human Judgment*, pp. 219-251.

van der Helm, Peter A, Rob J van Lier, and Emanuel J Leeuwenberg. 1992. Serial pattern complexity: irregularity and hierarchy. *Perception*, 1992, vol. 21, pp. 517-544.