IMPROVING VOICE SEPARATION BY BETTER CONNECTING CONTIGS

Nicolas Guiomard-Kagan[1] **Mathieu Giraud**[2] **Richard Groult**[1] **Florence Levé**[1,2]
[1] MIS, Univ. Picardie Jules Verne, Amiens, France [2] CRIStAL, UMR CNRS 9189, Univ. Lille, Lille, France
{nicolas,mathieu,richard,florence}@algomus.fr

ABSTRACT

Separating a polyphonic symbolic score into monophonic voices or streams helps to understand the music and may simplify further pattern matching. One of the best ways to compute this separation, as proposed by Chew and Wu in 2005 [2], is to first identify *contigs* that are portions of the music score with a constant number of voices, then to progressively *connect* these contigs. This raises two questions: Which contigs should be connected first? And, how should these two contigs be connected? Here we propose to answer simultaneously these two questions by considering a set of musical features that measures the quality of any connection. The coefficients weighting the features are optimized through a genetic algorithm. We benchmark the resulting connection policy on corpora containing fugues of the *Well-Tempered Clavier* by J. S. Bach as well as on string quartets, and we compare it against previously proposed policies [2, 9]. The contig connection is improved, particularly when one takes into account the whole content of voice fragments to assess the quality of their possible connection.

1. INTRODUCTION

Polyphony, as opposed to monophony, is music created by simultaneous notes coming from several instruments or even from a single polyphonic instrument, such as the piano or the guitar. Polyphony usually implies chords and harmony, and sometimes counterpoint when the melody lines are independent.

Voice separating algorithms group notes from a polyphony into individual voices [2,4,9,11,13,15]. These algorithms are often based on perceptive rules, as studied by Huron [7] or Deutsch [5, chapter 2], and at the first place *pitch proximity* – voices tend to have small intervals.

Separating polyphony into voices is not always possible or meaningful: many textures for polyphonic instruments include chords with a variable number of notes. Conversely, one can play several streams on a monophonic instrument. *Stream* separation algorithms focus thus on a narrower scale, extracting groups of coherent notes. These segments are not necessarily connected throughout the whole score: a voice can be split into several streams and a stream can cluster notes from different voices [14, 16].

Both voice and stream segmentation algorithms provide a better understanding of polyphony and make inference and matching for relevant patterns easier. We previously showed that voice and stream separation algorithms are two facets of the same problem that can be compared with similar evaluation metrics [6]. Pertinent evaluation metrics measure how segments or voices of the ground truth are grouped together in the algorithms predictions, as the transition-based evaluation [2] or the measure of mutual information [6, 12].

Based on these metrics, it appears that the contig approach, as initially proposed by Chew and Wu [2] (Section 2), is one of the best approaches to separate voices, starting from *contigs* having a constant number of voices. The results depends on how the contigs are *connected*, larger voice or stream segments being built starting from smaller ones.

In this article we propose and compare several criteria to ground the *connection policy*, that is both the choice of the order of the contigs to be connected, and the connection itself between contigs. In addition to the criteria used in the literacy, we introduce new criteria that take into account *more musical context*, averaging pitches and durations over voice fragments (Section 3). We weight these criteria using a genetic algorithm (Section 4). We show how some values of these criteria can partially simulate the previous methods, and evaluate the results on sets of fugues and string quartets. By improving this contig connection, we improve the precision of voice separation algorithms (Section 5). We further study the distribution of failures, showing that a higher precision can be obtained by stopping the contig connection before the connection quality drops.

2. VOICE SEPARATION BASED ON CONTIGS

The contig approach, proposed by Chew and Wu (denoted by CW in the following) first separates the music score into contigs that have a constant number of notes played at the same time then progressively connect these contigs to the whole score [2].

