This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAFFC.2017.2737984, IEEE Transactions on Affective Computing

MorpheuS: generating structured music with constrained patterns and tension

Dorien Herremans, *Member, IEEE,* and Elaine Chew, *Member, IEEE,*

Abstract—Automatic music generation systems have gained in popularity and sophistication as advances in cloud computing have enabled large-scale complex computations such as deep models and optimization algorithms on personal devices. Yet, they still face an important challenge, that of long-term structure, which is key to conveying a sense of musical coherence. We present the MorpheuS music generation system designed to tackle this problem. MorpheuS' novel framework has the ability to generate polyphonic pieces with a given tension profile and long- and short-term repeated pattern structures. A mathematical model for tonal tension quantifies the tension profile and state-of-the-art pattern detection algorithms extract repeated patterns in a template piece. An efficient optimization metaheuristic, variable neighborhood search, generates music by assigning pitches that best fit the prescribed tension profile to the template rhythm while hard constraining long-term structure through the detected patterns. This ability to generate affective music with specific tension profile and long-term structure is particularly useful in a game or film music context. Music generated by the MorpheuS system has been performed live in concerts.

Index Terms—Affective Computing, Music, Music retrieval and generation, Affective computing applications, Sound and Music Computing, Entertainment, Variable Neighborhood Search, Pattern Detection

1 Introduction

Technologies for digital music have become increasingly important, bolstered by rising global expenditures in digital music in excess of 64 billion USD in 2014 alone [1]. The popularity and relevance of automatic *music generation* has recently been underscored by the launch of Google's Magenta project[1], "a research project to advance the state of the art in machine intelligence for music and art generation". In this research, we develop a music generation system, called Morpheus [2], that tackles one of the biggest remaining challenges in the field of automatic music composition: long term structure. Long term structure is that which generates coherence over larger time scales from phrases up to the entire piece; it refers to more than simply traditional ABA form, and includes the modulation of features such as loudness and tension, and the use of recurrent patterns and motivic transformations over time, so as to generate coherence over these large time scales. While most existing music generation systems create pieces that may sound good over a short time span, these outputs often lack long-term structure. MorpheuS can take any existing polyphonic music piece as template and morph it into a new piece with a predefined tension profile. The new piece will also preserve the same long term structure (i.e. pattern structure) as the template.

To this day, it remains notoriously difficult to enforce constraints (e.g. long-term structure) in music generation systems based on machine learning methods such as Markov models [3]. In previous research, the first author therefore developed a novel method for constraining long-term structure through an optimization-based approach, combined with machine learning. The proposed framework consisted of an efficient variable neighborhood search (VNS) optimization algorithm that is able to generate melodies (or monophonic music) with a fixed semiotic structure (e.g. AABBACA) [4, 5] and evaluates its solution through the Euclidean distance between a Markov model built on a corpus and one trained on the generated piece. This research shows that the approach offers a viable way of constraining structure. In the current paper, the VNS algorithm is expanded to generate *complex polyphonic music*. Although the algorithm is able to work with any type of polyphonic music, as a proof of concept, we focus on piano music in this research.

A second novel aspect of MorpheuS is the inclusion of an original *tension* model. Tension shapes our music listening experience. In some music genres, such as game and film music, there often is a direct relationship between tension, a narrative, and the emotion perceived [6]. In this research, we aim to generate music that has a predefined tension profile through time. The target tension can be specified by the user, or calculated from a template piece using a computational model developed by the authors [7]. A system like Morpheus that can generate music having a particular tension profile could be employed by film makers, composers, and game programmers when music matching a specific narrative is desired.

The third contribution of this research is the integration of a state-of-the-art pattern detection algorithm [8], which is used to find recurring patterns and themes in the template piece. MorpheuS then uses the patterns found to configure

- *E. Chew is with the Centre for Digital Music, School of Electronic Engineering and Computer Science, Queen Mary University of London, UK.*
- *D. Herremans was at the Centre for Digital music (see above) for most of the research, and is currently with the Information Systems Technology and Design Pillar at Singapore University of Technology and Design. Email: see http://dorienherremans.com/contact*

Manuscript received 000, 2016; revised 000, 20xx.

1. https://magenta.tensorflow.org/welcome-to-magenta

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAFFC.2017.2737984, IEEE Transactions on Affective Computing

the structure in a newly generated piece by introducing the patterns as hard-constraints during the generation process.

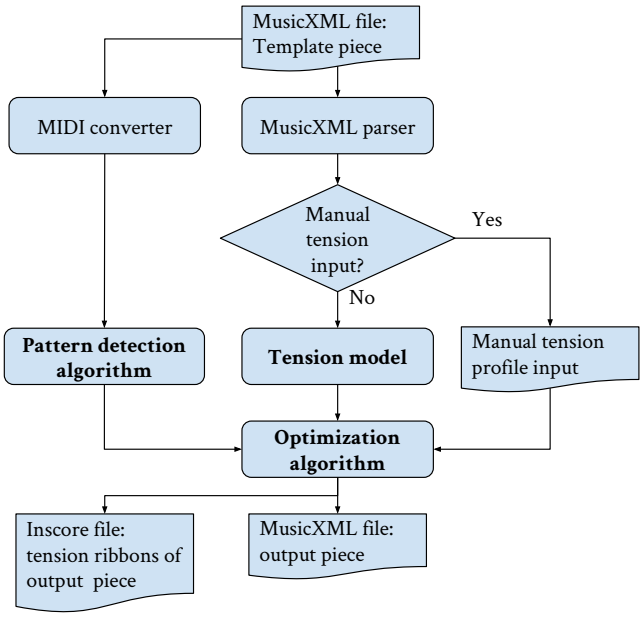


Fig. 1: Overview of MorhpeuS' architecture.

MorpheuS' functional architecture is displayed in Figure 1. The system is implemented entirely in Java. The main modules of the algorithm, in bold, will be further discussed in Sections 3, 4, and 5. Before embarking on these discussions, we briefly survey related systems in the next section.

2 Literature review

Before examining the individual components of the MorpheuS system, we give an overview of related research. The first subsection covers different techniques used in automatic music generation; the focus of this overview lies mainly on metaheuristic optimization algorithms. Next, we focus on the two types of long term structure that are incorporated in the system. A first aspect of long-term structure is ensured through a tension profile. By requiring that the music adhere to a coherent tension profile, MorpheuS can generate music displaying specific tension characteristics throughout the piece. This makes the output particularly well suited to game or film music scenarios. An overview thus follows of previous research on music generation with a narrative and tension. We then address the second aspect of long-term structure in MorpheuS, namely, recurring patterns in generated music, providing a review of how this has previously been tackled by researchers in automatic composition. For a more general survey of current music generation systems, the reader is referred to [9].

2.1 Generation techniques for music

The idea that computers could compose music is as old as the computer itself. Ada Lovelace, who worked with Charles Babbage on the Difference Engine, predicted that the engine when realised could one day "compose elaborate and scientific pieces of music of any degree of complexity or extent" [10].

Since then, many automatic systems for music generation have been developed. In the 50s, the first piece composed entirely by a computer, "The Illiac Suite", was generated by a stochastic rule-based system [11]. More recently, a number of systems based on Markov models were developed for simple melody generation [12, 13], to harmonization [14, 15] and improvisation systems [16, 17, 18]. In recent years deep learning models have entered the scene [19, 20, 21, 22]. While many of these systems produce output that sounds good on a note-to-note level, they often lack long-term coherence. We aim to tackle this challenge in this research by employing pattern detection techniques. In order to exploit the patterns found, we opt for an optimization-based approach, which allows us to constrain structure.

In Section 5 the problem of generating music with long-term structure is defined as a combinatorial optimization problem. This problem is computationally complex to solve, as the number of possible solutions grows exponentially with the length of the piece. As an example, a piece consisting of only 32 notes, with 24 possible pitches per note, has 32^{24} possible solutions.

There have only been limited attempts at solving music generation problems with *exact methods* such as integer programming. For example, Cunha et al. [23] uses integer programming with structural constraints to generate guitar solos based on existing licks. Their objective function is based on music theoretic rules. In research by [24], the authors propose a method to generate counterpoint—independent linear voices that combine to form a single harmonic texture. They formulate the generation task as an integer programming problem that uses existing composition rules as constraints to control global structure. However, this work remains a theoretical formulation, with no solution method as yet implemented.

In order to overcome the difficulty and often long computational run times required to calculate exact solutions to optimization problems, many practical applications use *metaheuristics*. A metaheuristic is defined by Sörensen [25] as "a high-level problem-independent algorithmic framework that provides a set of guidelines or strategies to develop heuristic optimization algorithms. The term is also used to refer to a problem-specific implementation of a heuristic optimization algorithm according to the guidelines expressed in such a framework." These techniques often employ a variety of strategies to find a good solution in a limited amount of computing time; they do not guarantee an optimal solution, but typically good solutions are found [26].

There exist three main groups of metaheuristics: population-based, constructive, and search-based algorithms [27]. The first group, which includes evolutionary algorithms, has seen recent gains in popularity in the literature. Population-based algorithms get their name from the fact that they inter-combine a set of solutions (population) to create new ones. Horner and Goldberg [28] were the first to develop a genetic algorithm for music generation. These techniques have later been used to generate jazz solos [29], counterpoint style music [30, 31, 32], and rhythmic patterns [33, 34], and to combine fragments for orchestration [35].

