MIDI TO SP-MIDI TRANSCODING USING PHRASE STEALING

Simon Lui *virtuoso@cs.ust.hk*

Andrew Horner *horner@cs.ust.hk*

Lydia Ayers *layers@cs.ust.hk*

Department of Computer Science, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong

ABSTRACT

The Musical Instrument Digital Interface (MIDI) is a popular music format used in multimedia messaging service, such as mobile phone ringtones. Scalable Polyphony MIDI (SP-MIDI) is an enhanced format that allows composers to specify how MIDI data should be performed by hardware devices with different numbers of polyphonic voices. Most current mobile phones only support SP-MIDI ringtones with specific polyphonic limits. Since most MIDI files are composed without regard to polyphonic limits, a common problem in the mobile phone industry is conversion from MIDI to SP-MIDI. However, simple MIDI to SP-MIDI reduction algorithms, such as note-stealing, may lose or interrupt important musical information. This paper presents a phrase stealing algorithm that drops the perceptually least important notes when reducing a MIDI file to SP-MIDI, and preserves the most important phrases. The phrase stealing algorithm produces SP-MIDI files with an average phrase length of 10 notes, in contrast to the note stealing algorithm which disrupts perceptually important melodic phrases. Formal listening test results show that listeners found the phrase stealing reduction very similar to the original, representing a big improvement over note stealing which listeners only found somewhat similar to the original.

1. INTRODUCTION

Music on mobile phones has become a hot topic in recent years. Sending customized ring tones is especially popular. Unlike most music synthesizers and soundcards, only a few mobile phones support the MIDI standard [1] which represents music in an efficient note-level format. Instead, most mobile phones support SP-MIDI [2], a constrained version of MIDI. This is a severe limitation since of the vast quantity of MIDI files available, relatively few are in SP-MIDI format. It also creates compatibility problems when MIDI phones send ringtones to SP-MIDI phones. Conversion from MIDI to SP-MIDI (called "transcoding") is problematic, and usually requires discarding notes and even whole channels [3]. This paper considers how to make such a transcoding so as to minimize the impact on the music.

SP-MIDI was conceived as a solution for Third Generation (3G) mobile applications and systems. It shares many similarities with MIDI, but channels are prioritized. The mobile phone plays as many SP-MIDI channels as possible without exceeding its number of hardware voices, which is called the MIP (Maximum Instantaneous Polyphony) [2]. Similarly, a song's MIP is simply the maximum number of notes played at any given time. If a song's MIP is larger than the hardware MIP limit, lower priority channels are discarded. The number of hardware voices limits the number of simultaneous voices at any given time, but not the number of channels.

Most MIDI files [4] were composed without concern for channel priority or MIP limits. To make use of these files, ring tone providers can manually write different versions of the same song (a monophonic version, a MIP=4 version, etc). However, this is tedious and time consuming. To automate the process, an effective MIDI to SP-MIDI transcoding algorithm is needed.

Most hardware music synthesizers solve the MIP limit problem using note stealing, an FIFO strategy where the oldest note is turned off when the number of simultaneous notes exceeds the number of hardware voices. Note stealing takes advantage of the perceptual importance of note attacks over sustains and releases [5]. However, it may truncate important notes in the melody, making it hard to recognize the song. Even worse, a song with many simultaneous voices may degrade to "thrashing" on a phone containing only a few hardware voices [6], with notes constantly turning on and off in a flurry of activity, leaving no resemblance to the original piece of music.

This paper introduces a phrase stealing algorithm for MIDI to SP-MIDI transcoding. It automatically reduces the polyphony of a MIDI file to a specified MIP limit. The phrase stealing algorithm takes musical context into account when deleting notes. Section 1 introduces the algorithm from a musical point of view. Section 2 describes how to eliminate trivially less important notes. Section 3 identifies important features in the music. Section 4 explains the core part of the phrase stealing algorithm first from a musical point of view and then from a technical point of view. Section 5 summarizes the algorithm, and Section 6 discusses the results we have tested. Section 7 concludes the paper.

2. PHRASE STEALING ALGORITHM

From a musical perspective, phrases are at a higher level than notes, and a lower level than forms. Music can be compared with a book, where notes correspond to words, phrases correspond to sentences, and forms correspond to paragraphs. If we want to limit the number of words in an article, randomly deleting a word in a sentence, (the literary equivalent to note-stealing) will result in a loss of information. Probably the best way to limit the number of words is to delete less-important complete sentences: this is the principle of phrase stealing. Phrase stealing may be viewed as a continuity-tracking algorithm, a special case of auditory stream segregation or auditory scene analysis [7].

