Music Harmony Transfer

Stephanie Lew
Dept. of Computer Science, National University of Singapore
Email: lewyh@comp.nus.edu.sg

Abstract

Automated music generation has attracted a lot of interest in the past decade. Despite current efforts, it is still very difficult for algorithms to produce convincing results. The fundamental challenge lies in the complex nature of music, which includes pitch relationship, phrase structure, section structure and creativity. Music style transfer was proposed as a method to overcome these difficulties. This QE paper proposes to work on a sub-task of music style transfer, specifically the transfer of harmony from a reference music piece to a target music piece. In other words, the problem is to transfer the reference chords to the target music. Since the pitches and chords are discrete items, this problem is a discrete matching problem. As a first attempt, a greedy matching algorithm is proposed to solve the problem. Test results show that the algorithm is able to match the transferred chords to the target melody and at the same time retain the style of the reference chords.

Supplementary Materials

Links to the example music discussed in this paper and the music generated by this research are available in the web page `https://www.comp.nus.edu.sg/~slewyh/QE-music-examples.html`

1 Introduction

Automated music generation has attracted a lot of interest in the past decade. A variety of methods have been developed to generate music automatically. For example, rules [1, 2], grammars [3, 4] and Markov models [5, 6] are used to generate accompaniments, whereas integer programming is used to incorporate constraints for the accompaniment in the music structure [7]. Genetic algorithms are used to generate variations of melody and rhythm [8–10]. More recently, deep neural networks are trained to generate music in the style of Bach [11, 12]. More details of automated music generation are available in [13, 14].

Despite current efforts, it is still very difficult for algorithms to produce convincing results. The fundamental challenge lies in the complex nature of music [15]. In general, music is a sequence or pattern of notes that follow three kinds of structures:

short-term structure, medium-term and long-term structure. **Short-term structure** refers to the **pitch or frequency relationship** between neighbouring notes. In a piece of well crafted music, for example *Twinkle, Twinkle, Little Star*, the pitch difference between neighbouring notes has to follow a certain pattern for humans to perceive it as musically pleasing. Note sequences with pitch differences that are too small, too large or random[1] are considered as lacking short-term structure or pitch relationship. Algorithmically generated music pieces based on fractals and neural networks often lack short-term structures and sound random[1]. **Medium-term structure** refers to a **phrase** or motif of a melody. A music phrase is a sequence of notes that begin and end with notes or silence that are analogous to pauses in English sentence. Music pieces with good medium-term structure contains consistent phrases that transition smoothly between them. When the music lacks medium-term structure, the phrases are inconsistent and phrase transitions occur randomly[1]. **Long-term structure** refers to distinct **sections** that mark the introduction, transition and ending of a piece of music. A well crafted long-term structure has sections which contain reusable, self-contained music phrases. Music pieces that lack long-term structures may have phrase combinations but sound like existing music phrases that are randomly cut and stitched together[1]. For example, simply repeating the first two bars of *Twinkle, Twinkle, Little Star* three times results in a piece of music that has proper phrases but sounds incomplete[1].

In addition to structure, a well-composed music exhibits **creativity**. A creative piece of music is composed of original, unique musical patterns that distinguish it from other music pieces. The patterns are often repeated with subtle variations that bring out the recurring melody, and yet provide different flavours to the music. Consider, for example, the first movement of Beethoven's *Moonlight Sonata* and Debussy's *Clair de Lune* ("moonlight" in French). Both are slow, soothing music pieces that carry the title of "moonlight", but they differ a lot in style. *Moonlight Sonata* is quiet and somber, and it feels a little sad and weighty. On the other hand, *Clair de Lune* feels serene, bright and light-weight. In contrast, *Twinkle, Twinkle, Little Star* feels simple and much less creative in composition.

Musical creativity is also expressed in terms of the complexity and sophistication of the musical patterns. For example, Chopin's *Fantasie-Impromptu* has a freestyle improvised sound, as though it is played spontaneously. It is a challenging piece to play because it is a fast piece and the right hand has to play at twice the speed of the left hand. Other uniquely creative piano pieces that are very difficult to play include Chopin's *Etude* (study piece) opus 10 no. 4, which requires rapid runs of both hands and frequent switching of melody between the two hands; Rimsky-Korsakov's *Flight of the Bumble Bee*, which mimics the flight and the sound of a busy bumblebee; and Liszt's *Étude* no. 3 *Un Sospiro*, which requires frequent crossing of two hands to produce continuous flow of notes over a wide range. In summary, creativity in music is exhibited through the composition of unique, complex and sophisticated musical phrases and sections that distinguish a piece of music from the others. To date, most music generation systems are still unable to exhibit creativity in generating music [13].

[1]Refer to the web page `https://www.comp.nus.edu.sg/~slewyh/QE-music-examples.html` for examples.