The second group, constructive metaheuristics, gradually build solutions from their constituent parts, for example, by growing individual notes in sequence. An example of this category includes ant colony optimization, which was first applied to music in 2007 to harmonize baroque music [36].

The third category, local search-based heuristics, typically make iterative improvements to a single solution. They include algorithms such as iterated local search, simulated annealing, and variable neighborhood search [27]. An example of these techniques in the field of music generation can be found in the research of Davismoon and Eccles [37], who used simulated annealing to generate music according to a fitness function that was derived from a Markov model. The first author of the current paper, was the first to develop a variable neighborhood search (VNS) algorithm that was able to generate counterpoint music [38]. This VNS was shown to outperform a genetic algorithm on the same task and has been modified in the current research to generate complex polyphonic music.

2.2 Narrative and tension

The tension profile, which is integrated in the algorithm so as to shape the tension of the generated music, is particularly important when generating music with a narrative, or program music. Program music has a long and illustrious history, a well-known example being Richard Strauss' "Don Quixote". Such narrative music tells a story, by using a set of organizational, representational, and discursive cues that deliver story information to the audience. Such cues can include tension profiles, leitmotifs (recurring melodic fragments associated with a person, idea, or story situation), and others. All of these elements typically elicit varying emotion responses during the unfolding of a piece when synchronized with simultaneous media such as video or game play. Existing systems in the domain of video and game music are discussed, followed by a more focused overview of literature on tension models.

2.2.1 Generating film music

A prominent application of music with narrative is film music. Music has been shown to be an important source of perceived emotion in film [6, 39]. While Prechtl et al. [40] has conducted research on generating music that evokes basic emotions in the context of games, very little research exists on developing music generation systems that follow the emotion content of films. Even commercial applications such as the web-based music generation app, Jukedeck[2], do not yet take into account the emotion narrative. Jukedeck generates background music for YouTube videos using a combination of rules and deep learning.

A prototype system that generates background music and sound effects for short animation films was developed by Nakamura et al. [41]. For each scene, music (harmony, melody, rhythm) is generated based on rules from music theory whilst taking into consideration the mood, the intensity of the mood, and the musical key of the previous scene. The sound effects are determined by the characteristics and intensity of the movements on screen. In the next subsection we will discuss the related topic of game music.

2.2.2 Game music – blending

The most dynamic form of narrative in music can be found in computer games, whereby a user creates his or her own unique scenario when moving through the game. The accompanying music needs to follow and support the suspense and emotion of the current game play. Game music is rarely generated on the fly. Short audio files are generally cross-faded together as the player moves through different game states [42]. An exception to this common practice can be seen in the game, Depression Quest[3], as the music is generated dynamically as the user moves through the different scenarios of the game. With current cross-fading techniques, it is not uncommon for two fragments to clash rhythmically or harmonically, causing a jarring change in the music. The automatic DJ-system developed by Müller and Driedger [43] ensures smooth blending, yet the audio fragments need to be harmonically and rhythmically similar for the blending to work successfully. By restricting the range of harmonies and rhythms in this way, one also limits the musical variations and expressive capacity of the music. To overcome this limitation, some games implement procedural music approaches that use control logics or rules that control playback. One of the first procedural music and audio approaches for computer games can be found in the game 'Otocky' for the Famicom platform. Otocky is a side-scrolling shooter, whereby the player controls a ship that fires balls at both enemies and flying musical notes. The melody part is formed by the player's firing behavior and in-game mechanics, and is rendered on top of a two note bass line [44]. For an overview of procedural music techniques, the reader is referred to Collins [44].

More recent work has focused on incorporating elements of tension and emotion into adaptive game music. Prechtl [45] created a system that generates music from scratch instead of using existing fragments. Prechtl uses a Markov model for chord generation that takes into account emotion parameters such as alarm or danger. His study uncovered correlation between mode and valence, and between tempo/velocity and arousal.

Casella and Paiva [46] created MAgentA (not to be confused with Google's music generation project Magenta), an abstract framework for a video game background music generation that aims to create "film-like" music based on the mood of the environment using a cognitive model. At the time of publication, the authors mention that the framework was being integrated from the abstract level into the FantasyA Virtual Environment, but no further references could be found.

The system developed by Brown [47] makes use of the concept of "Leitmotifs" , commonly used in Western opera. Brown [47]'s system stores different forms of each motif corresponding to varying degrees of harmonic tension and formal regularity. This allows the system to choose the appropriate fragment corresponding to the current state and story of a game. The reader is referred to Collins [44] for a more complete overview of dynamic music in games.

2. jukedeck.com

3. https://isaacschankler.bandcamp.com/album/depression-quest-ost

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAFFC.2017.2737984, IEEE Transactions on Affective Computing

2.2.3 Generating music with tension

Musical tension is an important tool for evoking emotion. According to Farbood [48], the way that different listening experiences translate into actual 'affect' is a complex process. Musical tension, measured based on musical structures, provides a stepping stone to understanding and quantifying subjective, emotional responses. The link between emotion and tension has become apparent in many studies [49, 50, 51, 52]. If a music generation system can generate music according to given tension profiles, it becomes directly relevant for applications in game and film music. Recent research has made advances in the quantification of aspects of musical tension, such as tonal tension [7, 53], even combining them to produce a composite model [54]. Based on an extensive empirical experiment, Farbood [48] built a tension model that takes into account multiple musical parameters to obtain one comprehensive tension rating. Farbood implemented an earlier version of her tension model, that does not yet integrate features, [54] in the graphical computer-assisted composition system called Hyperscore, in which users can intuitively edit and visualize musical structures as they compose music [55]. Hyperscore shows low-level and high-level musical features (such as color, shape, dynamics, harmonic tension) and maps them to graphical elements which can be controlled by the user when crafting compositions. Thus, a user can draw a tension profile and Hyperscore will generate music with a similar profile.

Similarly, Browne and Fox [56]'s system arranges prewritten motifs according to a pre-specified tension profile using simulated annealing. An artificial neural network was used to compute tension profiles. The objective function of the algorithm was then formed by taking the Kullback-Leibler divergence between the desired and observed tension profiles. The optimal arrangement was then taken to be the one that minimizes this distance.

In this study, we will focus on multiple aspects of *tonal* tension independently versus considering a composite tension characteristic. The tonal component has proven to be a particularly strong structural influence on emotions. In Rutherford and Wiggins [57]'s scary music study, they conclude that more scary music is generated by breaking the Western tonal music rules. This result was empirically verified by users who rated the scariness of the generated music. The computational model used in this research for calculating tonal tension is discussed in more detail in Section 3. The next section considers the importance of patterns in music.

2.3 Structural patterns in generated music

Music is more than just a succession of notes that only needs to sound good in the short term. Having long-term structure: motives, patterns and variations of those patterns are essential for an enjoyable and satisfying listening experience. Music generation systems that use traditional statistical sampling methods based on Markov models typically only ensure short term relationships between notes [5].

One approach to obtaining long term structure was implemented by Roig et al. [58], whose system concatenates rhythmic and melodic patterns in order to form new melodies based on a combination of rules and a statistical method. More complex statistical learning methods, such as recursive neural networks have recently gained popularity in the field of music generation due to the availability of large amounts of digital music data and increased computing power. While the first neural network for melody generation was implemented in the late 80s [59], this approach has become more relevant due to the ability of these networks to learn complex relationships between notes given a large enough corpus. Recent research in audio transcription by Boulanger-Lewandowski et al. [21] shows promising results for music generation as well. They use a piano roll representation for polyphonic pieces to build a Recurrent Temporal Restrictive Bolzmann Machine (RT-RBM)-based model. This model learns harmony and melody, and local temporal coherence. Long term structure is not yet captured by the model.

Another approach to long-term structure is explored by Herremans et al. [5] who examined the integration of Markov models in an optimization algorithm. By looking at different ways a statistical model can be used to construct an objective function, the approach ensures that the generated music has the same statistical distribution of features as a target dataset of pieces. By treating the problem of music generation as an optimization problem, Herremans et al. were able to impose larger-scale structure (e.g. ABBAC) on the generated music, in addition to short term statistical constraints. The resulting optimization problem was solved by a VNS metaheuristic to generate music for bagana, an Ethiopian lyre. This approach is extended in the current research to polyphonic music, with automatic detection of more complex long term patterns in the template piece. The detection method is described in greater detail in Section 4, after the next section, which focuses on the tension model.

3 Quantifying Tension in Music

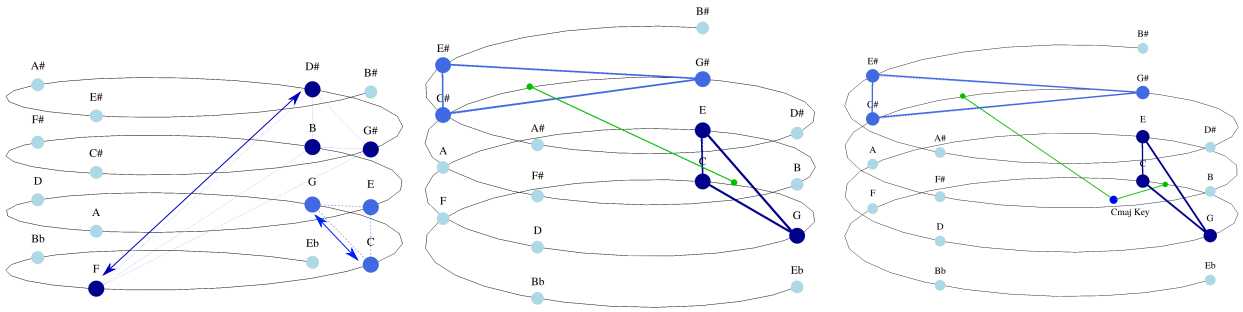
Tension is a composite characteristic, which makes it very hard to capture or measure in a quantitative way. According to Mary Farbood [48], "increasing tension is a feeling of rising intensity or impending climax, while decreasing tension can be described as a feeling of relaxation or resolution" (p. 387). In Herremans and Chew [7], the authors developed a model for tonal tension based on the spiral array [60], a three-dimensional model for tonality. The relevant part of the model is briefly described below, together with how it was implemented in the MorpheuS system to quantify tension in an optimization context.