Though we call our algorithm “phrase stealing”, it is really a “phrase preserving” algorithm because it identifies and keeps the most important phrases and deletes less important ones. In particular, phrase stealing follows the following principles:

1. Phrase stealing maintains smooth bass lines. The bass line determines the color and flow of musical progressions [8].
2. Phrase stealing maintains smooth phrases. A phrase is a continuous line of notes. It can be a melody, counter melody, or an important accompanying line.

3. PREPROCESSING

There are a couple types of less-important notes that are easy to identify and omit, even without phrase stealing.

First, certainly, unison notes belonging to the same channel (the same instrument) can be dropped without affecting recognition of the piece [9]. Second, combining the percussion channels into a single channel simplifies processing. Once combined, they can be prioritized as shown in Table 1. Since most percussion sounds are short and impulsive, when we reduce the percussion channel to MIP=1, only relatively small perceptual changes were heard in the examples we tested.

Priority	Instrument
1st	Cymbal on the first beat
2nd	Rolling Tom
3rd	Bass Drum
4th	Snare Drum
5th	Tambourine/Ride on weak beat
6th	Hi-hat
7th	Tambourine/Ride on strong beat
8th	Tom
9th	Others
10th	Cymbal not on first 1st beat / windchime

Table 1. Priority list for percussion instruments.

4. FEATURE EXTRACTION

We can use various musical features to help us decide which phrases to preserve and which to delete.

Finding the key signature can help identify other details such as chords and cadences. For most common-practice music, the notes used most frequently outline a diatonic scale. We can calculate the statistics of all 12 chromatic pitches, and rank them. The pitch with the highest rank gets 12 points, and the pitch with the lowest rank gets 1 point. After ranking, the pitches are fit to 12 different scales with each pitch acting as tonic. The score for each candidate tonic is the sum of points for its 7 diatonic notes, the tonic with the highest score is selected as the tonic of the piece.

Chord types are calculated for the purpose of identifying cadences and dividing phrases. Notes in a chord are mapped into a circular chroma-space, where the best match to triads can then be found. Extra notes are ignored, since it is enough for cadence recognition, and we only have voice leading tables for triads. The chord can be classified relative to the tonic.

Also, we use Top Notes and Bass Notes to help identify the bass lines. In our definition, a Top Note is not necessary a melody note, but it will never be a Bass Note. The determination of Top and Bass Notes determined is as follows. First, for each chord, notes which satisfy all the following are regarded as Top Notes:

1. It is the highest pitch in a chord, and
2. Its pitch is above the average pitch of the piece minus 10% of the standard deviation, and
3. It is closer to the Top Note than the Bass Note in the last chord (if the last chord has both Top and Bass Notes).

Next, for each chord, notes which satisfy all the following are regarded as Bass Notes:

1. It is the lowest pitch in a chord, and
2. Its pitch is below the average pitch of the piece plus 10% of the standard deviation, and
3. It is not a Top Note.

Finally, the Bass Notes form the bass line of the piece, which we will use in our subsequent analysis.

5. PHRASE STEALING ALGORITHM

The following steps are used in the phrase stealing algorithm, and are discussed below:

1. Phrase identification
2. Parallel phrase reduction
3. Phrase stealing

5.1. Phrase Identification

Before we decide which phrases to drop, we must segment and identify the phrases. Phrases are constructed by grouping notes that preserve the best voice leading, and we should only group notes from the same channel. Each group of notes forms a phrase.

Notes are grouped in the following manner: for each consecutive chord A and B, let A be the chord with the fewest notes. Group each note of A to a note in B. Each note can be grouped to only one other note.

MIDI File (Genre)	Max MIP	Phrases formed	Average Phrase length (in notes)	% of notes dropped *(PS)*	% of notes dropped / truncated *(NS)*	Average rate (musically trained listeners)		Average rate (musically untrained listeners)		Average rate (overall)	
						PS	NS	PS	NS	PS	NS
1. Pop	14	286	6	36%	35%	7.50	6.00	8.25	6.75	7.88	6.38
2. Pop	15	53	13	39%	39%	9.75	7.00	10.00	8.00	9.88	7.50
3. Baroque	16	487	30	60%	58%	8.00	5.25	8.00	4.75	8.00	5.00
4. Pop	26	92	4	63%	61%	4.50	3.00	5.25	3.25	4.88	3.13
5. Classical	15	1317	10	28%	28%	5.50	4.00	6.50	3.50	6.00	3.75
6. Jazz	34	1083	4	50%	46%	5.50	4.25	6.00	4.75	5.75	4.50
7. Jazz	48	898	3	64%	60%	6.00	4.75	5.75	4.50	5.88	4.63
8. Jazz	18	455	8	43%	40%	5.75	4.00	5.50	4.75	5.63	4.38
9. Jazz	10	68	14	10%	10%	4.50	4.75	5.50	4.50	5.00	4.63
Average:		527	10	43.67%	41.89%	6.33	4.78	6.75	4.97	6.54	4.88

Table 2. Phrase stealing and note stealing results and Listening test results (PS: Phrase Stealing, NS: Note Stealing)

For common chords such as I, V and vii, use voice leading tables to resolve tendency tones (tones which have a strong pull toward another), where voice leading tables describe the grouping priority. For the other chords, group each note of the preceeding chord with its nearest neighbour in the succeeding chord. Phrases are closed after cadences. Only simple cadences such as V-I or vii-I are identified.

