MorpheuS: automatic music generation with recurrent pattern constraints and tension profiles

Dorien Herremans
Centre for Digital Music
School of Electrical Engineering and Computer Science
Queen Mary University of London
Mile End Road, E1 4NS London, UK
Email: d.herremans@qmul.ac.uk

Elaine Chew
Centre for Digital Music
School of Electrical Engineering and Computer Science
Queen Mary University of London
Mile End Road, E1 4NS London, UK
Email: elaine.chew@qmul.ac.uk

***Abstract*—Generating music with long-term structure is one of the main challenges in the field of automatic composition. This article describes MorpheuS, a music generation system. MorpheuS uses state-of-the-art pattern detection techniques to find repeated patterns in a template piece. These patterns are then used to constrain the generation process for a new polyphonic composition. The music generation process is guided by an efficient optimization algorithm, variable neighborhood search, which uses a mathematical model of tonal tension to derive its objective function. The ability to generate music according to a tension profile could be useful in a game or film music context. Pieces generated by MorpheuS have been performed in live concerts.**

I. Introduction

Automated composition has made great strides in the past few decades since the advent of computers. One of the biggest remaining challenges in the field of automatic music composition is the crafting of long term structure. Most computer-generated music sounds good in the short term, but lacks recurring and developing themes, and long-term form. In [1], the authors developed an efficient variable neighborhood search (VNS) metaheuristic that generates music with a given semiotic structure (e.g. ABABBC). This research showed that considering music generation in an optimization context offers a viable way of constraining, and thus ensuring, structure. This paper expands the earlier work to polyphonic music, adding to the system a pattern detection algorithm and a computational model for musical tension. In the resulting system named MorpheuS, the pattern detection algorithm [2] finds reoccurring note patterns in a template piece. These patterns are then used to constrain structure in a newly generated piece. An early prototype by Screene and Wiggins [3] attempts to combine detected patterns with generation by Markov models. Because of the recursive nature of the pattern computation, their approach requires non-standard hierarchical models. Further research is needed on how to construct these models and best generated from them. Our research avoids this problem by using an optimization approach that constrains the patterns. MorpheuS generates music according to a tension profile that can be computed from the template piece or specified by a user. Tension shapes our experience of music listening; it is especially relevant in the context of film or game music. Figure 1 displays an overview of MorpheuS' functional architecture. The system can be used to generate polyphonic music of any type or genre.

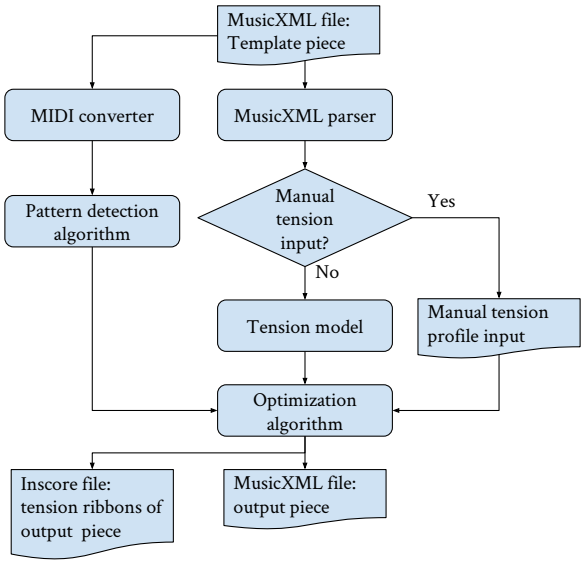


Fig. 1: Overview of MorhpeuS' architecture.

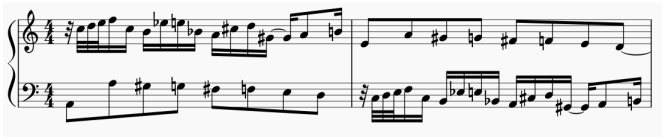
The next two sections give an overview of the tension model and pattern detection module. Section IV defines music generation formally as an optimization problem, and then describes the optimization algorithm. The final sections discuss the computational and musical results.

