SKEFIS – A SYMBOLIC (MIDI) KEY FINDING SYSTEM

Arpi Mardirossian

Daniel J. Epstein Department of
Industrial and Systems Engineering
University of Southern California
Los Angeles, CA
mardiros@usc.edu

Elaine Chew

Daniel J. Epstein Department of
Industrial and Systems Engineering
University of Southern California
Los Angeles, CA
echew@usc.edu

SKeFiS is a symbolic (Musical Instrument Digital Interface) key finding system that incorporates pitch spelling, key finding, and a cumulative window strategy for determining key.

After selection of the window to be considered, key recognition is considered a compound process: one of first determining the spelling of the pitches from numeric pitch information (MIDI), then extracting key from the pitch name information. By combining pitch spelling and key finding, we not only get the pitch class number of the tonic of the key, but also its letter name, and an appropriate key signature for the piece. The window size (length of music) is determined from the test data provided by the contest organizers.

Figure 1 depicts the key finding process from MIDI input to key output, including the length parameter selection based on the sample test data provided. The compound process of pitch spelling and key finding is embedded in the system and outlined by dashed boxes.

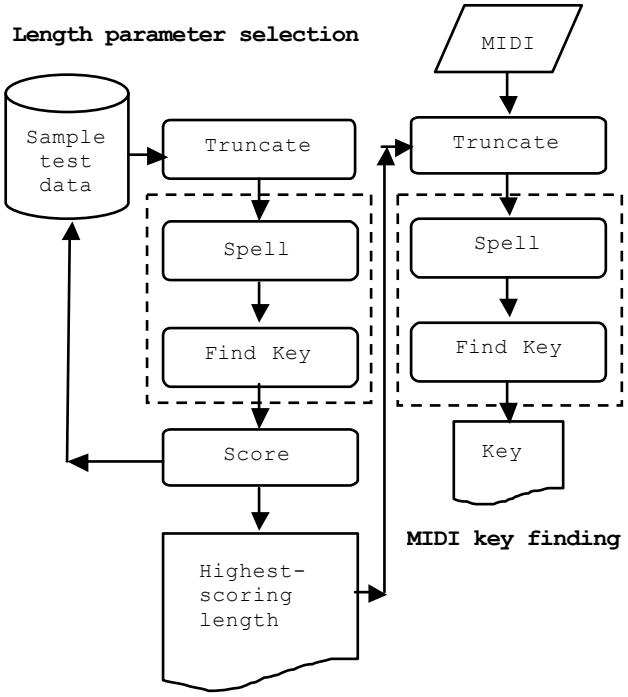

FIGURE 1 – SYSTEM DIAGRAM FOR KEY FINDING FROM MIDI

We first describe the selection of the stopping criteria, then we outline the pitch spelling and key finding algorithms implemented in this system (indicated by dashed boxes in Figure 1). The stopping criteria, or the length parameter, selection is based on the sample test data, and is described in Section 1. The methods for pitch spelling and key recognition are based on the Spiral Array model [1] and are described in Section 2.

1 LENGTH PARAMETER SELECTION

The rules for the audio and symbolic key finding competitions for the first Music Information Retrieval Evaluation exchange (MIREX) are outlined at [5]. In the implementation of these rules, 30-second segments from the beginnings of pieces are excerpted for testing by the contestants. Ninety six MIDI files, a sample representative of the evaluation corpus, were provided as a training set to all participants. The scoring scheme proposed was that a correct answer would get one point, an answer that was the perfect fifth of the correct one would get 0.5 points, and the relative major/minor and parallel major/minor errors would get 0.3 and 0.2 points respectively.

Given that 30 seconds of music are provided for each piece, and that our key finding method (the CEG algorithm) has been shown to require very little information to determine key [4], we decided to use only a subset of the 30 seconds of music that is provided. The question then arose as to how short or long of a segment did one need to determine key? In SkeFiS, our stopping criterion is based on selecting one optimal length for all pieces to be determined from the training data provided. We ran the CEG algorithm on truncated excerpts of the sample test files ranging in length from 0.1 through 30 seconds. We then compared the results against the ground truth to determine the score for each run. The percentage scores are plotted against the truncated segment lengths in Figure 2. The optimal segment length, having the highest score of 83.13%, was determined to be for segments that were 27.9, 28.0, and 28.1 seconds long. We chose to use 28.0-second segments.