The first step splits the input polyphonic data into blocks called *contigs* such that the number of simultaneous notes in a contig does not change (Figure 1). Notes cross-

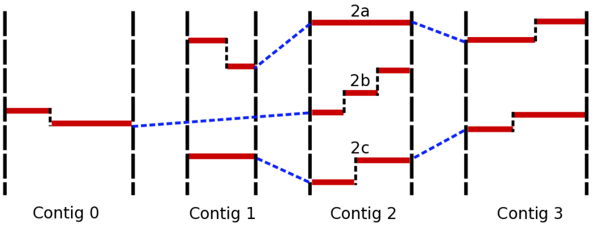

Figure 1. In this piano-roll symbolic representation, each segment describes a note. The horizontal axis represents time and the vertical axis represents pitches. The notes can be grouped in four *contigs*, each of them containing a constant number of notes played at the same time. Contig 2 contains three *voice fragments* $2a$, $2b$ and $2c$. The challenge of contig-based voice separation algorithms is to connect these voice fragments across contigs to build coherent voices throughout the score. The non-vertical dotted lines show a possible solution of the voice separation.

ing the border of several contigs are split in several notes. The idea behing building contigs is that the voice separation is relatively easy inside them: Notes in each contig are grouped by pitch height to form *voice fragments*.

The second step links together fragments from distinct contigs, following some musical principles (Figure 2). The algorithm has now to take two kinds of decisions, following what we call a *connection policy*:

- *which* contigs should be connected first?
- *how* should these two contigs be connected?

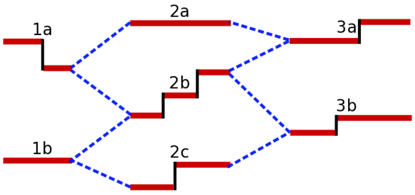

Figure 2. Any connection policy should decide which contigs should be connected (such as, for example, 1 and 2) and how to do this connection. There are here three possible connections (without voice crossing) between the contigs 1 and 2: $C_1 = \{(1a, 2a), (1b, 2b)\}$, $C_2 = \{(1a, 2a), (1b, 2c)\}$, and $C_3 = \{(1a, 2b), (1b, 2c)\}$.

Order of connection of contigs. In CW algorithm, the connection starts from the *maximal contigs* (i.e. contigs containing the maximal number of voices). Since the voices tend not to cross, the voice separation and connection in these contigs with many voices were thought to be more reliable. Then, CW continues the connection process to the left and to the right of these maximal contigs. In Figure 1, the CW policy will thus connect contigs $1, 2, 3$, then finally $0, 1, 2, 3$.

Ishigaki, Matsubara and Saito (denoted IMS in the following) suggested another connection policy, starting with minimal contigs and connecting contigs with an *increasing* number of fragments (i.e. the number of fragments in the left contig is lower or equal to the number of fragments in the right contig) [9]. The idea is that the (local) start of a new voice is a more perceptible event than the (local) end of a voice. Once all those possible connections are done, maximal contigs are considered as in CW algorithm to terminate the process. In Figure 1, IMS policy will connect contigs $0, 1$, then $0, 1, 2$, and finally $0, 1, 2, 3$.

Fragment connection. The policy to connect fragments of the original CW algorithm, reused by IMS, is based on two principles: Intervals are minimized between successive notes in the same stream or voice (pitch proximity); Voices tend not to cross. Formally, the connection between two contigs is a set of (ℓ, r) fragments that maximize a *connection score*. This score is here based on the absolute difference between the pitch of the last note of the left fragment ℓ and the pitch of the first note of the right fragment r. There is moreover a very large score for the connection of notes split between two contigs to keep them in the same final voice.

3. MORE MUSICAL FEATURES TO IMPROVE THE CONNECTION POLICY

3.1 A new view on the contig-based approach

We argue that the two questions of the connection policy (*which* contigs should be connected? *how* to connect them?) should be handled at a same time: to build coherent voices across a piece, one should always connect the contigs yielding the "safest" connections between voice fragments. The quality of these connections should be properly evaluated with musical features that will be introduced below.

Given two successive contigs i and $i + 1$, and one way C to connect them (set of pairs of fragments), we define a *connection score* $S(i, C)$, computed as a weighted sum of musical features, that measures the quality of this connection: The higher the connection score, the safer the connection. The connection scores will extend the ones used by CW and IMS, that did not systematically explore the relation between the two decisions of the connection policy.

