Real time Pattern Based Melodic Query for Music Continuation System

Sanjay Majumder
Indiana University - Purdue University Indianapolis
Indianapolis, Indiana, USA
sanjay@iupui.edu

Benjamin D. Smith
Indiana University - Purdue University Indianapolis
Indianapolis, Indiana, USA
bds6@iupui.edu

ABSTRACT

This paper presents a music continuation system using pattern matching to find patterns within a library of MIDI files using a real-time algorithm to build a system which can be used as interactive DJ system. This paper also looks at the influence of different kinds of pattern matching on MIDI file analysis. Many pattern-matching algorithms have been developed for text analysis, voice recognition and Bio-informatics but as the domain knowledge and nature of the problems are different these algorithms are not ideally suitable for real time MIDI processing for interactive music continuation system. By taking patterns in real-time, via MIDI keyboard, the system searches patterns within a corpus of MIDI files and continues playing from the user's musical input. Four different types of pattern matching are used in this system (i.e. exact pattern matching, reverse pattern matching, pattern matching with mismatch and combinatorial pattern matching in a single system). After computing the results of the four types of pattern matching of each MIDI file, the system compares the results and locates the highest pattern matching possibility MIDI file within the library.

CCS CONCEPTS

• **Information systems** → **Information retrieval query processing**; • **Theory of computation** → **Pattern matching**; *Approximation algorithms analysis*; • **Applied computing** → **Sound and music computing**; • **Computer systems organization** → *Real-time systems*;

KEYWORDS

MIDI pattern matching, music information retrieval, interactive music continuation system,exact pattern matching, reverse pattern matching,approximate pattern matching, combinatorial pattern matching

ACM Reference Format:
Sanjay Majumder and Benjamin D. Smith. 2018. Real time Pattern Based Melodic Query for Music Continuation System. In *Audio Mostly 2018: Sound in Immersion and Emotion (AM'18), September 12–14, 2018, Wrexham, United Kingdom.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3243274.3243283

AM'18, September 12–14, 2018, Wrexham, United Kingdom

ACM ISBN 978-1-4503-6609-0/18/09.
https://doi.org/10.1145/3243274.3243283

1 INTRODUCTION

The MIDI file format is one of the most popular and broadly applied forms of symbolic music representation. Despite having limitations, its wide adoption has resulted in many, many MIDI databases that are available on the Internet. With the rapid advancement of Digital Audio Workstations and the advent of inexpensive MIDI controllers and keyboards, musicians can easily compose, distribute and share MIDI files.

Pattern matching is a primitive problem in the field of computer science as well as in information retrieval. With the advancement of computer science, researchers have developed many pattern-matching algorithms, but most of these algorithms are problem specific. These algorithms are developed for voice recognition, text matching and in bioinformatics for DNA or RNA matching.

Music information retrieval is one of the popular topics in field of music and technology. Many researchers have developed computational representation system of Music and MIDI as well as algorithms for music and MIDI pattern matching. But these algorithms have some weakness [2, 3, 14]. In text matching a fixed number of letters and spaces between words are primary factors. With the help of the space between two words, the system can easily detect them as different instances. In Gnome, DNA and RNA pattern matching of bioinformatics, there is a fixed number of letters (A, T, G, and C) that are used for matching. MIDI, on the other hand, is a continuous file without spaces and with a large number of tokens. In MIDI pattern matching, every note event has equal importance in terms of pattern matching. So, taking the main logic from these algorithms, this paper implements pattern matching algorithms for finding patterns within MIDI files of a MIDI library. In this paper, exact pattern matching, reverse pattern matching, pattern matching with mismatch and combinational pattern matching are implemented to support a real-time music continuation system. Taking live input from a musician through a MIDI keyboard, this system searches a corpus of MIDI files, finds the best match, and continues playing when the musician pauses. In order to account for user mistakes or inaccurate musical input combinatorial pattern matching and approximate parent matching help the user to find the desired content.