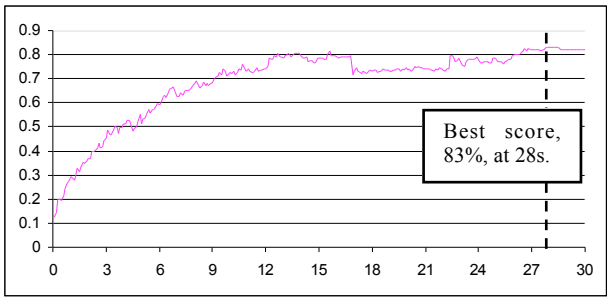

FIGURE 2 – GRAPH OF KEY FINDING SCORE VS LENGTH

2 KEY RECOGNITION

The pitch spelling and key recognition algorithms implemented in SkeFiS are based on the Spiral Array model [1], a mathematical model for tonality that uses spatial proximity to represent perceived closeness. The Spiral Array model consists of several nested spirals,

including representations for pitches and keys. Within this model, any collection of notes maps to pitch positions in the Spiral Array, and by taking a weighted average of these pitch representations, we can generate a center of effect (c.e.) that summarizes its tonal information. The key of the pitch collection is then determined by a nearest neighbor search for the closest key presentation. This method has been described as the Center of Effect Generator (CEG) algorithm in [1] and [4].

In western tonal music several pitches are approximated by the same frequency (these pitches are said to be enharmonically equivalent). In a MIDI file, enharmonically equivalent pitches are represented by the same numerical value. In order to use the CEG algorithm, we needed to first convert MIDI pitch numbers to contextually correct pitch names. Real-time pitch spelling algorithms using the Spiral Array and various contextual windows have been proposed by Chew and Chen [2],[3]. The method implemented in SkeFiS is the sliding window algorithm detailed in [2]. This method incrementally generates pitch spellings for note events (note by note) based on tonal contexts derived from a short history window (five beats). We use the history window to generate a c.e. that acts as a proxy for the key. The algorithm maps each numeric pitch number to its plausible pitch names on the Spiral Array, and selects the best match through a nearest-neighbor search.

The input to the CEG key-finding algorithm is the pitch names derived from the above pitch spelling algorithm. At this point, it is possible to calculate the c.e. for the segment of music by mapping the pitch names in all time slices in the segment to their positions in the Spiral Array, and generating a point that is the composite sum of the pitch positions weighted by the number of "hits". Since keys are also defined as points in the same space, it is then straightforward to compute the distance between this c.e. and all key representations to determine which key is closest to the c.e.

3 EVALUATION RESULTS

The evaluation was performed using 1252 MIDI files, and the results reported in [5]. Table 2 records the evaluation results for our system.

TABLE 1 – RESULTS OBTAINED BY SKEFIS

System and algorithms	SKEFIS
Rank	5
Total score	934
Percentage score	74.6%
Correct keys	799
Perfect 5th errors	210
Relative major/minor errors	80
Parallel major/minor errors	30
Other errors	133
Runtime (s)	471
Machine	OS:CentOS; Processor: Dual AMD Opteron 64 1.6Ghz; RAM:4GB

4 SYSTEM COMPARISONS

Five groups participated in the MIREX 2005 symbolic key finding contest. They included submissions from A. Ehmann, A. Mardirossian & E. Chew, D. Rizo & J. Iñesta, D. Temperley, and Y. Zhu. This section is a comparison of the submitted systems. We are only able to discuss three of the systems as the other two, by A. Ehmann and Y. Zhu, were not made available at the time of writing of this abstract. We base our comparisons on the preliminary abstracts submitted by Rizo & Iñesta, and Temperley (the first parts of [6] and [7] respectively).

TABLE 2 – COMPARISONS OF SUBMITTED ALGORITHMS

	Mardirossian & Chew	Rizo	Temperley
Representation	Mapping of keys on spiral	Coding scheme of trees	Segments of pieces
Key Recognition Method	CEG (nearest neighbor) applied to first 28s of music	Post-order traversal of tree to rank all keys	Bayesian probability with dynamic programming
Key Templates	Key representations in the Spiral Array	Rating model	Koska-Payne profile
Scope of data inspected	First 28 seconds	All notes	Segments of of 1.2 seconds.
Query	CE, sum of pitch positions weighted by durations	Tree representation of music	Duration profile for each segment
Key Selection Criteria	Select key of the first 28 seconds	Select key with highest ranking	Perform probabilistic DP, pick label for first 1.2s

Table 1 outlines the details of the algorithms for which an abstract was made available. Each of these algorithms is very different in its approach to key finding. Our system has been described in Sections 1 and 2.