At each step of the algorithm, the (i, C) maximizing S is selected, giving both the "best contigs" to connect and the "best way" to connect them. Once this connection is made, the connections scores between the newly formed contig and its left and right neighbors have to be computed.

Definitions. Let n be the maximal number of simultaneous notes in the piece. Let n_i (respectively n_{i+1}) be the maximal number of voices of the contig i $(i + 1)$. After some connections have been made, a contig may have a different number of simultaneous notes at its both extremities, but the hanging voices are "projected" to these extremities.

For two successive contigs i and $i + 1$, let C be a set of pairs (ℓ, r), where ℓ is a fragment of the (left) contig i and r a fragment of the (right) contig $i + 1$, each fragment appearing at most once in C (Figure 2). C has thus at most $m = \min(n_i, n_{i+1})$ elements, and, in the following, we only consider sets with m elements, that is with the highest possible number of connections. Denoting $M = \max(n_i, n_{i+1})$, there are $M!/(M-m)!$ different such combinations for C, and only $\binom{M}{m}$ if one restricts to the combinations without voice crossing.

We consider that we have N features $f_1(i, C), f_2(i, C) \dots f_N(i, C)$ characterizing some musical properties of the connection C between contigs i and $i+1$. Each feature $f_k(i, C)$ has a value between 0 and 1. Finally let $\alpha_1, \alpha_2, \dots, \alpha_N$ be N coefficients such that $\sum_{k=1}^{N} \alpha_k = 1$. We then define the connection score as a linear combination of the features $S(i, C) = \sum_{k=1}^{N} \alpha_k f_k(i, C)$.

In the two following paragraphs, we propose different features $f_k(i, C)$ depending on the musical properties of contigs and fragments. The values of the coefficients α_k will be discussed in Section 4.

3.2 Features on the contigs

First we consider features that are not related to the connection C but depend only on the contigs, more precisely on the maximum number of voices in each contig.

- *maximal_voices(i)* $= \max(n_i, n_{i+1})/n$. The closer the number of voices to the maximal number of voices, the higher the connection score .
- *minimal_voices(i)* $= (n+1-\min(n_i, n_{i+1}))/n$. The closer the number of voices to 1, the higher the connection score.

One can in particular favor some contig connection based on the comparison of the number of voices between the left and the right contigs:

- *difference_nb_voices(i)* $= 1 - (|n_i - n_{i+1}|/(n-1))$. The closer the number of voices of the left and the right contigs, the higher the connection score.

Or with the following binary features, that will equal 0 if the condition is not met:

- *increase(i)* $= 1$ iff $n_i < n_{i+1}$;
- *increase_one(i)* $= 1$ iff $n_i + 1 = n_{i+1}$;
- *increase_equal(i)* $= 1$ iff $n_i \leq n_{i+1}$;
- *decrease(i)* $= 1$ iff $n_i > n_{i+1}$;
- *decrease_one(i)* $= 1$ iff $n_i - 1 = n_{i+1}$;
- *decrease_equal(i)* $= 1$ iff $n_i \geq n_{i+1}$.

Those features are inspired by the connection policy of the existing algorithms. The *maximal_voices(i)* feature reflects the idea used by the CW algorithm: It is safer to first connect contigs having a large number of voices. The reverse idea, as measured by *minimal_voices(i)*, was proposed together with the *increase(i)* idea by the IMS algorithm, favoring the connection of contigs with an increasing number of voices. The idea is that the (local) start of a new voice is a more perceptible event than its (local) end. This is even more remarkable in contrapuntal music such as fugues where enterings of voice on thematic patterns (subjects, counter-subjects) are often clearly heard.

We propose to further use the *increase_one(i)* feature that should better assert an entry of exactly *one* new voice. Conversely, we also evaluate the opposite idea (*decrease(i)*, *decrease_one(i)*, *decrease_equal(i)*).

Finally the connection could favor successive contigs sharing a same note:

- *maximal_sim_notes(i)* $= n^{=}/\min(n_i, n_{i+1})$, where $n^{=}$ is the number of notes with the same pitch and same onset (i.e. note split in two) at the extremities of contigs i and $i+1$. The more the contigs share common notes, the higher the connection score is.