II. Quantifying tension

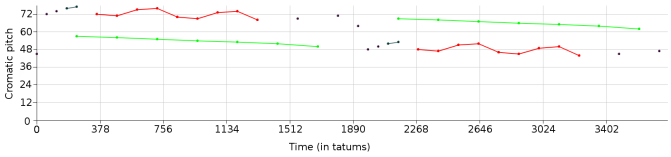
The experience of tension is an intrinsic part of music listening. The authors have developed a mathematical model for quantifying tonal tension [4] based on the spiral array, a three-dimensional model of tonality [5]. To calculate tonal tension, a piece is first segmented into equal-length segments. The notes from each segment are mapped to a "cloud" of points in the spiral array. The model includes three measures of tension: cloud diameter, which captures the dispersion (dissonance) of the notes in tonal space; cloud momentum, which quantifies the degree of tonal change between clouds; and, tensile strain, which captures the tonal distance from the cloud center the global key.

III. Pattern detection

MorpheuS uses two greedy compression algorithms, COSIATEC and SIATECCompress, to detect patterns in a template piece [2]. Both algorithms discover repeated patterns such as themes and motives. Starting from a point-set representation of the score, they compute a compressed encoding in the form of a set of translational equivalence classes of maximal-length patterns (MTP TECs). Figure 2 shows an example COSIATEC output. The algorithm clearly detects the two main patterns that alternate between the right and left hand in the first and second bar.



(a) First two bars of Bach's 20th prelude (Book II).



(b) Patterns detected with COSIATEC [2]

.

Fig. 2: COSIATEC applied to a short musical excerpt.

COSIATEC generates a strict partitioning of the input dataset, whereas patterns may share points in SIATECCompress. SIATECCompress achieved 45% precision and 60% recall on the thematic analysis task, and may potentially be more useful in the music analysis context [6].

IV. Optimization problem

MorpheuS uses an optimization approach in its music generation to constrain global structure, which is formulated as repeated patterns, and fit music to a tension profile. This section formally defines the music generation process as a combinatorial optimization problem.

a) Variables: MorpheuS takes a template piece as input and treats its rhythm and dynamics as constants in the generated piece. The aim of the algorithm is to find a set of pitches, x, for each note of the template piece that satisfies the constraints and minimizes the objective function.

b) Objective function: The objective is to find a solution x that most closely fits the tension profile. The tension profile of a solution x consists of three parts: $T_i(x)$, for each of the three tension measures $i \in \{0, 1, 2\}$ described in Section II. The target tension profile $T_i(t)$ can be based on the template piece t, or it can directly inputted by the user. The distance between the desired tension and that of a solution x is measured by the Euclidean distance between the two signals:

$$D_i(x) = \sqrt{(T_i(x) - T_i(t))^2}. \tag{1}$$

The objective function is the sum of the distances for each of the three tension measures:

$$D(x) = \sum_{i=0}^{2} D_i(x). \tag{2}$$

The aim of the optimization algorithm is thus to minimize the distance $D(x)$.

c) Hard constraints: The patterns detected on the template piece by the algorithms described in Section III serve as hard constraints for the pitches, x. This creates recurring patterns that serve as themes and motives. The algorithm only has to decide on new pitches for the original occurrence of a pattern, and can automatically set the pitches in other repetitions of the pattern. This reduces the number of variables in the set x for which the algorithm has to find new values and this makes the algorithm more computationally effective.

A further hard constraint is the pitch range for each hand. The lowest and highest pitch for each hand follows that of the template piece. All pitches within the set range are allowed.

d) Soft constraints: The user can fix certain pitches of notes in the rhythmic template if desired. An additional term was added to the objective function $D(x)$ from Equation 2 to constrain the pitch ($pitch(n_j)$) of j notes to a set pitch ($setpitch(n_j)$). The resulting objective function for a current piece x thus becomes:

$$D'(x) = D(x) + \sum_{j=0}^{j} a \times C(n_j), \tag{3}$$

where

$$C(n_j) = \begin{cases} 0, & \text{if } pitch(n_j) = setpitch(n_j) \\ 1, & \text{otherwise} \end{cases}$$

and a is an arbitrary large number.

V. Variable neighborhood search algorithm

This section outlines the role metaheuristics have played in music generation then describes MorpheuS' VNS algorithm.