2 RELATED WORK

A lot of work has been done to find a pattern within a MIDI file, as well as implementations of pattern matching for text matching, speech recognition, and bioinformatics. Other work has been done to find the desired MIDI file in a large MIDI database.

A system [15] with content-based retrieval has been used to find the same music composition with a different version of a song

This is the author's manuscript of the article published in final edited form as:
Majumder, S., & Smith, B. D. (2018). Real Time Pattern Based Melodic Query for Music Continuation System. Proceedings of the Audio Mostly 2018 on Sound in Immersion and Emotion, 35:1–35:5. https://doi.org/10.1145/3243274.3243283

within a large MIDI database. In the first step, MIDI data preprocessing is done by parsing MIDI note events and melodies. By taking the concept from Bag-of-words representation (a text matching algorithm), note distributions are found. A query by humming system is used to input the melody's features. Clausen et. al. [4] use content based retrieval approach for MIDI and audio. The authors introduce two search engine frame-works, MiDiLiBi project for audio and Notify! for MIDI, for searching a database. While searching in the database, only MIDI note events are considered. In addition, the metrical position of the notes is also taken into account and this metrical position and pitch is used to index the notes. These two parameters are then compared in the database to find the most similar MIDI file in the data-base. Another content-based retrieval paper [11] uses graph structure to the measure similarity between melodic content, symbolic music.

Raffel et. al. [13] introduce a system based on convolution network-based cross-modality hashing scheme which can effectively match and align MIDI files corresponding to their audio content to in a large corpus of audio.

Dovey [8] tries to deploy an Online Music Retrieval and Searching project for polyphonic music searching. A piano roll based algorithm is used to identify a polyphonic query within a large polyphonic music library. In the piano roll based algorithm, polyphonic music is represented as a piano roll where punched holes signify the playing of a note and the horizontal position in the roll signifies playing time of that note. A command line test framework named OMARS is used to test different search algorithms.

In recent years, combinational pattern matching and approximate pattern matching have gained the attention of researchers. An approximate pattern matching algorithm for text matching tries to solve the string matching problem by reducing time complexity in a practical way [1]. First, text strings are divided into words or blocks by the appearance of spaces. Comparison is done on each word with the given word by character and by finding the matched character(s). If there is any match between characters it will count as 1, otherwise it will be counted as 0. If a character occurs more than once in a word, it is considered as distinct and it is counted. Then it counts the total characters matched in each word. Collins et. al. [6] combine pattern discovery algorithm and inexact pattern matching algorithm to solve the inexact pattern matching problem of the current geometric intra-opus pattern discovery approach within a piece of music.

Different types of approximate string matching algorithms have been applied to text matching. Researchers try to deploy some of these algorithms in music pattern matching. Clifford et. al. [5] discuss the advancement of approximate string matching and demonstrate the use these algorithms on string based music representation i.e MIDI. This paper gives an overview on some widely used approximate string matching algorithms with example and then show their practical application in music. The paper also discuss about the time complexity of different types of pattern matching algorithms. Kline et. al. [10] conduct an experiment to show the error rate of query by humming technique is with for the existing algorithm to find the voice matching. This paper also develops an approximate pattern matching algorithm which improves the search result for error filled vocal user quires within a music library. Crochemore et. al. [7] discussed about three version of Boyer-Moore algorithms δ-BM1, δ-BM2 and δ-BM3 for approximate pattern matching in music score. This paper uses (γ, δ)- matching where γ is a bound on total number of errors and δ is the version of suffix tries and sub-word graphs.

Pachet [12] presents an interactive music continuation system using Markov model of musical styles to resolve the limitations of the user's management of rhythm, beat, harmony, and imprecision. Another paper [9] describes a real time system which takes musical phrases from a MIDI keyboard, converts it in a score level description and then automatically rendered it as a graphic score.