This feature derives from the original implementation of CW, where connectig contigs with shared notes was awarded a very large score.

3.3 Features on the fragments

Now we consider features based on the individual fragment connections (ℓ, r) composing C.

Pitches. How can we measure the quality of connecting a fragment ℓ to a fragment r? The main criterion of the CW and IMS algorithms was to follow the pitch proximity principle, favoring connections of fragments having a small pitch interval. Given C and $(\ell, r) \in C$, let *last_pitch*(ℓ) and *first_pitch*(r) be the pitches of the extreme note of the left fragment ℓ and the right fragment r:

- *extreme_pitch(C)* $= 1 - \sum_{(\ell,r) \in C} |$*last_pitch*$(\ell) -$ *first_pitch*$(r)|/\nu$. The closer the pitches between the connected notes, the higher the connection score.

The normalization factor $\nu = 60 \cdot |C|$ semitones was chosen in order to range the feature value between 0 (5 octaves between connected pitches) and 1 (equal pitches). However, this *extreme_pitch(C)* score only considers one note on each side. We propose to extend this feature by evaluating the *pitch range coherence*, taking into account the average pitch (*average_pitch*) of *all notes* of one or both fragments. Indeed, voices tend to have the same pitch range throughout the piece, and moreover through the fragments:

- *avg_pitch_right(C)* $= 1 - \sum_{(\ell,r) \in C} |$*last_pitch*$(\ell) -$ *average_pitch*$(r)|/\nu$;
- *avg_pitch_left(C)* $= 1 - \sum_{(\ell,r) \in C} |$*average_pitch*$(\ell) -$ *last_pitch*$(r)|/\nu$;
- *avg_pitch(C)* $= 1 - \sum_{(\ell,r) \in C} |$*average_pitch*$(\ell) -$ *average_pitch*$(r)|/\nu$.

Some voice separation algorithms assign each note to the voice with the closest average pitch [10]. These algorithms are quite efficient, and the *avg_pitch(C)* feature reproduces this idea at a local scale: Given a fragment with a few notes, even if one may not know to which (global) voice it belongs, one already knows a local pitch range.

Durations. Similarly, we can measure the difference of durations to favor connection of contiguous fragments with a same rhythm. Indeed, the musical textures of each voice tend to have coherent rhythms. For instance, a voice in whole notes and another one in eights will often be heard as two separate voices, even if they use very close pitches. Given C and $(\ell, r) \in C$, let *last_dur*(ℓ) and *first_dur*(r) be the durations, taken in a log scale, of the extreme notes of the left fragment ℓ and the right fragment r:

- *extreme_dur(C)* $= 1 - (\sum_{(\ell,r)\in C} |$*last_dur*$(\ell) -$ *first_dur*$(r)|/\lambda)$. The closer the durations between the connected notes, the higher the connection score.

The normalization factor $\lambda = 6 \cdot |C|$ accounts for the maximal difference (in a log scale) between whole notes (6) and 64th notes, the shortest notes in our corpora (0). Once more, this feature can also be extended to take into account the average log duration (*average_dur*) of one or both fragments instead of the duration of the extreme note:

- *avg_dur_right(C)* $= 1 - \sum_{(\ell,r)\in C} |$*last_dur*$(\ell) -$ *average_dur*$(r)|/\lambda$;
- *avg_dur_left(C)* $= 1 - \sum_{(\ell,r)\in C} |$*average_dur*$(\ell) -$ *last_dur*$(r)|/\lambda$;
- *avg_dur(C)* $= 1 - \sum_{(\ell,r)\in C} |$*average_dur*$(\ell) -$ *average_dur*$(r)|/\lambda$.

These features measure how a fragment may be "mostly in eights" or "mostly in long notes", even if it contains other durations as for ending notes. They handle also rhythmic patterns: a fragment repeating the pattern "one quarter, two eights" has an *average_dur* of about $3 + 1/3$.

Voice crossings. Finally, two features control the voice crossing. On one hand, voice crossings do exist, on the other hand, they are hard to predict. Voice separation algorithms (such as CW and IMS) usually prevent them.