3.1 The Spiral Array

The spiral array is a three-dimensional geometric model for tonality [60]. It consists of an outermost helix representing pitch classes (shown in Figure 2), and inner helices (not shown) representing higher level tonal constructs such as chords (major and minor) and keys (major and minor) successively generated from their lower level components. Any set of points in the spiral array can be weighted and summed to generate a *center of effect* (*c.e.*), representing the aggregate tonal effect of its components.

Tonal representations in the spiral array mirror close tonal relationships between the entities, such as a perfect

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAFFC.2017.2737984, IEEE Transactions on Affective Computing



(a) Cloud diameter of a C major chord (small) versus the Tristan chord (large).

(b) Cloud momentum from a C major chord to a C# major chord.

(c) Tensile strain of a C major and C# major chord in the key of C major.

Fig. 2: An illustration of the three tension measures in the pitch class helix of the spiral array.

fifth between pitches, by their proximity in 3D space. For example, pitches one fifth apart are adjacent to each other in the pitch class helix (e.g. C-G) and pitches one major third apart are arranged vertically above one another (e.g. C-E). Similarly, the most likely key or chord of a cluster of pitches can be identified through a search for the key representation nearest to the c.e. of the pitch cluster. The tension model in MorpheuS uses only the pitch class and the major and minor key helices. The spiral array takes pitch spelling into account, meaning that enharmonically equivalent (but differently-spelled) pitches, such as G# and Ab have different spatial representations. The interested reader can refer to [60] for a full description of the spiral array model.

3.2 A quantitative model of tension

The model, developed by the authors in [7], represents tonal tension as captured by the geometry in the spiral array. The software that calculates the tension according to this model is freely available online[4]. In order to calculate the tonal tension of a musical fragment, the piece is first divided into equal length segments, which can be mapped to clouds of points in the spiral array. The segment length is expressed in beats and can be set by the user (default setting is an $\frac{1}{8}$ note), a more detailed discussion of the effect of the segment length can be found in [7]. Based on these clouds, three measures of tonal tension can be computed:

Cloud diameter captures the diameter of the cloud of notes, which measures the dispersion of the cloud in tonal space.

Cloud momentum reflects the movement in tonal space between two consecutive clouds of notes, by quantifying the distance between their c.e.'s.

Tensile strain measures the distance between the c.e. of a cloud and the position of the global key in the array.

Figure 2 illustrates each of the three tension measures with the pitch class helix of the spiral array. On the left, the (small) cloud diameter of a C major triad is shown together with the (much larger) diameter of the tristan chord, a well known tense chord [61]. The large tonal distance traversed by a transition from the C major to the C# major chord is illustrated in Figure 2b, an example of the cloud momentum measure. Finally, Figure 2c visualizes the tonal distance between the c.e.'s of each these two chords and the key of C major, which shows two contrasting tensile strain measures.

For exact mathematical details of how to calculate the three measures of tension, the reader is referred to [7]. MorpheuS uses these three tension characteristics to evaluate the musical output and match it to given template tension profiles. The weights for each of these characteristics can be set by the user, reflecting the aspect of tension deemed most important. The integration of tension in the objective function of the optimization is discussed in detail in Section 5. The next section focuses on the the pattern detection algorithm implemented in MorpheuS to improve long-term coherence.

4 Detecting Recurring Patterns

Automatic recognition, description, classification and grouping of patterns are important problems in many domains [62]. Applications include image segmentation [63], human action recognition [64], face description [65], DNA sequence analysis [66], speech recognition [67], music genre recognition [68], and affective computing [69]. We focus on pattern analysis for polyphonic music.

When listening to a musical piece, a listener is able to recognize structure through perceiving repetition and relationships between parts of the piece of music. In order for a generated musical piece to sound natural, such patterns should exist. MorpheuS uses recurring patterns such as themes and motives from a template piece to fix these structural elements in a new composition. The detected patterns consist of groups of notes that can recur transposed in different places throughout the piece. There has been research on pattern detection techniques for music audio [70, 71], but our focus is on symbolic music (i.e. MIDI).

MorpheuS uses two state-of-the-art greedy compression-based algorithms for MIDI, COSIATEC and SIATECCompress [8], both based on Meredith's "Structure Induction Algorithm" (SIA) and SIATEC. SIA finds all the maximal translatable patterns (MTP) in a point-set and SIATEC discovers all translational equivalence classes (TECs) of MTPs in a point-set [72]. The performance of both algorithms is benchmarked on a compression task in [8]. The specific application of finding patterns for music generation requires special consideration when applying these algorithms. A

4. http://dorienherremans.com/tension

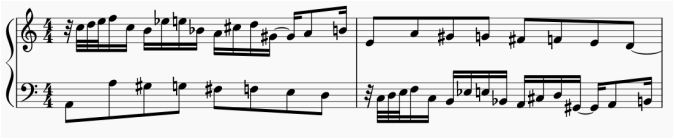
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAFFC.2017.2737984, IEEE Transactions on Affective Computing

discussion of the effect of parameter choices on the chosen pattern detection algorithm can be found in Section 7.1. MorpheuS offers the user a choice of which algorithm to use as each has its own strengths and weaknesses.

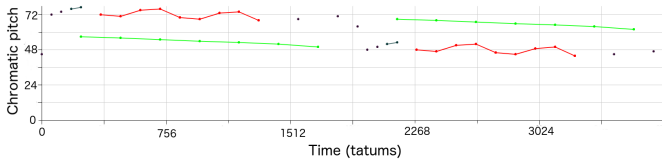
When applied to polyphonic MIDI files, the compression algorithms use a point-set representation of the score, which positions each note in a two-dimensional pitch/time space. They then compute a compressed encoding, which takes the form of a set of TECs of maximal-length patterns. An example output of COSIATEC in pitch/time space is shown in Figure 3, whereby the time is expressed in tatums, i.e., "the regular time division that mostly coincides with all note onsets" [73]. Two longer patterns (displayed in red and green) are detected in the figure. A pattern (or a repetition of a pattern) is shown as a connected sequence of points, and its TEC consists of a musical transposition of the original pattern (one translation unit is a semitone). The two main patterns in the fragment recur, transposed, in the other hand. The red pattern, for instance, starts on the fifth note of the right hand (C), it recurs in the second bar (left hand) at the fifth note, transposed two octaves down. The encoded representation of the red (wavy) pattern in the figure is as follows:

```
T(P(p(360,72),p(480,71),p(600,75),p(720,76),p(840,70)),
V(v(0,0),v(480,-2),v(1920,-24),v(2400,-26)))
```

whereby the set of pairs `P()` represents a maximal-length pattern, consisting of individual points `p()` in pitch/time space. The set `V()` contains the translation vectors `v()`, which when applied to `P()` form a translational equivalent pattern. The combination of the pattern and its translation vectors form `T()`, a translational equivalence class of maximal-length patterns (MTP TEC).



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



(b) Patterns detected with COSIATEC [8]

.

Fig. 3: COSIATEC applied to a short musical excerpt. Each TEC is represented with a different color.

The first algorithm implemented in Morpheus, SIATECCompress, runs SIATEC once to get a list of MTP TECs and then selects a subset of TECs that covers the input dataset [74]. The second algorithm, COSIATEC, on the other hand, iteratively uses SIATEC to find the best TEC, then removes this from the input dataset and repeats the process [8]. Both algorithms result in a set of TECs with high compression ratios that cover a point-set. The encodings generated by COSIATEC are generally more compressed, meaning that the size of the file listing all TECs is smaller. SIATECCompress produces patterns that may intersect, which may be more relevant in the context of music analysis, as a note may belong to more than one musically meaningful pattern. SIATECCompress performed best in the 2013 and 2014 MIREX competition on "Discovery of repeated themes and sections", and COSIATEC outperformed SIATECCompress on a Dutch folk song classification task with an accuracy rate of 84% [74].

The next section describes polyphonic music generation as an optimization problem which imposes the way the patterns repeat in the generated piece using hard constraints. This allows us to constrain the form and repetition structures of the newly generated piece.

5 Optimization Problem

In this research, generating music is modeled as an optimization problem. The main advantage of this is that it allows us to constrain global structure, consisting of repeated patterns, and to optimize the music to fit a tension profile. In this section, the resulting combinatorial optimization problem is formally defined.

5.1 Variables

The algorithm starts with a template piece whose rhythm and dynamics are treated as constants in the generated piece. The aim of MorpheuS is then to find a new set of pitches, x, for each note of the template piece, that minimizes the objective function and satisfies the repeated pattern constraints.