As the Music Information Retrieval field growing rapidly, few works have been done to identify the issues of pattern matching algorithms. Two paper [2, 3] discuss about the existing issues of musical pattern matching and retrieving pattern from musical strings. Velardo et. al. [14] provide a relative analysis of existing algorithms that are used to calculate symbolic melodic similarity based on eight criteria.

So far, different papers use different techniques for searching desired music content in a database or online. Content based searching is widely used for music searching in a database [4, 11, 15] Approximate pattern matching and combinatorial pattern matching are rarely used for music information retrieval though they are already used in text pattern matching [1]. This paper implements these two types of pattern matching along with the exact pattern matching and the reverse pattern matching to retrieve music from a library of MIDI files and also evaluates and compares their significances.

3 SYSTEM OVERVIEW

Pattern matching within a MIDI files searches every occurrence of a given pattern in several ways within the MIDI file database. The aim of this work is to build an alternative music continuation system which is interactive as well as cost effective. To build a system for music continuation/ DJ, time complexity is a great factor. The second aim of this work is to build the system which time complexity is lower or equal to a DJ system while moving one track to another, without compromising the accuracy of the pattern matching. The final aim is to evaluate the efficiency of each type of pattern matching while finding pattern in the MIDI library.

The whole work is divided into two sections. First, the MIDI files and user input are transcript and then, with the help of the user input pattern, pattern matching is done. In the data transcription phase, note lists are extracted from the MIDI files, and then a MIDI keyboard takes the user input. The user data is then processed and saved as a pattern. After getting the user pattern, different types of pattern matching (i.e. exact pattern matching, reverse pattern matching, pattern matching with mismatch and combinatorial pattern matching) are performed on the MIDI files. For each type of pattern matching a file will be constructed to represent the matching. After getting all the individual results for each pattern matching, a final list will show the cumulative results which represent the highest matching possibility of each position in the MIDI files of the library. Then the system will sum all the matching probabilities and the highest match music will be played via Digital Audio workstation. Figure 1 illustrates the overview of the proposed system.

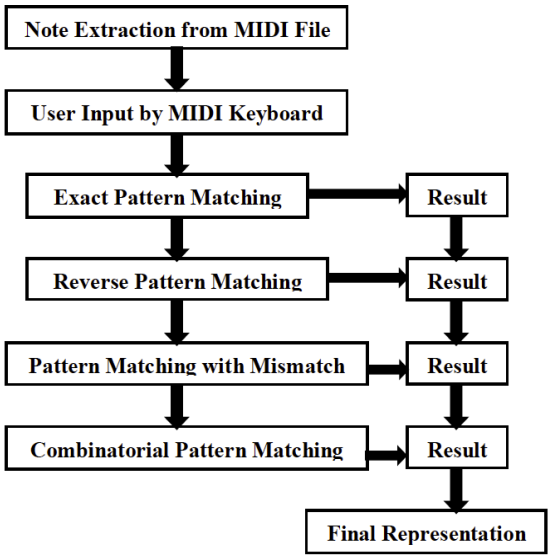

Figure 1: Proposed system.

4 DESCRIPTION OF THE SYSTEM

In this paper, only MIDI note events are used for the pattern matching. To extract the MIDI note events, the MIDI files of the MIDI library are read and all the notes of each MIDI are stored in separate files that are used by the pat-tern-matching model.

A MIDI keyboard is used to take the real-time pattern as user input. The system will wait for 3 seconds of 'silence 'for user inputs before searching the database. After playing a query of MIDI notes, the system will calculate the length of MIDI notes and the system will save it as a pattern. If the user uses any other system control, for example the mod-wheel or foot pedal the system, the system will discard that information. The pattern matching process is done with four different types of pattern matching. For each type of pattern matching, a file is constructed to represent the matching in accordance with the position of the MIDI file. The user can give input via MIDI keyboard any time while the system is calculating the patter matching or the file with highest match played via Digital Audio Workstation. The system will take the new input as a new pattern and search again.