The algorithm by Rizo & Iñesta creates a tree representation of polyphonic music. The search for the main key of the piece is based on a post-order traversal of the tree. Each node incorporates a rating of the likelihood of its pitch collections' membership in a key. Once the children have the ranks for all keys calculated, their results are combined to establish a ranking for each key at the parent level. The root node of the tree contains a list of keys, ordered by rank. The highest ranked key is selected as the answer.

Temperley's algorithm begins by dividing a piece into 1.2-second segments. The model then searches for the optimal "key structure", where a key structure is a labeling of each segment with a key using a probabilistic dynamic programming method that penalizes key changes, and maximizes the probability of a given key given the observed information. The probabilities are computed using key profiles determined from the Koska-Payne corpus. After the optimal labelling of all segments is computed, the main key of the piece is determined by choosing the key of the first segment.

5 DISCUSSION AND CONCLUSION

Although it is possible that the spelling step in SkeFiS may have introduced some error (neither of the other two system descriptions included any spelling, considering only numeric pitch classes), we believe that the disadvantage of our system derives primarily from the naïve use of a cumulative window for key recognition.

Both Temperley's and Rizo & Iñesta's contribution provided essentially a parsing of the entire musical excerpt to derive a reasonable key label for the sample. Rizo & Iñesta built a tree structure for explaining the tonal groupings in a polyphonic sample, while Temperley's built a different kind of tree (in a dynamic programming algorithm) to create the optimal labelling of the passage. We hypothesize that Temperley's winning approach is due in part to his use of Bayesian reasoning, based on pattern matching using key profiles learned from the Kostka-Payne corpus, in the dynamic programming algorithm.

The results of the competition show that the cumulative window approach is far from optimal, even for excerpts as short as 30 seconds, and that there is room for improvement in the application of a key recognition algorithm such as the CEG to the analysis of real data. For example, the dynamic programming approach described by Temperley can be set up so as to optimize the likelihood of keys as measured by distances in the Spiral Array; or, key rankings in each node in Rizo & Iñesta's tree-based approach can be computed using distances returned by the CEG method.

6 ACKNOWLEDGEMENTS

We thank the MIREX team led by Stephen Downie, and in particular, Andreas Ehmann, Emmanuel Vincent, and Kris West for amassing the database and running the key finding evaluations.

This research has been funded in part by a National Science Foundation (NSF) Information Technology Research Grant No. 0219912, and made use of the shared facilities at the Integrated Media Systems Center, an NSF Engineering Research Center Cooperative Agreement No. EEC-9529152. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors, and do not necessarily reflect those of NSF.

Keywords: polyphonic MIDI key finding, evaluation.

REFERENCES

[1] E. Chew. "Towards a Mathematical Model of Tonality", Doctoral dissertation, Department of Operations Research, Massachusetts Institute of Technology, 2000.

[2] E. Chew, and Y.-C. Chen. "Mapping MIDI to the Spiral Array: Disambiguating Pitch Spellings", H.K. Bhargava and Nong Ye (Eds.), Computational Modeling and Problem Solving in the Networked World, Proceedings of the 8th INFORMS Computer Society Conference, 2002.

[3] E. Chew, and Y.-C. Chen. "Real-Time Spelling Using the Spiral Array", Computer Music Journal, 2005.

[4] E. Chew. "Modeling Tonality: Applications to Music Cognition", Proceedings of the 23rd Annual Meeting of the Cognitive Science Society, 2001.

[5] MIREX 2005 - The 1st Music Information Retrieval Evaluation eXchange . Url: *www.music-ir.org/mirexwiki/index.php/Main_Page*. Contest results url: *www.music-ir.org/evaluation/mirex-results/sym-key/index.html*.

[6] Rizo, D., and Iñesta, J., "Tree Symbolic Music Representation for Key Finding," Abstract of the Music Information Retrieval Evaluation Exchange, 2005.

[7] Temperley, D., "A Bayesian Key Finding Model," Abstract of the Music Information Retrieval Evaluation Exchange, 2005.