5.2 Objective function

The objective of the optimization problem is to find a solution x that matches a given tension profile as closely as possible. This tension profile can either be calculated from the template piece t or could be manually input by the user. It comprises of three parts: one for each of the three tension measures $i \in \{0, 1, 2\}$ from Section 3, represented as a vector $T_i(x)$ with length l_i. Since we want to match the tension profile of the solution x to that of the template t, we calculate the Euclidean distance between these two tension profiles:

$$D_i(x) = \sum_{j=1}^{l_i} \sqrt{(T_{ij}(x) - T_{ij}(t))^2}. \quad (1)$$

The sum of the distances between each of the tension measures forms the objective function $D(x)$, which we aim to minimize.

$$D(x) = \sum_{i=0}^{2} a_i \times D_i(x), \quad (2)$$

where a_i is the weight for tension measure i. The weights offer the user a way to specify the relative importance of certain tension measures. In this paper, the weights are all set to 1.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAFFC.2017.2737984, IEEE Transactions on Affective Computing

5.3 Soft constraints

In addition to the hard constraints to be described in the next section, the user can elect to fix certain pitches in the solution. In order to do this, an additional term was added to the objective function $D(x)$ which imposes an arbitrary high penalty if $pitch(n_j)$ of note j is not set to the required pitch ($setpitch(n_j)$):

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

whereby

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

and b is an arbitrary large number.

5.4 Hard constraints

A number of the variables (pitches) of the solution x are hard constrained to enforce the patterns detected in the template piece (as described in Section 4). This constraint ensures the recurrence of themes and motives in the output musical piece. The data structure used to store the solution is such that only the pitches of the original occurrence of the patterns `p()` need to be decided. All other occurrences of a pattern are automatically set based on the pitches for the original pattern and the set of translational equivalence vectors `V()` of the pattern. This setup speeds up the algorithm as it drastically reduces the size of the variables in the set x.

In addition to the pattern constraints, an additional hard constraint is imposed on the pitch range for each track. This range is set based on the lowest and highest occurring pitch in the template piece for each track. Within this range, all possible pitches are allowed.

6 Variable Neighborhood Search

In this section, we describe the variable neighborhood search (VNS) algorithm used to solve the optimization problem defined above. Much of the research on the development of metaheuristics stems from more traditional fields such as vehicle routing and scheduling. In this research, we chose to implement a VNS algorithm because it has been shown to outperform several other heuristics (including genetic algorithms) on a range of problems [75]. Since its inception in the late 90s, it has been successfully applied to problems in combinatorial optimization including project scheduling [76], finding extremal graphs [77], vehicle routing [78], graph coloring [79], and piano fingering [80].

A VNS algorithm has previously been developed for generating counterpoint music [4, 38]. This algorithm has proven to be efficient and outperformed a genetic algorithm implemented on the same problem. The inner workings of the algorithm have been modified to work with complex polyphonic piano music, and the constraints and objective function described in Section 5 have been integrated into the algorithm.

6.1 Local search components

The core of a VNS algorithm is a local search strategy. Local search typically starts from an initial solution x, and iteratively makes a small change (i.e. a move) in order to find a better solution. We refer to the set of solutions x' that can be reached by applying one type of move to a solution as the *neighborhood*. In this case this means that the neighborhood will consist of all solutions that can be reached by applying one type of move to any of the time slices of the piece. A *first descent strategy* was implemented in MorpheuS, whereby the neighborhood is built for one note/time slice at a time. As soon as a (feasible) solution is found that has a better value for the objective function $D(x')$, this solution is accepted as the new current solution x.

An additional strategy for accelerating the search applies the moves chronologically from the start to the end of the piece. When a move is successful, this change will affect the tension profile only in its immediate vicinity. Therefore, the algorithm will backtrack only 4 time slices then resume the search.

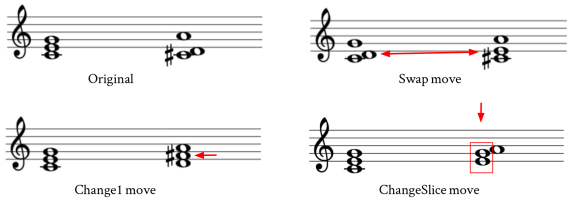


Fig. 4: An example of a potential move using each of the three different types of moves.

Three types of moves are implemented in MorpheuS based on [38]. An example of each type of move is displayed in Figure 4 using a very short fragment. The change1 move changes the pitch of one note to all of the other possible pitches in the allowed range to form the neighborhood. The swap move consists of all musical pieces that can be created by swapping the pitch of any two notes of the current piece. Finally, changeSlice changes the pitches of two randomly chosen notes in a vertical time slice to all of the other allowed pitches in the range. The respective size of each neighborhood generated by these three types of moves is displayed in Table 1. In order to speed up the algorithm, a first descent strategy is implemented, in which the neighborhood is built one move at a time. Whereas a steepest descent strategy would generate the full neighborhood, the first descent strategy accepts a new solution as soon as it improves the value of the current solution (see previous subsection).

TABLE 1: Size of the neighborhood generated by each move type for a piece consisting of n chords, each containing m notes, and with a pitch range of p.

Move type	Neighborhood size
change1	$m \times n \times p$
swap	$((n \times m) - 1)!$
changeSlice	p^2

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAFFC.2017.2737984, IEEE Transactions on Affective Computing

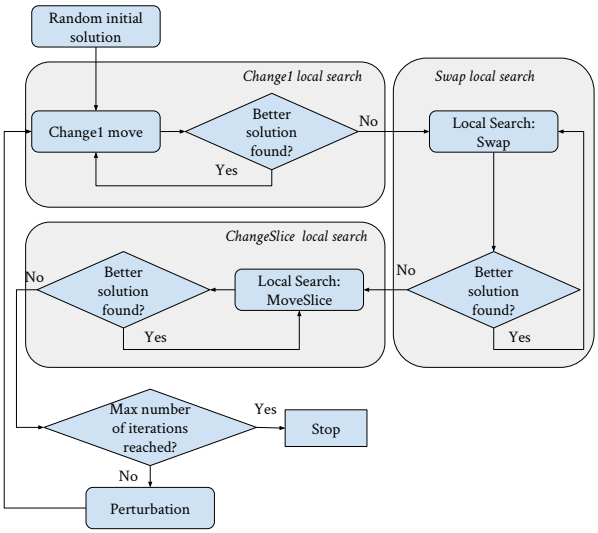


Fig. 5: Flow chart of the variable neighborhood search algorithm.

6.2 Outline of the VNS

A system diagram of the full algorithm implemented in MorpheuS is shown in Figure 5. The VNS starts from a random (feasible) solution, which is built by assigning pitches from the range in a uniformly random manner. This initial solution is set as the current solution x.

The algorithm then performs local search using the change1 neighborhood. When no improving feasible solution can be found, the VNS will then switch the local search to a different neighborhood type (e.g. changeSlice), which then allows the search to continue [81]. This process is repeated until no better solution can be found in any of the neighborhoods, in which case the algorithm is said to have arrived at a local optimum. The VNS algorithm implements a perturbation strategy to escape this local optimum and then continues the search for the global optimum [82]. A perturbation re-assigns a significant proportion of the pitches each to a uniform random (feasible) pitch. Based on the research of Herremans and Sörensen [4], the number of perturbed pitches was set to 12%. In an iterative local search strategy, the algorithm restarts from a totally random solution when a local optimum is reached. The perturbation strategy implemented in the VNS, however, leads to far better results [83]. The search process continues until the stopping criterion (i.e. a maximum allowed number of iterations) is reached. The order in which the different types of moves are applied is based on the increasing computational complexity of calculating the full neighorhood. In the next section, we evaluate the implemented algorithm and its musical results.

7 Results

The MorpheuS system is evaluated on three levels. The first examines the effects of pattern detection on the musical outcome. Next, we consider the efficiency of the optimization algorithm. Last but not least, the generated musical output is evaluated and compared to the original template piece.

7.1 Effect of pattern detection algorithm

The selected pattern detection algorithm (COSIATEC versus SIATECCompress) exerts a big influence on the resulting pieces. Short but frequent patterns can overly constrain the generation process, thus forcing it to converge quickly to the original piece. Infrequently repeated patterns, even though they may be long, easily lose sight of the goal of constraining long term structure. The user can specify which algorithm is implemented: COSIATEC, which uniquely captures each note in precisely one pattern, or SIATECCompress, which captures more relationships between different notes, resulting in overlapping patterns. Each of these algorithm in turn has additional settings, such as maximum and minimum pattern lengths. We have generated three different pattern sets based on an excerpt of Rachmaninov's "Étude Tableau Op. 39, No. 6", shown in Figure 6.

Based on this excerpt, we have calculated three sets of repeated patterns, as displayed in Figure 7. Their main properties, namely, compression ratio, number of notes in patterns, `P()`, and the size of the TECs, are shown in Table 2. Each of these three pattern sets was then used as a structural template during music generation in MorpheuS. The resulting music pieces generated, based on a short run of 10 iters (less then 1 minute generation time on a Dell XPS13 laptop with i7core and 8GB RAM) are displayed in Figure 11 in Appendix A. An example of each detected pattern, together with the set of translational equivalent vectors is shown on each score in green and orange respectively.