4.1 Exact Pattern Matching

In exact pattern matching, the given pattern will be matched in the MIDI file. First the system will calculate the length of MIDI file and user pattern. Then the system will try to match each note of the pattern with the MIDI file according to position. The program tries to match the first note of the pattern with the first note of a MIDI file. If first note matched, the second note of the pattern will be matched and so on. If a pattern is matched, then the initial position will be denoted by 1. If the first note is not matched, the position will be denoted by 0 and the system will try to match the first note of the pattern with the second note and so on. In that way, a file will be constructed which has the same length of the MIDI file but it will contain a bit-map with the exact pattern probability for each position.

4.2 Reverse Pattern Matching

In reverse pattern matching, the reverse order of the given pattern is matched within the MIDI file. In case of reverse pattern matching, if the reverse pattern is matched, then the initial position will be denoted by 0.75. If the reverse pattern is not matched, then each position will be denoted by 0. In that way, a file will be constructed which has the same length of the MIDI file but it will contain the reverse pattern matching probability for each position. A reverse match is considered less desirable than an exact match, hence the 0.75 weighting for reverse matches.

4.3 Pattern Matching with Mismatch

Approximate pattern matching is one of the major challenges in pattern matching problems. Approximate pattern matching algorithm compares a data sequence with a pattern and find the patterns with N number of Mismatch.

If D is the MIDI Note sequence, P is the pattern and N is the number of mismatch,

D = 45, 37, 65, 46, 78, 34, 89

P = 65, 36, 78, 34

Result: 45, 37, 65, 46, 78, 34, 89.

N = 1

In this section, an algorithm is implemented for MIDI pattern matching with N number of mismatches. All in-stances of a MIDI pattern will be matched in a MIDI file with N number of mismatches for each instance of the MIDI pattern in the MIDI file. Here N starts with one unit less than the length of the pattern. The amount of mismatch will be calculated dynamically by the system. If N is the number of mismatch and q is the length of the pattern and u is the amount of match, then

$$u = ((N-2) - q)/N \tag{1}$$

The threshold of mismatch is

$$q - N >= 2 \tag{2}$$

which means at least 2 notes should be present in the mismatch.

To find the patterns with mismatch, the system checks the number of mismatch allowed. Then it tries to match the first note of the pattern with the first note of a MIDI file. If first note matched , it will update the matched note counter. And then go to the next position of the MIDI file. If first note not is matched it will update the mismatch counter with 1. The system will check the current number of mismatches allowed. If the number of mismatch allowed is True, the system will try to match the second note of the pattern with the second note of the MIDI file. If second note matched the system updates number of matched note. It also matches the first note of the pattern with the second note of the MIDI file to check whether or not they match. If the number of mismatch exceeds, the program ends the first instance of pattern matching but continues the second instance which is running from the second note of the MIDI file and repeat the whole process.

The number of mismatches is computed dynamically by the system. For each number of mismatch, the system perform the above mentioned process.

4.4 Combinational Pattern Matching

In combinational pattern matching, the system takes different combinations of the given pattern and tries to find each occurrence of each combination in the MIDI files. First, the system calculates the length of the pattern and performs permutations of the pattern. Then the system follows the exact pattern matching algorithm for each combination to find matches. If any matches are found, the system will denote it with 0.50.

4.5 Final Representation

The final result represents the whole MIDI file according to the highest matching of each position for the different pattern matching types. In this phase, the value of each position in the different pattern matching results will be compared and the highest pattern matching value will be presented. The MIDI file with cumulative highest match will be played via a DAW.

5 SYSTEM EVALUATION

The evaluation of the system is divided into three sections. First, it is evaluated by the efficiency of the system for use as a music continuation / DJ system to play music files from a MIDI library. Next, the significance of the four different pattern matching algorithms is evaluated to find their contribution in the whole pattern matching of MIDI files within MIDI library. Finally, the efficiency of the algorithms are evaluated.