- *crossed_voices(C)* $= 1$ if C contains a crossing voice, and 0 otherwise;
- *no_crossed_voices(C)* $= 1$ if C does not contain a crossing voice, and 0 otherwise.

4. LEARNING COEFFICIENTS THROUGH A GENETIC ALGORITHM

The selection of features coefficients $\alpha = (\alpha_1, \alpha_2, \ldots \alpha_N)$ was achieved with a genetic algorithm with mutation and crossover operators [1]. For computation efficiency, a generation is a set of 60 solutions, each solution being a set of coefficients totaling 1. The first generation G_0 is a set of solutions drawn with random values. The following generations are built through mutations and crossovers.

Mutation. Given a generation G_t, each solution is mutated 4 times, giving 4×60 mutated solutions. Each mutation consists in randomly transferring a part of the value of a randomly chosen coefficient into another one. A new set of 40 solutions is selected from both the original solutions and the mutated solutions, by taking the 30 best solutions and 10 random other solutions.

Crossover. The solutions in this set are then used to generate 20 children solutions by taking random couples of parents. Each parent is taken only once, and a child solution is the average of the coefficients of the parent solutions. The new generation G_{t+1} is formed by the 40 parents and the 20 children solutions.

5. RESULTS

We trained the coefficients weighting the features with the genetic algorithm on the 24 fugues in the first book of the *Well-Tempered Clavier* by J. S. Bach (corpus "wtc-i"). This gives the set of coefficients GA1 after 36 generations (the process stabilized after that). We then evaluated these GA1 coefficients and other connection policies on the 24 fugues of the second book of the *Well-Tempered Clavier* (corpus "wtc-ii") and on 17 first movements of classical and romantic string quartets (Haydn op. 33-1 to 33-6, op. 54-3, op. 64-4, Mozart K80, K155, K156, K157 and K387, Beethoven op. 18-2, Brahms op. 51-1 and Schubert op. 125-1). Our implementation is based on the Python framework music21 [3], and we worked on `.krn` files downloaded from `kern.ccarh.org` [8]. The explicit voice separation coming from the *spines* of these files forms the ground truth on which the algorithms are trained and evaluated.

5.1 Learned coefficients

The column GA1 of Table 1 shows the learned coefficients of the best solution. The high *no_crossed_voices(C)* coefficient confirms that trying to predict crossing voices currently gives many false connections. It may suggest that such detection should be avoided until specific algorithms could handle these cases. We draw two other observations:

- The pitch is the most important feature (the four *pitch* coefficients totaling 0.271). However, *avg_pitch_right(C)* is higher than *extreme_pitch(C)* – and summing *avg_pitch_left(C)*, *avg_pitch_right(C)* and *avg_pitch(C)* gives 0.181, twice *extreme_pitch(C)*. This confirms that using the pitch range coherence is more reliable than using the pitch proximity alone;
- The durations are also important features, especially when one takes the average durations (*avg_dur(C)* or *avg_dur_right(C)*, totaling 0.121). Note that the *extreme_dur(C)* coefficient is very low, confirming the idea that even if the individual durations change, rhythmic textures or small-scale patterns are conserved inside voice fragments.

Finally, the *increase_equal(i)* feature as suggested by IMS is high, but, surprisingly, the *decrease_equal(i)* feature is also high. These two features combined seem to underline that the contig connection is safer when both fragments have the same number of notes. Further experiments should be made to explore these features.

5.2 Quality of the connection policy

Evaluation metrics. The *transition recall (TR-rec)* (or *completeness*) is the ratio of correctly assigned transitions (pair of notes in the same voice) over the number of transitions in the ground truth. The *transition precision (TR-prec)* (or *soundness*) is the ratio of correctly assigned transitions over the number of transitions in the predicted voices [2,6,11]. The TR-rec and TR-prec metrics are equal for voice separation algorithms connecting voices throughout all the piece. Stream segmentation algorithms usually lead to higher TR-prec values as they predict fewer transitions. The ground truth and the output of the algorithms can also be considered as an assignation of a label to every note, enabling to compute the S_o *and* S_u *metrics* based on normalized entropies $H(\text{output}|\text{truth})$ and $H(\text{truth}|\text{output})$. These scores report how an algorithm may over-segment (S_o) or under-segment (S_u) a piece [6, 12]. They measure whether the clusters are coherent, even when streams cluster simultaneous notes. Moreover, we point out the *contig connection correctness (CC)*, that is the ratio of correct connections over all connections done.