The first pattern set (A), shown in Figure 7(a), was detected using COSIATEC with a minimum pattern length of 5. This resulted in 6 TECs with a compression ratio of 1.65, and 69 unique notes that needed to be optimized by MorpheuS (see Table 2). The resulting piece, created by iterating through the VNS 10 times is displayed in Figure 11a. The music retains some of the contours of the original piece, but also contains a great deal of original musical content.

When constraining COSIATEC to detect only very short patterns of maximum length 2, we obtain a set of TECs (B), shown in Figure 7(b). This yields a very similar compression ratio, yet the piece generated based on this template pattern is very different. In this case, the original piece is almost replicated exactly due to the many constraints posed by the set of TECs. The prevalence of such short patterns typically severely limit the originality of the music generated. Here, MorpheuS only has 11 notes to optimize; the others were derived from the translation vectors of the pattern vector.

Pattern set C, shown in Figure 7(c), shows the results of running SIATECCompress without any constraints on pattern length. Although the compression ratio of COSIATEC is often higher than that of SIATECCompress [74], musically speaking, the latter may have further benefits. In SIATECCompress, a note can be contained in multiple sets of TECs, which allows the algorithm to find a different and larger set of TECs than COSIATEC. This may result in the algorithm capturing more meaningful musical relationships. The resulting music generated using pattern set C as a template offers a mix between the highly constrained nature

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAFFC.2017.2737984, IEEE Transactions on Affective Computing

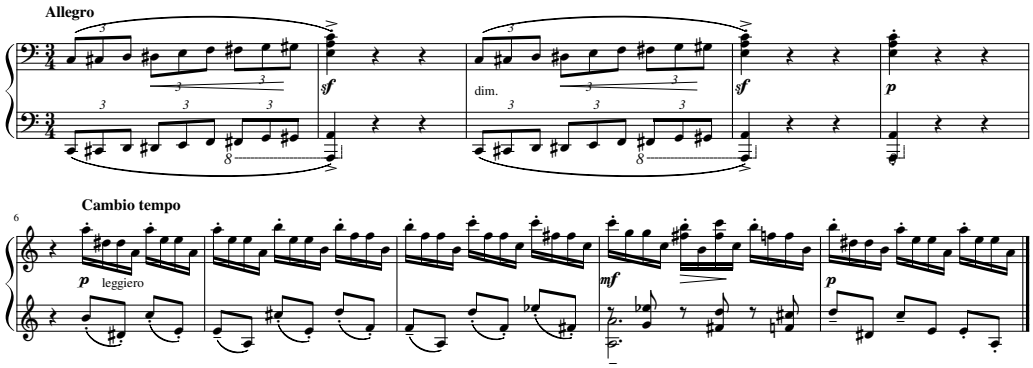
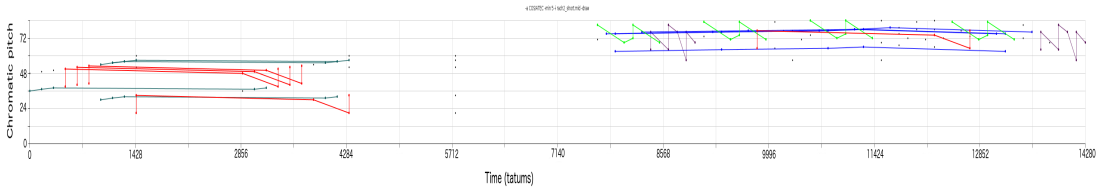
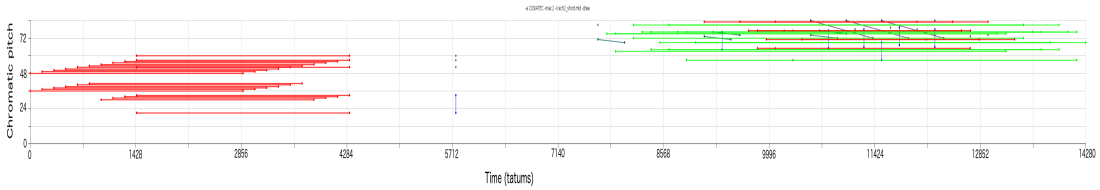


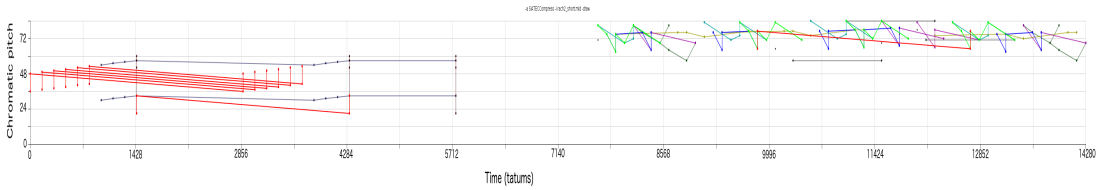
Fig. 6: Original excerpt from Rachmaninov's "Étude Tableau Op. 39, No. 6"



(a) Pattern set A: COSIATEC, minimum pattern length 5, compression ratio: 1.65, Number of TECs: 6



(b) Pattern set B: COSIATEC, maximum pattern length 2, compression ratio: 1.67, number of TECs: 6



(c) Pattern set C: SIATECCompress (no restrictions on the pattern length), compression ratio: 1.58, number of TECs: 11. Each TEC is represented with a different color.

Fig. 7: Different patterns detected in Rachmaninov's "Étude Tableau Op. 39, No. 6"

of pattern set B and the freedom of pattern set A. An example of this can be found in the ascending pattern in bars 1 and 3. In pattern set C, both bars have an ascending line, yet the starting note is different. Pattern set B generates a more constrained output, whereby both bars are identical. With pattern set A, we see a much freer interpretation, whereby the two bars bear minimal resemblance to each other. Although each of the three example patterns offer a way to constrain long-term structure in generated music, the degree to which they constrain pitch patterns has significant effect on the musical outcome.

7.2 Evolution of solution quality

A formal comparison with other existing systems was not possible due to the fact that MorpheuS is the first algorithm that implements a tension based objective function with pattern constraints. The use of VNS for generating counterpoint, on which MorpheuS is based, has been tested extensively in [38] and shown to outperform a genetic algorithm on the same task. We can thus assume that, in the more constrained (due to the patterns imposed) musical task represented here, the algorithm will be at least as effective.

TABLE 2: Pattern sets generated for Rachmaninov's "Étude Tableau Op. 39, No. 6"

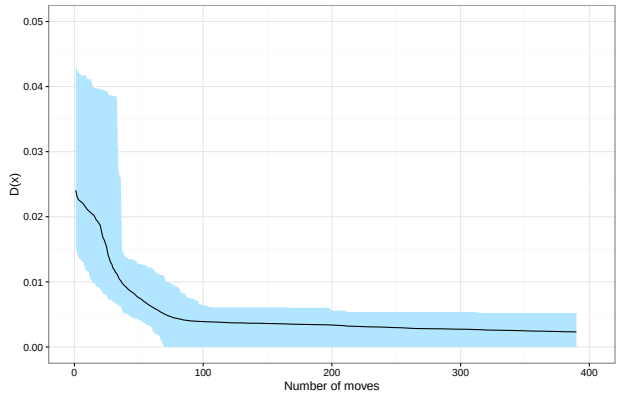
	Algorithm	CR	UP	TECs
Pattern set A	COSIATEC	1.65	69	6
Pattern set B	COSIATEC	1.67	11	6
Pattern set C	SIATECCompress	1.58	34	11

CR: compression ratio, UP: number of pitches to be set by MorpheuS

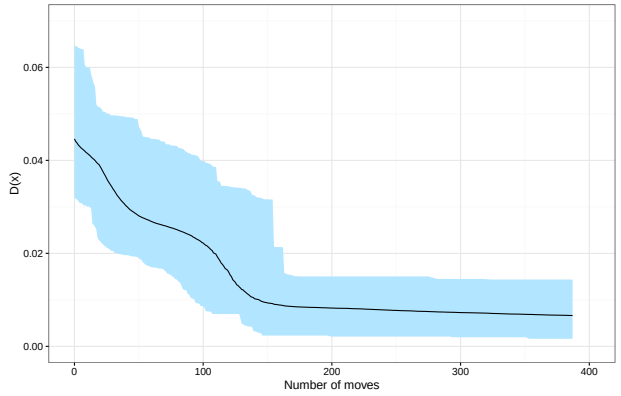
In order to verify the effectiveness of the algorithm

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAFFC.2017.2737984, IEEE Transactions on Affective Computing

it was run 100 time on Kabalevsky's "Clowns" (from *24 Pieces for Children*, Op. 39 No. 20) with SIATECCompress patterns, and the Rachmaninov "Étude Tableau Op. 39, no. 6" shown above (using pattern set C). Figure 8 shows the range and average of the best objective function value found over time for 100 runs of the VNS for both pieces. The experiment was performed on a Dell XPS Ultrabook with i7Core and 8GB RAM. The average running time of the VNS was 136 seconds for "Clowns" and 526 seconds for the Rachmaninov piece. The size of the solution was 34 and 84 notes, respectively.



(a) For Rachmaninov's "Étude Tableau Op. 39, No. 6" (with Pattern set C)



(b) Kabalevsky's "Clowns".

Fig. 8: Evolution of the objective function over time, for 100 runs of the VNS. The plotted line shows the mean best solution found by the VNS for 100 runs. The ribbons show the maximum and minimum objective function values for the best solution found over the 100 runs at each move.