5.1 Data Set

For evaluating the system, a library of 100 MIDI files are taken as data set. This data set comprises western music, Irish music, and Indian music. Table 1 shows the data set that are used for system evaluation.

Table 1: Data Set for System Evaluation

Genre	Number of MIDI Files
Western Hip-hop	20
Western Dance music	20
Irish Music	20
Indian Dance Music	20
Contra Dance Music	20

To evaluate the system, 10 users from Music Technology department are selected. 3 of them have strong technical knowledge with less musical knowledge. Other 3 users have strong music knowledge with less technical background. Rest 4 users have equal knowledge both music and technology. The experience of these three groups of users help to determine the system 's efficiency as music continuation system as well as the Contribution of four types of pattern matching in this music continuation system.

5.2 Music Continuation System

The MIDI library of the 100 MIDI files are taken to test the effectiveness of the current system as a music continuation system. A MIDI keyboard is used to take in-put/pattern. The user changes track by using only a small keyboard, CME Xkey. According to the evaluator, this system simplifies the current DJ/ music continuation system and it can be a good alternative of current for mu-sic continuation system. According to the user, this pro-posed system will removes the barrier of buying and carrying large and costly DJ equipment.

5.3 Contribution of Four Types of Pattern Matching

In this part, the contribution and significance of four types of pattern matching in a MIDI library are evaluated. The system is evaluated by taking 100 patterns via MIDI keyboard and the number of occurrences identified by each pattern algorithm are counted. The mean weight for each algorithm across all 100 database searches is calculated to reflect the actual use of each option. Table 2 shows the number of occurrence and mean of each type of pattern matching for 100 patterns.

Table 2: Result of Four Types of Pattern Matching

Pattern Matching Type	Number of occurrence	Mean
Exact Pattern Matching	27	0.27
Reverse Pattern Matching	22	0.22
Pattern Matching with Mismatch	317	3.17
Combinatorial Pattern Matching	19	0.19

From these 100 patterns, 27 times the input pattern was exactly matched in the MIDI library and the reverse pat-tern was matched 22 times. On the other hand a total of 317 approximate matches were found in the MIDI library. While the exact and reverse searches were useful around 20% of the time, the combinatorial search only identified 19 matches and contributed 9.5% of the matching weight. This was unexpected as its impact was anticipated to be much more significant. On the other hand, the mismatch search had the highest impact, identifying the most results. From this analysis it is clear that approximate pattern matching has the highest influence compared to the other types of pattern matching. Furthermore, the exact pattern matching works well with the system evaluators who have strong musical background. For other types of system evaluators, approximate pattern matching seems to be helpful to play with the system.

5.4 Efficiency of the system

To evaluate the efficiency, the Boyer-Moore algorithm and Rabin-Karp algorithm were implemented for pattern matching in MIDI files within a MIDI library from the method described by Clifford and Ilopoulos [5]. The mean time complexity of the our original system is 0.74 second where the mean time complexity of the system using Boyer-Moore algorithm is 0.81 second and Rabin-Karp algorithm is 0.87 second.

6 FUTURE WORK

To minimize the time complexity and to increase the efficiency of the system, Knuth-Morris-Pratt algorithm, and Bitap algorithm will be implemented for pattern matching in MIDI files . By implementing these algorithms, time complexity will be analyzed for each

algorithm, thus it will be easy to find the most efficient algorithm for different kinds of pattern matching in a MIDI file.

A number of other application areas exist which will be pursued and examined. Live performance and improvisation with human musicians as well as other artificial agents is a distinct possibility for the system described herein.

7 CONCLUSION

Pattern matching is a challenging task for Music information retrieval. MIDI is a popular file format in music, and is readily available for pattern matching applications. Many researches have been done to find the desired MIDI file in a database but using pattern matching within a MIDI corpus for an interactive music continuation system remains as an area for improvement. So far, less researches have been done to examine the possibilities of using approximate pattern matching and combinatorial pattern matching to find the desired music file in a database. The possibilities therein motivate us to do research on pattern matching in a MIDI library and to that end, this paper is introduced.