Dai et al. [16] propose to overcome these difficulties using the concept of **music style transfer**. They define music style transfer as altering the style of a music while preserving its content. Due to the multi-level nature of music, content and style at different levels refer to different aspects of music. At the lowest level of sound, content refers to the performance control of music such as loudness and speed, and style refers to timbre, which is the perceptual quality produced by various musical instruments. This kind of style transfer is called timbre style transfer. At the middle level of performance, content refers to the elements of a music score and style refers to performance control, giving rise to performance style transfer. At the highest level of music score, content refers to melody and style refers to other score elements. This is called **composition style transfer**.

The definition of music style transfer proposed by Dai et al. [16] has some weaknesses. Firstly, it is too simple and general because it does not state the source of the style information. For example, a method that alters the style by random assignment of musical notes fits the definition even though the style is not transferred from any source. Secondly, its definition is incomplete. At the timbre level, the music content should include not only performance control but also score elements such as the melody and chords.

This QE paper refines the definition of music style transfer as the synthesis of the style of a target music based on the style of a reference music while preserving the content of the target music. The synthesized style is derived specifically from the reference style and retains the stylistic qualities of the reference style. In contrast to the definition of Dai et al. [16], the refined definition clearly states the source of style information and the required properties of style synthesis.

In music composition, there are three general ways of creating different styles for a piece of music [17, 18], namely harmonization, arrangement, and variation. Harmonization creates the chords and chord progression to generate different harmony between chords and melody. Arrangement may include harmonization, paraphrasing of melody, and adding or changing music materials for the introduction, transitions and ending of music. Variation may change several aspects of music including melody, rhythm, harmony, timbre, etc. When applied to music style transfer, they should keep the melody of a target music unchanged.

This QE paper focuses on a sub-task of composition style transfer, specifically the transfer of harmony. Given a target music score, the computational problem is to generate the target's harmony based on the harmony of a reference music score while preserving the target's melody. At present, this paper does not consider multiple reference music pieces. The primary reason is that the composition style of every piece of music is unique, especially for western classical music. For example, the composition styles of Chopin's *Marzuka in A-flat major* (opus 7 no. 4) and Chopin's *Marzuka in F minor* (opus 24 no. 4) are very different even though both pieces are of the same genre (mazurkas) composed by the same composer (Chopin). Even the four marzukas in the same opus, e.g., Chopin's opus 7, have subtle differences in the chords that accompany the melodies. For example, opus 7 no. 4 has a prolonged D-flat base note in the chord, which is absent in opus 7 no. 1 to no. 3. On the other hand, opus 7 no. 3 ends with an unusually long bright chord in a harmonically dark music setting. Any attempt at combining the composition styles of multiple music

pieces very likely result in a kind of averaged style that dilutes the uniqueness of a specific composition style [19].

There are existing methods that are similar to but different from music style transfer [20–25]. These methods synthesize the style of the target music based on the properties of a desired style. These properties can be either hand-crafted or learned from training examples. The synthesized style is derived from the properties but not from one specific training example. For example, in [20], the newly synthesized style is determined by the timbre and performance control properties learned from a set of training examples. The synthesized style is a kind of averaged style of the training examples, which dilutes the uniqueness of the styles of the training examples. To differentiate these methods from music style transfer, they are grouped into a separate category called **music style generation**.

2 Overview of Music Structures

Music score represents the structure of music by encoding a set of musical elements in visual form (Fig. 1). Its fundamental element is a **note**, which has three main attributes: pitch, duration and onset. **Pitch** is the perceptual property that relates to the frequency of sound. It is denoted by the vertical position of a note. The higher the position, the higher is the pitch. **Duration** is denoted by different types of note symbols (Fig. 2). The notes with longer durations are multiples of those with shorter durations. **Onset** is the starting time of a note, and it is denoted by the horizontal position of the note, where time goes from left to right. When no note is present, a **rest** is indicated. Like notes, rests have varying durations (Fig. 3).

The horizontal sequence of notes is divided into segments called **bars** by vertical bar lines (Fig. 1). A bar's relative duration is encoded by the **time signature**, which consists of two numbers such as 2/4, 3/4, 4/4 and 6/8. The bottom number relates to the duration of a time unit called a **beat** and the top number denotes the number of beats in a bar. For example, 2/4 means that there are two beats in a bar, and the relative duration of a beat is a quarter note or crotchet. The actual duration of a bar is encoded by the **tempo**, which indicates the number of beats per minute. The notes that are held across bars are marked with a **tie**.

Pitch is related to frequency. For a pitch p that is double in frequency of another pitch q, p is said to be an octave above q. In western music, there are 12 distinct pitches in an octave, and the smallest pitch difference is called a **semitone**. For example, E and F are a semitone apart. A pitch can be raised or lowered by a semitone using the sharp symbol ♯ or the flat symbol ♭, respectively (Fig. 2). For example, C♯ is a semitone above C, and B♭ is a semitone below B. The complete set of 12 pitches in an octave consists of C (B♯), C♯ (D♭), D, D♯ (E♭), E (F♭), F (E♯), F♯ (G♭), G, G♯ (A♭), A, A♯ (B♭) and B (C♭). A set of common pitches that form the basis of a music piece is denoted as the **key** of a piece. Frequently occurring pitches modified by sharp or flat symbols are denoted as a set of sharp or flat symbols on the music score, called a **key signature**.