Figures 8a and 8b clearly show a steep improvement during the initial seconds of the algorithm's run for both pieces. This pattern can be observed for each of the 100 runs, as the maximum values for the best solution found (i.e. worst run of the algorithm), goes down quickly. Even after 2 minutes, the algorithm manages to find small improvements to the current solution.

In Figure 9, we isolate one particular run of the algorithm on "Clowns". A clear descending trend can be observed when looking at the best solution found over time. The peaks in the graph indicate points of perturbation. Whenever the search gets trapped in a local optimum, the current solution is perturbed, leading to a temporarily worse solution. Note that even after 500 moves, the perturbation

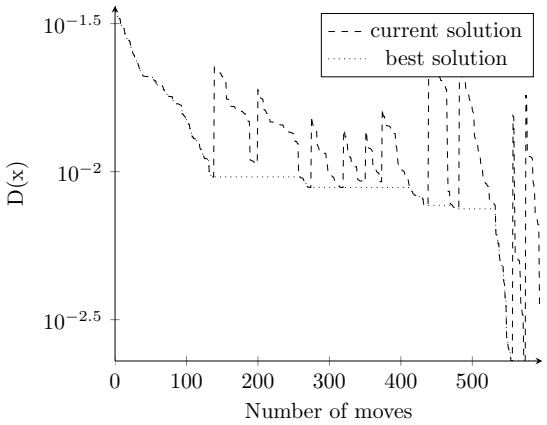


Fig. 9: Evolution of the objective function over time for one run of the VNS, for Kabalevsky's "Clowns".

step manages to escape from a local optimum to find a better solution, thus confirming that the perturbation strategy is successful.

The three tension profiles: cloud diameter, tensile strain and cloud momentum, are shown in Figure 10. The graphs show the original tension profile of the template piece (dashed line), and snapshots of the tension profiles of the output piece before optimization (random piece) and after optimization. It can be noticed that the tension profile of the original piece fluctuates between tension and relaxation, a dynamic that Lerdahl and Jackendoff [84] discuss in their Generative Theory of Tonal Music.

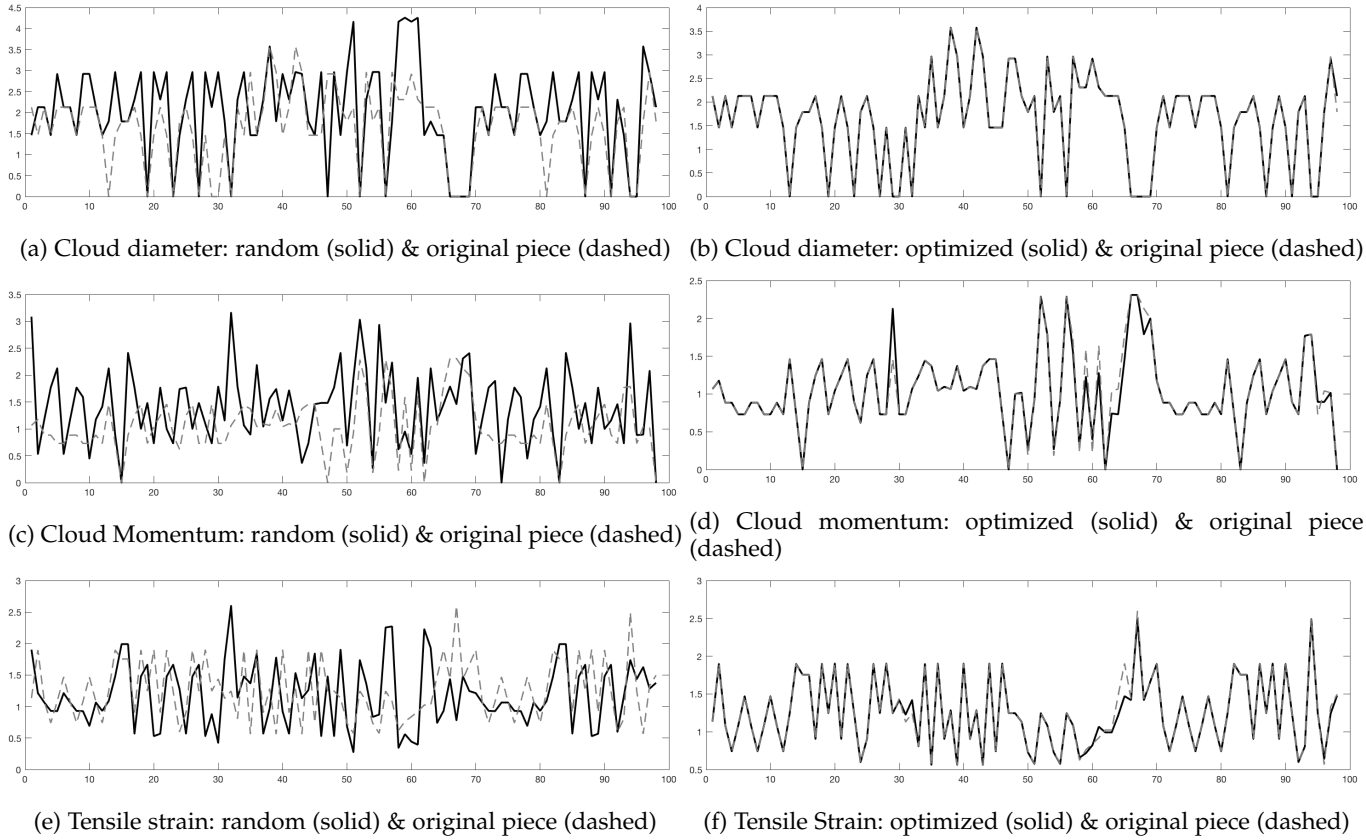
The graphs of the random piece, however, show a much more erratic tension profile distant from that of the original piece. Overall, the tension is also higher for the random pieces, which can in part be explained by the dissonance that we can expect in pieces with random pitches. A striking similarity can be seen between the tension profiles of the original piece (template) and the optimized piece, yet again confirming that the optimization algorithm indeed finds a solution that minimizes the objective function. This is confirmed by the correlation coefficients, which are high between the tension profiles of the optimized and template piece (0.9748, 0.9918, 0.9993), and lower between the random initial solution and the template piece (0.4103, 0.2053, and 0.6174). In the next section we will focus on the actual musical results.

7.3 Musical outcome

It must be noted that in the examples given in this paper, the goal of the optimization is to fit the original tension profile of the *template* piece as closely as possible. This explains a tendency to revert to the original piece, but offers us a way to verify that the optimization algorithm performs well. A future version of MorpheuS may include a 'similarity-penalty' in the objective function, to enforce originality in the generated pieces. Currently, the user is free to design their own tension profile to create an original piece, or use the tension profile of a template piece.

Appendix B (Figure 12) shows the musical output of the optimization process described in the previous section together with the initial (random) starting piece, based on

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAFFC.2017.2737984, IEEE Transactions on Affective Computing



(a) Cloud diameter: random (solid) & original piece (dashed)

(b) Cloud diameter: optimized (solid) & original piece (dashed)

(c) Cloud Momentum: random (solid) & original piece (dashed)

(d) Cloud momentum: optimized (solid) & original piece (dashed)

(e) Tensile strain: random (solid) & original piece (dashed)

(f) Tensile Strain: optimized (solid) & original piece (dashed)

Fig. 10: The three types of tension profiles before and after optimization of Kabalevsky's "Clowns", together with the template profile (dashed). The y-axes represent the value of each tension score (cloud diameter, tensile strain and cloud momentum), and the x-axes represent score time.

the template piece "Clowns" by Kabalevsky. A significant improvement in musical quality can be noticed between the random and the optimized piece. One of the most evident aspects is that the latter (optimized) piece is much more tonal. We can also see some long-term recurrent structures; for example, the theme from bars 1-5 returns in bars 18-22.

The reader is invited to listen to this and other output generated by MorpheuS online[5] While these first tests are promising, there is still room for improvement. One interesting improvement is to add constraints on playability. While the current pitch range constraint ensures that assigned notes occur within the range of the template piece, the output can be far from idiomatic for the instrument and variables such as the unexpectedness of the note sequences can make them difficult to play. Improved playability could be achieved by a statistical machine learning approach (e.g. Markov model or recurrent neural network) to integrate transition probabilities in the current objective function.

Following live performances of MorpheuS' pieces, we have received a range of comments from expert musicians, reflecting interesting perspectives on MorpheuS' compositions to inform future developments of the system. Upon hearing MorpheuS' version of a Haydn sonata movement, an academic composer remarked that the system shows some competence with the repetition of material, however it does not develop the material; furthermore, it does not know (like Stravinsky) to use the 'right' kind of 'wrong' notes; MorpheuS' use (or misuse) of cadences, cadential figures were inserted in odd places, were the most obvious anomalies distinct from human composition; also, the phrase structure, the evolution of harmony with respect to phrase structure, does not work as the tension levels do not relate to the phrase structure.

Several expert listeners remarked on the humor apparent in MorpheuS' pieces, particularly the ways in which it naively violates conventional expectations, often with surprising and sometimes adventurous but plausible results. The advantage of this naiveté, as one person puts it, is that the system "has no fear" and thus "has courage to try things", unencumbered by conditioning that constrains human composers to behave or make choices only in ways that are deemed acceptable. This was in the context of three morphed pieces by Bach and three morphed pieces by Kabalevsky. In a few of these pieces, there were awkward harmonic moments when returning to the beginning of a repeated section, leading one expert listener to comment that MorpheuS lacked the ability to make good transitions. However, the listener found it fascinating to hear the original Bach pieces (from "A Little Notebook for Anna Magdalena") through the lens of MorpheuS' compositions. In contrast, another expert listener found the Kabalevsky pieces "more honest" than the Bach ones, likely because the original Bach pieces were too recognizable in the morphed ones, yet lacked certain characteristics typically associated with the pieces.