Results. Table 2 details the evaluation metrics on the training set and the evaluation sets, both for the GA1 coefficients and for coefficients SimCW and SimIMS simulating the CW and IMS policies, displayed on Table 1. The metrics reported here may be slightly different from the results reported in the original CW and IMS implementations [2, 9]. The goal of our evaluation is to evaluate connection policies inside a same implementation. On all corpora, the GA1 coefficients obtain better TR-prec/TR-rec/CC results than the SimCW and SimIMS coefficients. The GA1 coefficients indeed make better connections (more than 87% of correct connections on the test corpus "wtc-ii"). The main source of improvement comes from the new features that consider the average pitches and/or lengths, as showed by the example on Figure 3.

5.3 Lowering the failures by stopping the connections

The first step of CW, the creation of contigs, is very reliable: TR-prec is more than 99% on both fugues corpora (lines "no connection" in Table 2). Most errors come from the connection steps. We studied the distribution of these errors. With the SimIMS coefficients, and even more with the GA1 coefficients, the first connections are generally reliable, more errors being done in the last connections (Figure 4). This confirms that considering more musical features improves the connections.

By stopping the algorithm with the GA1 coefficients when 75% of the connections have been done, almost half of the bad connections are avoided, giving streams with a good compromise between precision and consistency (lines "GA1-75%" in Table 2).

Feature	GA1	SimCW	SimIMS
increase(i)	0.004	0	0
increase_one(i)	0.004	0	0
increase_equal(i)	**0.137**	0	0.250
decrease(i)	0.013	0	0
decrease_one(i)	0.019	0	0
decrease_equal(i)	**0.112**	0	0
difference_nb_voices(i)	0.009	0	0
maximal_voices(i)	0.026	0.500	0
minimal_voices(i)	0.007	0	0.250
maximal_sim_notes(i)	0.007	0	0
crossed_voices(C)	0.009	0	0
no_crossed_voices(C)	**0.248**	0.250	0.250
extreme_pitch(C)	**0.090**	0.250	0.250
avg_pitch_right(C)	**0.117**	0	0
avg_pitch_left(C)	0.023	0	0
avg_pitch(C)	0.041	0	0
extreme_dur(C)	0.007	0	0
avg_dur_right(C)	0.048	0	0
avg_dur_left(C)	0.006	0	0
avg_dur(C)	**0.073**	0	0

Table 1. Coefficients weighting the musical features used to measure the connection quality, with best coefficients learned on the wtc-i corpus (GA1) and coefficients simulating the connection policy of CW and IMS.

5.4 Other sets of coefficients

To assess reproducibility, we ran the experiment two other times. The learned coefficients GA1′ and GA1″ are very close to GA1 (data not shown) and give comparable results on the learning corpus "wtc-i" (TR-prec = 97.83% and 97.81%, instead of 97.84%). We also optimized coefficients to find a worst solution (data not shown). The coefficients values *crossed_voices(C)* and *minimal_voices(i)* stand out. This confirms that predicting crossing voices is difficult and than small contigs are difficult to connect.

6. CONCLUSION

Voice and stream separation are improved when one optimizes at the same time *when* and *how* the voice fragments should be connected. We explored several features to evaluate the quality of these connections on fugues and string quartets. Taking into account the average pitches and durations of fragments leads to better connections. The resulting algorithm connects voice fragments more reliably than with the previous contig policies, and especially computes high-quality connections at the first steps. This work could be extended by considering more corpora and by evaluating further melodic or structural analysis on the resulting voices or streams. The proposed principles apply to contig-based algorithms but may also be used by other methods clustering notes into voices or streams.