A. Metaheuristics and music

Solving the optimization problem defined in the previous section is a computationally complex task, since the number of possible solutions increases exponentially with the number of notes in the template piece. Limited research exists on solving the music generation problem with exact methods such as integer programming [7], since these systems face challenges such as running out of physical computer memory before a solution is found. Metaheuristic optimization techniques offer a valid alternative to exact algorithms. They do not guarantee an optimal solution, but use a variety of strategies to reach a good solution in a reasonable amount of time [8].

Metaheuristics are typically grouped into three classes [9]. The first class consists of *population-based* metaheuristics (evolutionary algorithms, path relinking,...), whereby a set of solutions (population) are combined to create new ones. In music, the first genetic algorithm was developed in 1991 by [10].

The second class of metaheuristics are the *constructive* metaheuristics, which build solutions from their constituent parts, such as GRASP and ant colony optimization. The first ant colony algorithm applied to a music generation problem was developed in 2007 to harmonize baroque music [11]. Lastly, algorithms that iteratively improve a single solution (e.g. tabu search and variable neighborhood search (VNS)) form the group of *local search* algorithms [9]. Local search techniques have, for instance, been used at IRCAM to solve music constraint problems [12] and by the first author, who developed the first VNS to generate counterpoint [13].

B. Variable Neighborhood Search

Variable neighborhood search, or VNS, has been applied to a wide range of combinatorial problems [14], including vehicle routing [15] and project scheduling [16]. [14] found that, for several of these problems, VNS outperforms existing heuristic algorithms and is computationally efficient. MorpheuS' implementation of VNS expands on the first author's previous work [13] to handle complex polyphonic music, detected patterns and tension profiles.

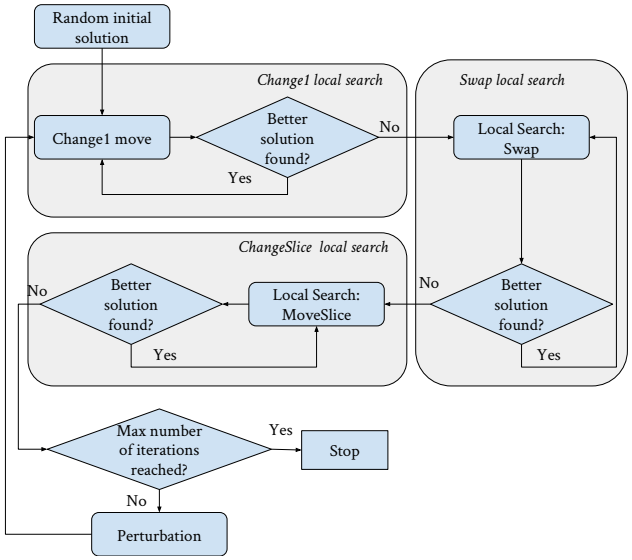


Fig. 3: VNS algorithm as implemented in MorpheuS.

Figure 3 shows a system diagram of the VNS algorithm as implemented in MorpheuS. The algorithm first assigns uniformly random pitches to each note to obtain an initial solution x. Then it iteratively searches locally for a solution with a lower $D'(x)$ value by making small changes (called *moves*) to the current solution. The three types of moves used by MorhpeuS are displayed in Figure 4. They are *change1*: change the pitch of one note any other allowed pitch; *swap*: swap the pitch of any two notes; and, *changeSlice*: change the pitch of 2 notes in a vertical time slice to any other pitch.

The set of all solutions x' that can be reached from the *current* solution with only one move is referred to as the *neighborhood* $N(x)$. A *first descent* strategy, which selects the first feasible element from the neighborhood that improves the current solution, was implemented to further speed up

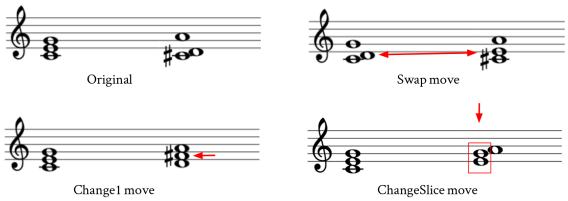


Fig. 4: The three types of moves used by the VNS algorithm.

the VNS. The algorithm iteratively searches through a local neighborhood, until no further feasible solution can be found, at which point it has arrived at a local optimum and switches to a different type of neighborhood (e.g. swap move) [17]. When no further improvement can be found in any neighborhood, the VNS perturbs the current solution by randomly re-assigning 12% of the pitches. This strategy was shown to work in [1] and allows the search to resume.