5. http://dorienherremans.com/morpheus

Finally, with regard to interaction design, it would be interesting to test a transformational approach in which the optimization starts from the original piece and searches for a new piece that matches a tension profile provided by the user, thus transforming the original piece.

8 Conclusions

MorpheuS consists of a novel framework that allows us to tackle one of the main challenges in the field of automatic music generation: long-term structure. An efficient variable neighborhood search algorithm was developed that enables the generation of complex polyphonic music, with recurring patterns, according to a tension profile. Two pattern detection techniques extract repeated and translated note patterns from a template piece. This structure is then used as scaffolding to constrain long-term structure in new pieces. The objective function of the optimization problem is based on a mathematical model for tonal tension. This allows for the generation of pieces with a predefined tension profile, which has potential applications to game or film music. The pieces generated by MorpheuS have proved to be promising and have been tested in live performance scenarios at concerts internationally.

In future research it would be interesting to explore the integration of machine learning methods, e.g. deep learning techniques or Markov models, in the objective function of the optimization algorithm. This way, a style could be learned from a large corpus of music, and in turn be reflected in the generated piece. We expect that this would also improve the playability of the pieces and reduce awkward transitions. A second expansion would be to allow for more flexible pattern detection, such as recognition of inverted patterns, augmentations, and diminutions and other variations. It would equally be interesting to evaluate if the generated music elicits the same emotion responses as expected given a tension profile, by measuring physiological responses or by recording listener judgments of tension as described in [85, 86]. The tension model could also be expanded to capture other characteristics of tension such as timbre and cadence. Finally, in the context of adaptive game music generation, the VNS algorithm could be modified to allow for real-time generation, much like the system Herremans and Sorensen [87] implemented as the FuX 2.0 mobile music generation app.

Acknowledgments

This project has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 658914. We would also like to thank Prof. Dr. David Meredith for providing us with implementations of his COSIATEC and SIATECCompress algorithms.

Dorien Herremans is an Assistant Professor at the Information Systems Technology and Design Pillar at the Singapore University of Technology and Design. Before that, she was a Marie Skłodowska-Curie Fellow at the Centre for Digital Music at Queen Mary University of London. Prof. Herremans is currently working on the project: "MorpheuS: Hybrid Machine Learning Optimization techniques To Generate Structured Music Through Morphing And Fusion". She received her Ph.D. on the topic of Computer Generation and Classification of Music through Operations Research Methods. She graduated as a business engineer in management information systems at the University of Antwerp in 2005. After that, she worked as a Drupal consultant and was an IT lecturer at the Les Roches University in Bluche, Switzerland. She also worked as a mandaatassistent at the University of Antwerp, in the domain of operations managment, supply chain management and operations research. Dr. Herremans' research focuses the intersection of machine learning/optimization and music. Prof. Herremans is a member of IEEE society and co-organizer of the First International Workshop on Deep Learning and Music as part of IJCNN, and in the organizing committee of ORBEL26 (Conference of the Belgian Operations Research Society).

Elaine Chew is Professor of Digital Media in the School of Electronic Engineering and Computer Science at Queen Mary University of London (QMUL) where she is affiliated with the Centre for Digital Music. Prior to joining QMUL in 2011, she was a tenured associate professor at the University of Southern California, where she was the inaugural holder of the Viterbi Early Career Chair. She was a recipient of the US Presidential Early Career Award in Science and Engineering and NSF CAREER Award, and the Edward, Frances, and Shirley B. Daniels Fellow at the Radcliffe Institute for Advanced Study. She is also an alum of the NAS Kavli Frontiers of Science and NAE Frontiers of Engineering Symposia. Her research centers on the mathematical and computational modeling of music structure, musical prosody, music cognition, and ensemble interaction. She is author of over 100 peer-reviewed chapters and articles, and author and editor of 8 books and journal special issues on music and computing. She has served as program and general chair of the International Conferences on Music Information Retrieval (2008) and of Mathematics and Computation in Music (2009, 2015), and was invited convenor of the Mathemusical Conversations international workshop in 2015. She was awarded PhD and SM degrees in operations research at the Massachusetts Institute of Technology, and a BAS in mathematical and computational sciences (hon) and music (distinction) at Stanford University.

REFERENCES

[1] McKinsey&Company, "Global media report 2015: Global industry overview," 2015.

[2] D. Herremans and E. Chew, "Morpheus: Automatic music generation with recurrent pattern constraints and tension profiles," in *Proceedings of IEEE TENCON*, Singapore, November 2016.

[3] F. Pachet and P. Roy, "Markov constraints: steerable generation of markov sequences," *Constraints*, vol. 16, no. 2, pp. 148–172, 2011.

[4] D. Herremans and K. Sörensen, "Composing fifth species counterpoint music with a variable neighborhood search algorithm," *Expert Systems with Applications*, vol. 40, no. 16, pp. 6427–6437, 2013.

[5] 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.

[6] A. J. Cohen, "Music as a source of emotion in film," *Music and emotion: Theory and research*, pp. 249–272, 2001.

[7] D. Herremans and E. Chew, "Tension ribbons: Quantifying and visualising tonal tension," in *Proceedings of the International Conference on Technologies for Music Notation and Representation - TENOR2016*, Cambridge, UK, 2016, pp. 8–18.

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

[9] J. D. Fernández and F. Vico, "Ai methods in algorithmic composition: A comprehensive survey," *Journal of Artificial Intelligence Research*, pp. 513–582, 2013.

[10] A. Lovelace, "'notes on l. menabrea's 'sketch of the analytical engine invented by charles babbage, esq.' '," *Taylor's Scientific Memoirs*, vol. 3, 1843.

[11] A. Hiller Jr, Lejaren and L. M. Isaacson, "Musical composition with a high speed digital computer," in *Audio Engineering Society Convention 9*. Audio Engineering Society, 1957.

[12] R. C. Pinkerton, "Information theory and melody," *Scientific American*, vol. 194, no. 2, pp. 77–86, 1956.

[13] D. Conklin and I. H. Witten, "Multiple viewpoint systems for music prediction," *Journal of New Music Research*, vol. 24, no. 1, pp. 51–73, 1995.

[14] F. Pachet and P. Roy, "Musical harmonization with constraints: A survey," *Constraints*, vol. 6, no. 1, pp. 7–19, 2001.

[15] C.-H. Chuan and E. Chew, "Generating and evaluating musical harmonizations that emulate style," *Computer Music Journal*, vol. 35, no. 4, pp. 64–82, 2011.

[16] S. Dubnov and G. Assayag, "Music design with audio oracle using information rate," in *Musical Metacreation: Papers from the 2012 AIIDE Workshop*, 2012.

[17] G. Assayag, G. Bloch, and M. Chemillier, "Omax-ofon," *Sound and Music Computing (SMC)*, vol. 2006, 2006.

[18] A. R. François, I. Schankler, and E. Chew, "Mimi4x: An interactive audio-visual installation for high-level structural improvisation," *International Journal of Arts and Technology*, vol. 6, no. 2, pp. 138–151, 2013.

[19] D. Eck and J. Schmidhuber, "A first look at music composition using lstm recurrent neural networks," *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale*, vol. 103, 2002.

[20] C.-C. Chen and R. Miikkulainen, "Creating melodies with evolving recurrent neural networks," in *Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on*, vol. 3. IEEE, 2001, pp. 2241–2246.

[21] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription," in *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, J. Langford and J. Pineau, Eds. New York, NY, USA: ACM, 2012, pp. 1159–1166.

[22] D. Herremans and C.-H. Chuan, "Modeling musical context using word2vec," in *First International Workshop on Deep Learning and Music*, Anchorage, US, 2017, pp. 11–18.

[23] N. Cunha, A. Subramanian, and D. Herremans, "Uma abordagem baseada em programação linear inteira para a geração de solos de guitarra," Vitória, Brasil, 09/2016 2016.

[24] 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.

[25] K. Sörensen, "Metaheuristics–the metaphor exposed," *International Transactions in Operational Research*, vol. 22, no. 1, pp. 3–18, 2015.

[26] 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.

[27] K. Sörensen and F. Glover, "Metaheuristics," in *Encyclopedia of Operations Research and Management Science*, 3rd ed. S. I. Gass and M. C. Fu, eds., Springer, New York, 2012.

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

[29] J. Biles, "Autonomous genjam: eliminating the fitness bottleneck by eliminating fitness," in *Proceedings of the GECCO-2001 Workshop on Non-routine Design with Evolutionary Systems*. San Francisco, California, USA: Morgan Kaufmann, 7 2001.

[30] R. McIntyre, "Bach in a box: The evolution of four part baroque harmony using the genetic algorithm," in *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence*. IEEE, 1994, pp. 852–857.

[31] J. Polito, J. Daida, and T. Bersano-Begey, "Musica ex machina: Composing 16th-century counterpoint with genetic programming and symbiosis." in *Evolutionary Programming VI*, ser. Lecture Notes in Computer Science, vol. 1213. Springer, 1997, pp. 113–124.