5.2. Parallel Phrase Reduction

A second phrase may largely parallel the melody [10]. Such a phrase is a good candidate for elimination. Accompanying chords often follow the same rhythm, and we can use this cue to reduce such parallel phrases. Only the top phrase is especially important, and others can be eliminated. Parallel phrase reduction is thus done as a pre-processing step before phrase stealing in order to reduce the number of rhythmically similar phrases.

5.3. Phrase Stealing

We delete phrases in an LIFO manner. Each phrase is deleted or kept as a whole. We found it useful to handle different cases depending on the MIP value. For MIP=1 SP-MIDI, we preserve the melody line at a higher priority than the bass line. For MIP=2 SP-MIDI, we preserve the melody line and percussion line at a higher priority than the bass line. However, if the percussion channel is empty, we preserve the bass line. For SP-MIDI with MIP=3 or more, we can afford to preserve a smooth baseline, so all the bass notes together are combined into a preserved phrase.

6. RESULTS

6.1. Statistics

We conducted statistical tests to confirm the perceptual effectiveness of the phrase stealing algorithm. Our analysis focuses on MIP=4 reduction, since it is common in current mobile phones. Also traditional note stealing usually yields poor results for MIP=4 reduction.

Table 2 shows that many examples had average phrase lengths of 8 or more notes. The overall average phrase length was 10, which indicates that significant phrase continuities have been identified and preserved. A large percentage of notes were dropped, and the percentage is nearly identical for both phrase stealing (43.67%) and note stealing (41.89%). Note that for phrase stealing, each phrase and hence each note is deleted or kept as a whole, while for note stealing, it will either drop a whole note or just truncate a note' s tail and preserve the head. However, informal listening tests found that the phrase stealing reduced files still sound reasonably similar to the originals, and much better than the note stealing reduced files.

In the test, 4 musically trained listeners (with at least 4 years of training) and 4 musically untrained listeners were used. Listeners heard the original, then the reduced file, and finally the original again. They were then asked to rate the reduced file from 0 -10, where 10 means entirely identical and 0 means entirely unrelated. It was found that phrase stealing has an absolute advantage over note stealing. In particular, there is an obvious improvement for Baroque and pop pieces. Such pieces usually have contrapuntal melody lines which phrase stealing well preserves. Since jazz pieces may have a lot of melodic leaps, phrase stealing is not always able to perfectly identify phrases, but it is still better than note stealing.

Musically untrained listeners are usually only sensitive to abstract materials (like melody) while trained listeners may focus on other musical aspects. This explains why the results for musically untrained listeners are surprisingly slightly higher (see Table 2). However, both groups rated phrase stealing higher than note stealing. This indicates that besides the obvious musical materials, phrase stealing is better able to preserve important musical structures (e.g., the bass line).

6.2. Examples

As an example, Figure 1 shows an excerpt from a Baroque piece. The phrase stealing algorithm helps follow the switching between different instruments. Figure 2 shows the reduced output. Figure 3 shows a pop piece, and Figure 4 shows the reduction. Although the number of percussion notes is reduced by 50%, the effect is still perceptually similar to the original, since mostly only the hi-hat has been eliminated. The chord accompaniment has also been reduced. However, since it retains its rhythmic structure, there is no significant perceptual imp act.

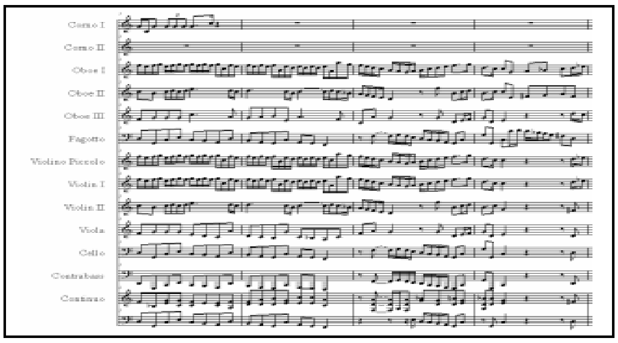

Figure 1. Baroque excerpt (MIDI File 3) – the original.