An acceleration strategy was implemented that forces the moves to be applied chronologically from the start to the end of the piece. The algorithm goes through each note, starting in the beginning of the piece, and tries to apply a move. When a particular move is successful, this affects the tension profile only in its immediate vicinity; therefore, the algorithm will only backtrack 4 time slices to resume the search. This strategy significantly improves the speed of the optimization.

VI. Results

The VNS algorithm described above was applied with Bach's Prelude No. 1 in C major in his *Well-tempered Clavier* (Book I) as a template. Figure 5 shows how the value of the objective function evolves each time a move is performed. The objective function value of the initial solution is extremely high, meaning that the solution is poor, but improves quickly with each iteration. While the improvements slow down after a while, the dashed "peaks", which represent the perturbations, show that the algorithm is able to escape from local optima.

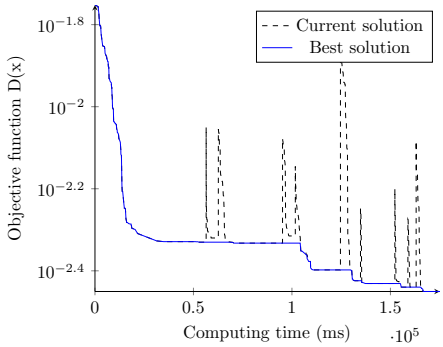


Fig. 5: Evolution of objective function values over time during a run of the VNS algorithm.

The original, initial, and final musical scores are displayed in Figure 6 together with their cloud diameter tension profiles. The initial (random) solution displayed in Figure 6b clearly displays some of the original patterns, which were detected using COSIATEC and hard constrained during generation.

Without a tension constraint, the fairly eclectic combination of notes of the initial solution results in tension ribbons that are significantly different from those of the original piece. The result after 1 iteration of the VNS is shown in Figure 6c. The piec type or genre. e is now much more tonal, and has a tension profile that resembles the original piece.

A piece generated by MorpheuS using the same method but based on Haydn's Sonata in Eb, Hob XVI:45 (finale) has been performed at the Center for New Music in San Francisco and the TENOR conference in Cambridge. The reader is invited to listen to MorhpeuS' compositions online[1].

While the music sounds promising and contains reoccurring patterns, there is still room for improvement. Some future research directions include adding constraints for playability, and imposing statistical properties of particular music styles.

VII. Conclusion

MorpheuS uses an efficient variable neighborhood search optimization algorithm to generate polyphonic music with recurring themes and tension profiles. Patterns detected in a template piece must recur in the generated music, which must also follow a designated tension profile. The ability to generate according to a tension profile could be useful in a game of film music context. The output of MorpheuS is promising and has been performed on multiple occasions around the world. In future research, the authors aim to make the algorithm even more efficient. The quality of the musical output could also be improved by imposing more constrains such as those related to playability and to statistical properties of a style of music.

VIII. Acknowledgment

This project is funded in part by the European Unions Horizon 2020 research and innovation programme under grant agreement No 658914.

References

[1] D. Herremans, S. Weisser, K. Sörensen, and D. Conklin, "Generating structured music for bagana using quality metrics based on Markov models," *Expert Systems with Applications*, vol. 42, no. 21, pp. 7424–7435, 2015.

[2] D. Meredith, "COSIATEC and SIATECCompress: Pattern discovery by geometric compression," in *International Society for Music Information Retrieval Conference*, 2013.

[3] F. Screene and G. A. Wiggins, "Learning large scale musical form to enable creativity," in *Proceedings of the sixth International Conference on Computational Creativity*, Utah, USA, 2015.

[4] D. Herremans and E. Chew, "Tension ribbons: Quantifying and visualising tonal tension," in *Second International Conference on Technologies for Music Notation and Representation (TENOR)*, Cambridge, UK, 05/2016 2016.