[32] S. Phon-Amnuaisuk, A. Tuson, and G. Wiggins, "Evolving musical harmonisation," in *Artificial neural nets and genetic algorithms: proceedings of the international conference in Portorož, Slovenia, 1999*. Springer Verlag Wien, 1999, p. 229.

[33] N. Tokui and H. Iba, "Music composition with interactive evolutionary computation," in *Proceedings of the Third International Conference on Generative Art*, vol. 17:2, 2000, pp. 215–226.

[34] D. Horowitz, "Generating rhythms with genetic algorithms," in *Proceedings of the International Computer Music Conference*. San Francisco: International Computer Music Association, 1994, pp. 142–143.

[35] G. Carpentier, G. Assayag, and E. Saint-James, "Solving the musical orchestration problem using multiobjective constrained optimization with a genetic local search approach," *Journal of Heuristics*, vol. 16, no. 5, pp. 1–34, 2010.

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

[37] S. Davismoon and J. Eccles, "Combining musical constraints with Markov transition probabilities to improve the generation of creative musical structures," in *Applications of Evolutionary Computation*. Springer, 2010, pp. 361–370.

[38] 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.

[39] R. Parke, E. Chew, and C. Kyriakakis, "Quantitative and visual analysis of the impact of music on perceived emotion of film," *Computers in Entertainment (CIE)*, vol. 5, no. 3, p. 5, 2007.

[40] A. Prechtl, R. Laney, A. Willis, and R. Samuels, "Methodological approaches to the evaluation of game music systems," in *Proceedings of the 9th Audio Mostly: A Conference on Interaction With Sound*. ACM, 2014, p. 26.

[41] J.-I. Nakamura, T. Kaku, K. Hyun, T. Noma, and S. Yoshida, "Automatic background music generation based on actors' mood and motions," *The Journal of Visualization and Computer Animation*, vol. 5, no. 4, pp. 247–264, 1994.

[42] K. Collins, *Game sound: an introduction to the history, theory, and practice of video game music and sound design*. Mit Press, 2008.

[43] M. Müller and J. Driedger, "Data-driven sound track generation." in *Multimodal Music Processing*, 2012, pp. 175–194.

[44] K. Collins, "An introduction to procedural music in video games," *Contemporary Music Review*, vol. 28, no. 1, pp. 5–15, 2009.

[45] A. Prechtl, "Adaptive music generation for computer games," Ph.D. dissertation, The Open University, 2016.

[46] P. Casella and A. Paiva, "Magenta: An architecture for real time automatic composition of background music," in *Intelligent Virtual Agents*. Springer, 2001, pp. 224–232.

[47] D. Brown, "Mezzo: An adaptive, real-time composition program for game soundtracks," in *Proceedings of the AIIDE Workshop on Musical Metacreativity*, 2012.

[48] M. Farbood, "A parametric, temporal model of musical tension," *Music Perception*, vol. 29, no. 4, pp. 387–428, 2012.

[49] J. A. Sloboda, "Music structure and emotional response: Some empirical findings," *Psychology of music*, vol. 19, no. 2, pp. 110–120, 1991.

[50] C. L. Krumhansl and D. L. Schenck, "Can dance reflect the struc-

tural and expressive qualities of music? a perceptual experiment on balanchine's choreography of mozart's divertimento no. 15," *Musicae Scientiae*, vol. 1, no. 1, pp. 63–85, 1997.
[51] K. R. Scherer and M. R. Zentner, "Emotional effects of music: Production rules," *Music and emotion: Theory and research*, pp. 361–392, 2001.
[52] N. Steinbeis, S. Koelsch, and J. A. Sloboda, "The role of harmonic expectancy violations in musical emotions: Evidence from subjective, physiological, and neural responses," *Journal of cognitive neuroscience*, vol. 18, no. 8, pp. 1380–1393, 2006.
[53] F. Lerdahl and C. L. Krumhansl, "Modeling tonal tension," *Music Perception: An Interdisciplinary Journal*, 2007.
[54] M. M. Farbood, "A quantitative, parametric model of musical tension," Ph.D. dissertation, Massachusetts Institute of Technology, 2006.
[55] M. Farbood, H. Kaufman, and K. Jennings, "Composing with hyperscore: An intuitive interface for visualizing musical structure," in *Proceedings of the International Computer Music Conference (ICMC)*, vol. 59, 2007.
[56] T. Browne and C. Fox, "Global expectation-violation as fitness function in evolutionary composition," in *Applications of Evolutionary Computing*. Springer, 2009, pp. 538–546.
[57] J. Rutherford and G. Wiggins, "An experiment in the automatic creation of music which has specific emotional content," in *Proc. for the 7th International Conference on music Perception and Cognition*, 2002.
[58] C. Roig, L. J. Tardón, I. Barbancho, and A. M. Barbancho, "Automatic melody composition based on a probabilistic model of music style and harmonic rules," *Knowledge-Based Systems*, vol. 71, pp. 419–434, 2014.
[59] P. Todd, "A connectionist approach to algorithmic composition," *Computer Music Journal*, pp. 27–43, 1989.
[60] E. Chew, *Mathematical and Computational Modeling of Tonality: Theory and Applications*. New York: Springer, 2014, vol. 204.
[61] B. Magee, *The Tristan Chord: Wagner and Philosophy*. Macmillan, 2002.
[62] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: A review," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 1, pp. 4–37, 2000.
[63] S. L. Sclove, "Application of the conditional population-mixture model to image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, no. 4, pp. 428–433, 1983.
[64] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 221–231, 2013.
[65] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 12, pp. 2037–2041, 2006.
[66] T. R. Gingeras, G. Ghandour, E. Wang, A. Berno, P. M. Small, F. Drobniewski, D. Alland, E. Desmond, M. Holodniy, and J. Drenkow, "Simultaneous genotyping and species identification using hybridization pattern recognition analysis of generic mycobacterium dna arrays," *Genome research*, vol. 8, no. 5, pp. 435–448, 1998.
[67] F. Itakura, "Minimum prediction residual principle applied to speech recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 23, no. 1, pp. 67–72, 1975.
[68] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on speech and audio processing*, vol. 10, no. 5, pp. 293–302, 2002.
[69] R. Picard, *Affective computing*. MIT press Cambridge, 1997, vol. 252.
[70] R. B. Dannenberg and N. Hu, "Pattern discovery techniques for music audio," *Journal of New Music Research*, vol. 32, no. 2, pp. 153–163, 2003.
[71] J.-J. Aucouturier, B. Defreville, and F. Pachet, "The bag-of-frames approach to audio pattern recognition: A sufficient model for urban soundscapes but not for polyphonic music," *The Journal of the Acoustical Society of America*, vol. 122, no. 2, pp. 881–891, 2007.
[72] D. Meredith, G. A. Wiggins, and K. Lemström, "Method of pattern discovery," *PCT patent application number PCT/GB02/02430, UK patent application*, no. 0211914.7, 2002.
[73] J. A. Bilmes, "Timing is of the essence: Perceptual and computational techniques for representing, learning, and reproducing expressive timing in percussive rhythm," Ph.D. dissertation, Massachusetts Institute of Technology, 1993.
[74] D. Meredith, "Music analysis and point-set compression," *Journal of New Music Research*, vol. 44, no. 3, pp. 245–270, 2015.
[75] P. Hansen, N. Mladenović, and D. Perez-Britos, "Variable neighborhood decomposition search," *Journal of Heuristics*, vol. 7, no. 4, pp. 335–350, 2001.
[76] 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.
[77] G. Caporossi and P. Hansen, "Variable neighborhood search for extremal graphs: 1 the autographix system," *Discrete Mathematics*, vol. 212, no. 1, pp. 29–44, 2000.
[78] 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.
[79] C. Avanthay, A. Hertz, and N. Zufferey, "A variable neighborhood search for graph coloring," *European Journal of Operational Research*, vol. 151, no. 2, pp. 379–388, 2003.
[80] M. Balliauw, D. Herremans, D. P. Cuervo, and K. Sörensen, "A variable neighbourhood search algorithm to generate piano fingerings for polyphonic sheet music," *International Transactions Of Operations Research, Special Issue on Variable Neighbourhood Search*, vol. in press, 2015.
[81] N. Mladenovic and P. Hansen, "Variable neighborhood search," *Computers & Operations Research*, vol. 24, no. 11, pp. 1097–1100, 1997.
[82] P. Hansen and N. Mladenović, "Variable neighborhood search," *Handbook of metaheuristics*, pp. 145–184, 2003.
[83] M. O. Lourenço, H. and T. Stützle, "Iterated local search," *Handbook of metaheuristics*, pp. 320–353, 2003.
[84] F. Lerdahl and R. Jackendoff, *A generative theory of tonal music*. MIT Press, 1987.
[85] J. Kim and E. André, "Emotion recognition based on physiological changes in music listening," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 12, pp. 2067–2083, 2008.
[86] K. Agres, J. Forth, and G. A. Wiggins, "Evaluation of musical creativity and musical metacreation systems," *Computers and Entertainment*, vol. 14, no. 3, pp. 3:1–3:33, Jan. 2017.
[87] D. Herremans and K. Sorensen, "FuX, an android app that generates counterpoint," in *Computational Intelligence for Creativity and Affective Computing (CICAC), 2013 IEEE Symposium on*, April 2013, pp. 48–55.