REFERENCES

[1] Ricardo A. Baeza-Yates and Chris H. Perleberg. 1996. Fast and Practical Approximate String Matching. *Inform. Process. Lett.* 59, 1 (July 1996), 21–27. https://doi.org/10.1016/0020-0190(96)00083-X

[2] Emilios Cambouropoulos. 2000. Extracting 'Significant' Patterns from Musical Strings: Some Interesting Problems. In *Invited paper presented at London String Days (LSD) 2000*. King's College London.

[3] Emilios Cambouropoulos, Tim Crawford, and Costas S. Iliopoulos. 2001. Pattern Processing in Melodic Sequences: Challenges, Caveats and Prospects. *Computers and the Humanities* 35, 1 (February 2001), 9–21.

[4] Michael Clausen, Frank Kurth, and Roland Engelbrecht. 2001. Content-based Retrieval in MIDI and Audio. *ECDL WS Generalized Documents* TR-120407 (2001).

[5] Raphael Clifford and Costas Iliopoulos. 2004. Approximate string matching for music analysis. *Soft Computing* 8, 9 (September 2004), 597–603. https://doi.org/10.1007/s00500-004-0384-5

[6] Tom Collins, Andreas Arzt, Sebastian Flossmann, and Gerhard Widmer. 2013. SIARCT-CFP: Improving Precision and the Discovery of Inexact Musical Patterns in Point-set Representation. In *Proceedings of the 14th International Conference on Music Information Retrieval*. Music Information Retrieval Society.

[7] Maxime Crochemore, Costas S. Iliopoulos, Thierry Lecroq, Wojciech Plandowski, and Wojciech Rytter. 2002. Three Heuristics for δ-Matching: δ-BM Algorithms. In *Proceedings of the 13th Annual Symposium on Combinatorial Pattern Matching*. 178–189.

[8] Matthew J. Dovey. 2001. A technique for "regular expression" style searching in polyphonic music. In *Proceedings of the 2nd International Symposium on Music Information Retrieval*. Music Information Retrieval Society.

[9] D. Fober, J. F. Kilian, and F. Pachet. [n. d.]. Real-Time Score Notation from Raw MIDI Inputs. *GRAME Computer Music Research Lab, Technical Report* TR-120407 ([n. d.]).

[10] Richard L. Kline and Ephraim P. Glinert. 2003. Approximate Matching Algorithms for Music Information Retrieval Using Vocal Input. In *Proceedings of the 11th ACM international conference on Multimedia*. 130–139.

[11] Nicola Orio and Antonio Rodà. 2009. A measure of Melodic Similarity Based on a Graph Representation of the Music Structure. In *Proceedings of the 10th International Conference on Music Information Retrieval*. Music Information Retrieval Society.

[12] François Pachet. 2003. The continuator: Musical interaction with style. *Journal of New Music Research* 32, 3 (2003), 333–341. https://doi.org/10.1076/jnmr.32.3.333.16861

[13] Colin Raffel and Daniel P. W. Ellis. 2015. Large-Scale Content-Based Matching of MIDI and Audio Files. In *Proceedings of the 16th International Conference on Music Information Retrieval*. Music Information Retrieval Society.

[14] Valerio Velardo, Mauro Vallati, and Steven Jan. 2016. Symbolic Melodic Similarity: State of the Art and Future Challenges. *Computer Music Journal* 40, 2 (June 2016), 21–27. https://doi.org/10.1162/COMJ_a_00359

[15] Guangyu Xia, Tongbo Huang, Yifei Ma, Roger Dannenberg, and Christos Faloutsos. 2013. MidiFind: Fast and Effective Similarity Searching in Large MIDI databases. In *Proceedings of the 10th International Symposium on Computer Music Multidisciplinary Research*. 209–224.