Corpus	Connection policy	CC	TR-rec	TR-prec	S_o	S_u
wtc-i (training set)	no connection	–	86.78%	**99.32%**	**0.98**	0.34
	GA1-75%	**92.61%**	93.45%	98.54%	0.91	0.42
	GA1	**89.30%**	**97.84%**		**0.72**	**0.72**
	worst	16.93%	85.25%		0.06	0.09
	SimCW	81.26%	96.58%		0.65	0.64
	SimIMS	80.62%	96.55%		0.68	0.69
wtc-ii	no connection	–	86.66%	**99.29%**	**0.98**	0.35
	GA1-75%	**92.54%**	92.53%	98.36%	0.91	0.40
	GA1	**87.50%**	**97.14%**		**0.71**	**0.71**
	worst	25.06%	84.22%		0.05	0.07
	SimCW	83.27%	96.22%		0.69	0.68
	SimIMS	81.61%	96.07%		0.69	0.68
string quartets	no connection	–	82.61%	**97.00%**	**0.94**	0.29
	GA1-75%	**85.30%**	87.06%	94.80%	0.83	0.32
	GA1	**78.44%**	**92.59%**		0.44	0.44
	worst	31.88%	80.59%		0.12	0.13
	SimCW	75.99%	92.29%		0.39	0.38
	SimIMS	74.53%	91.79%		**0.62**	**0.61**

Table 2. Evaluation of the quality of various connection policies. Note that the two first policies (No connection, GA1-75%) do not try to connect the whole voices: they have very high TR-prec/S_o metrics, but poorer TR-rec/S_u metrics.

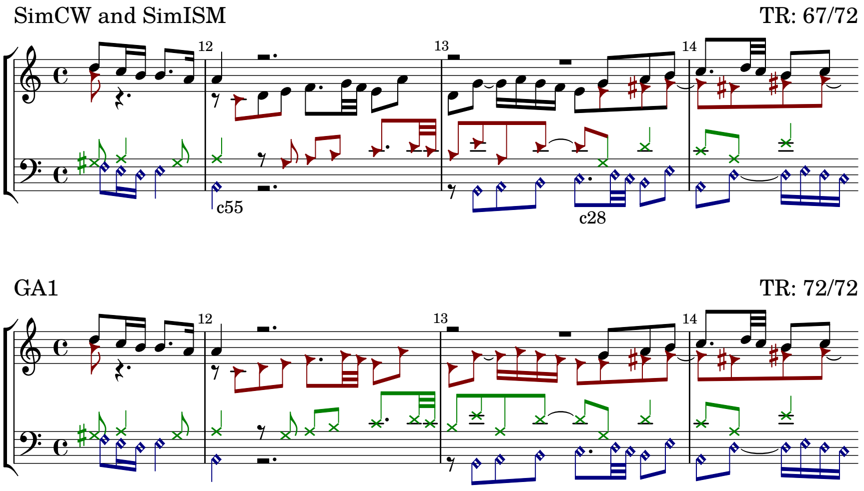

Figure 3. Extract of the fugue in C major BWV 846 by J.-S. Bach. (Top.) The connection policy of previous algorithms fails on connection c28 because of the fifth leap between the D and the G in the tenor voice. This error leads to the wrong connection c55 at a later stage of the algorithm. (Bottom.) Because the coefficients GA1 take into account the feature *avg_pitch*(C) and the related features, the connection is correct here.