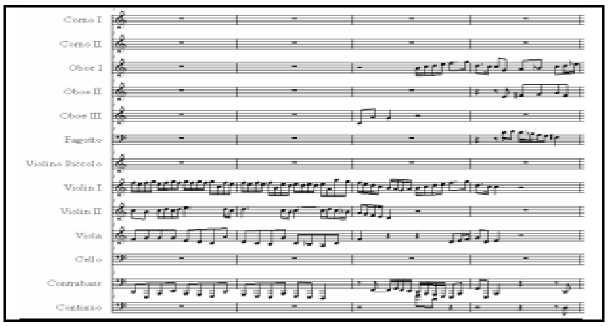

Figure 2. Baroque excerpt (MIDI File 3) – MIP=4 reduction. A smooth melody / counter melody is retained in the score.

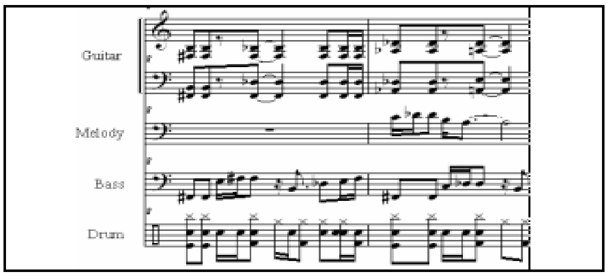

Figure 3. Pop excerpt (MIDI File 1) – the original.

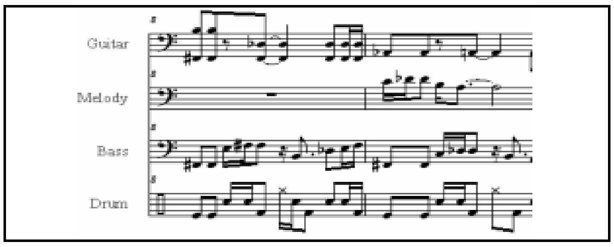

Figure 4. Pop excerpt (MIDI File 1) – MIP=4 reduction. The 4th drumset staff has been reduced. The 1st staff with guitar chords has been reduced from MIP=3 to MIP=1 or 2.

6.3. Discussion

The phrase stealing algorithm keeps events that a listener expects to hear [11]. It usually preserves the melody and phrase lines, that is, those elements that listeners are the most likely to be listening to closely and following. It preserves smooth bass lines when there are enough hardware voices to do this. A smooth bass often establishes the underlying mood of the piece.

7. CONCLUSION

A new phrase stealing algorithm for MIDI to SP-MIDI reduction has been introduced. It uses feature extraction to preserve smooth phrase lines, which are perceptually important. The results show that melody lines can usually be preserved, except when they are very discontinuous. Listening test results show the reduced files are perceptually similar (sometimes very similar) to the originals. Phrase stealing helps mobile phones automatically convert MIDI files to SP-MIDI with the least perceptual impact upon the music.

8. ACKNOWLEDGEMENT

This work was supported in part by the Hong Kong Research Grant Council' s Project HKUST6167/03E and the Innovation and Technology Fund Project ITS/122/03. We would like to thank Eno Tsin and Chris Wong of Sensestream Ltd. for their discussion and valuable comments, and all the participants of the listening test for their time and effort.

9. REFERENCES

[1] The MIDI Manufacturers Association, MIDI 1.0 Detailed Specification. Version 4.2, 1996.

[2] The MIDI Manufacturers Association, Scalable Polyphony MIDI Specification. Version 1.0, 2002.

[3] Research Section of the Nokia Web Site, http://www.nokia.com/nokia/0,8764,5343,00.html

[4] The MIDI Manufacturers Association, Standard MIDI files 1.0, RP-001, 1996.

[5] Drake, C. and Palmer, C. Accent structures in music performance, Music Perception, 1993.

[6] Maher, R. C. Wavetable synthesis Strategies for Mobile Devices, JAES, volume 53, No.3, 2005.

[7] Bregman, A.S. Auditory Scene Analysis, MIT Press, 1990.

[8] Temperley, D. The Cognition of Basic Musical Structures Cambridge, MA: MIT Press, 2001.

[9] Davies, J.B. The Psychology of Music, Hutchinson & Co., London, 1978.

[10] Cambouropoulos, E. Musical parallelism and melodic segmentation, Proceedings of the XII Colloquium of Musical Informatics, Italy, 1998.

[11] Cambouropoulos, E. Melodic Cue Abstraction, Similarity, and Category Formation: A Formal Model. Music Perception, 18(3), 347ff, 2001.