[5] E. Chew, "Mathematical and computational modeling of tonality," *AMC*, vol. 10, p. 12, 2014.

[6] D. Meredith, "Music analysis and point-set compression," *Journal of New Music Research*, vol. 44, no. 3, pp. 245–270, 2015.

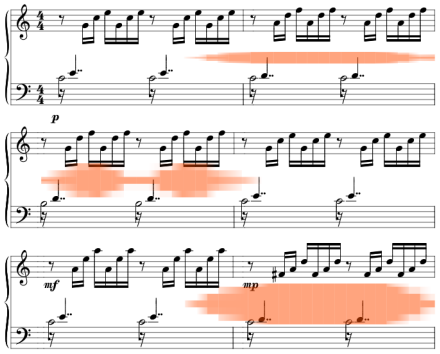
[7] T. Tanaka and K. Fujii, "Describing global musical structures by integer programming on musical patterns," in *Mathematics and Computation in Music*. Springer, 2015, pp. 52–63.

[8] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison," *ACM Computing Surveys (CSUR)*, vol. 35, no. 3, pp. 268–308, 2003.

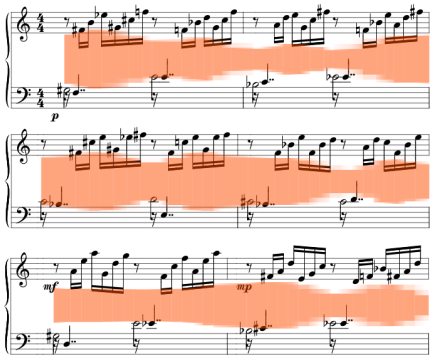
[9] K. Sörensen and F. W. Glover, "Metaheuristics," *Encyclopedia of Operations Research and Management Science*, pp. 960–970, 2013.

[10] A. Horner and D. Goldberg, "Genetic algorithms and computer-assisted music composition," *Urbana*, vol. 51, no. 61801, pp. 437–441, 1991.

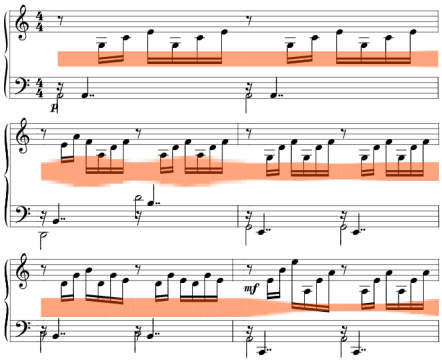
[1] dorienherremans.com/morpheus



(a) Original piece (opening).



(b) Random initial solution.



(c) Optimized solution after 1 iteration of the algorithm.

Fig. 6: MorpheuS example with cloud diameter tension profile.

[11] M. Geis and M. Middendorf, "An ant colony optimizer for melody creation with baroque harmony," in *IEEE Congress on Evolutionary Computation*. IEEE, 2007, pp. 461–468.

[12] C. Truchet and P. Codognet, "Musical constraint satisfaction problems solved with adaptive search," *Soft Computing-A Fusion of Foundations, Methodologies and Applications*, vol. 8, no. 9, pp. 633–640, 2004.

[13] D. Herremans and K. Sörensen, "Composing first species counterpoint with a variable neighbourhood search algorithm," *Journal of Mathematics and the Arts*, vol. 6, no. 4, pp. 169–189, 2012.

[14] N. Mladenovic and P. Hansen, "Variable neighborhood search," *Computers & Operations Research*, vol. 24, no. 11, pp. 1097–1100, 1997.

[15] O. Bräysy, "A reactive variable neighborhood search for the vehicle-routing problem with time windows," *INFORMS Journal on Computing*, vol. 15, no. 4, pp. 347–368, 2003.

[16] K. Fleszar and K. Hindi, "Solving the resource-constrained project scheduling problem by a variable neighbourhood search," *European Journal of Operational Research*, vol. 155, no. 2, pp. 402–413, 2004.

[17] N. Mladenovic and P. Hansen, "Variable neighborhood search," *Computers & Operations Research*, vol. 24, no. 11, pp. 1097–1100, 1997.