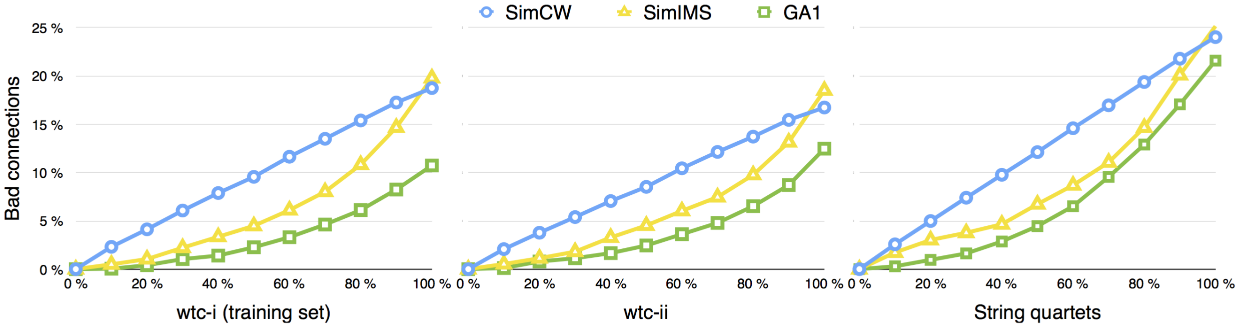

Figure 4. Errors done during the successive connection steps. The lower the curves, the better. Coefficients SimCW (blue): the error rate is almost constant. Coefficients SimIMS (yellow): the first connections are more reliable. Coefficients GA1 (green): the first connections are even more reliable, enabling to improve the algorithm by stopping before too much bad connections happen. The highest number of bad connections for string quartets (compared to fugues) is probably due to a less regular polyphonic writing, with in particular stylistic differences leading to larger intervals.

7. REFERENCES

[1] Albert Donally Bethke. Genetic algorithms as function optimizers. In *ACM Computer Science Conference*, 1978.

[2] Elaine Chew and Xiaodan Wu. Separating voices in polyphonic music: A contig mapping approach. In *International Symposium on Computer Music Modeling and Retrieval (CMMR 2005)*, pages 1–20. 2005.

[3] Michael Scott Cuthbert and Christopher Ariza. music21: A toolkit for computer-aided musicology and symbolic music data. In *International Society for Music Information Retrieval Conference (ISMIR 2010)*, pages 637–642, 2010.

[4] Reinier de Valk, Tillman Weyde, and Emmanouil Benetos. A machine learning approach to voice separation in lute tablature. In *International Society for Music Information Retrieval Conference (ISMIR 2013)*, pages 555–560, 2013.

[5] Diana Deutsch, editor. *The psychology of music*. Academic Press, 1982.

[6] Nicolas Guiomard-Kagan, Mathieu Giraud, Richard Groult, and Florence Levé. Comparing voice and stream segmentation algorithms. In *International Society for Music Information Retrieval Conference (ISMIR 2015)*, pages 493–499, 2015.

[7] David Huron. Tone and voice: A derivation of the rules of voice-leading from perceptual principles. *Music Perception*, 19(1):1–64, 2001.

[8] David Huron. Music information processing using the Humdrum toolkit: Concepts, examples, and lessons. *Computer Music Journal*, 26(2):11–26, 2002.

[9] Asako Ishigaki, Masaki Matsubara, and Hiroaki Saito. Prioritized contig combining to segragate voices in polyphonic music. In *Sound and Music Computing Conference (SMC 2011)*, volume 119, 2011.

[10] Jürgen Kilian and Holger H Hoos. Voice separation – a local optimization approach. In *International Conference on Music Information Retrieval (ISMIR 2002)*, 2002.

[11] Phillip B Kirlin and Paul E Utgoff. Voise: Learning to segregate voices in explicit and implicit polyphony. In *International Conference on Music Information Retrieval (ISMIR 2005)*, pages 552–557, 2005.

[12] Hanna M Lukashevich. Towards quantitative measures of evaluating song segmentation. In *International Conference on Music Information Retrieval (ISMIR 2008)*, pages 375–380, 2008.

[13] Andrew McLeod and Mark Steedman. HMM-based voice separation of MIDI performance. *Journal of New Music Research*, 45(1):17–26, 2016.

[14] Dimitrios Rafailidis, Alexandros Nanopoulos, Yannis Manolopoulos, and Emilios Cambouropoulos. Detection of stream segments in symbolic musical data. In *International Conference on Music Information Retrieval (ISMIR 2008)*, pages 83–88, 2008.

[15] Dimitris Rafailidis, Emilios Cambouropoulos, and Yannis Manolopoulos. Musical voice integration/segregation: Visa revisited. In *Sound and Music Computing Conference (SMC 2009)*, pages 42–47, 2009.

[16] David Temperley. *The Cognition of Basic Musical Structures*. The MIT Press, 2001.