A sequence of notes forms a **melody**. It is usually perceived by the listener as a distinct entity because of its repeatable pattern of duration called **rhythm** and unique sequence of pitches. A melody consists of one or more shorter sequence of

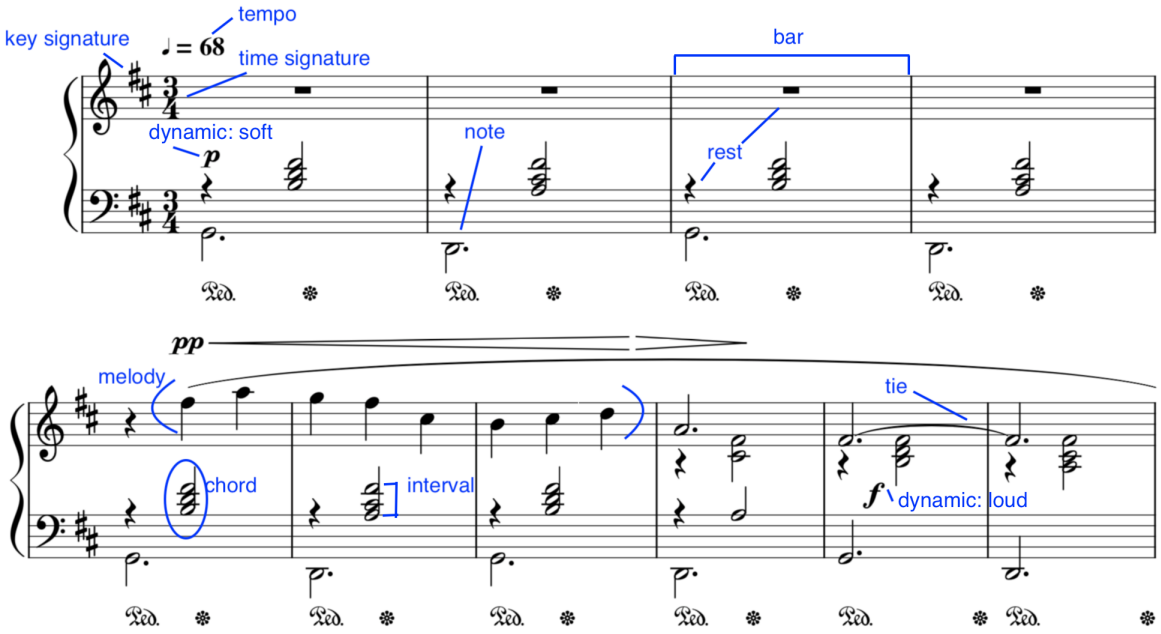


Figure 1: Excerpt from Gymnopedie No. 1 by Erik Satie.

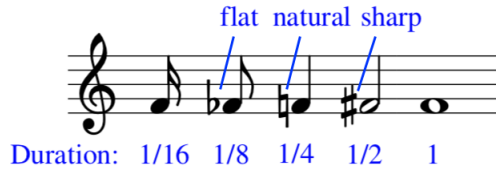


Figure 2: Notes with accidentals and varying duration values.

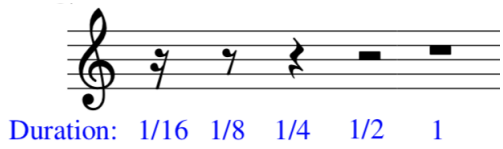


Figure 3: Rests with varying duration values.

notes called **phrase** or **motifs**. It can be described by the rising and falling of individual notes which forms a **melodic contour**.

A set of 2 or more notes played concurrently forms a **chord**. These notes can also be sequentially played in rising and descending order. The notes of a chord sounding together forms **harmony**. A set of chords in motion forms a **chord progression** which can interact with the melody to form complex harmonies. Chords can also be identified by the pitch difference between its notes. The pitch difference between two notes forms an **interval**, which is measured in semitones. Certain intervals in chords cause unpleasant clashes in sound called **dissonance**, which creates the perception of tension in music. Tension caused by the clash between notes in the melody and chords is called **harmonic tension**.

Apart from chords sounding together, a group of melodies can also interact to

form harmonies. This arrangement is known as **counterpoint**. In counterpoint, the melodies tend to be harmonically interdependent and yet independent in rhythm and contour. The melodies in counterpoint music can be restricted by rules of increasing complexity. This type of counterpoint is known as **species counterpoint**.

3 Related Work

This section reviews existing methods of music style generation and music style transfer. Section 3.1 presents methods on music style generation which captures a wide variety of works. Section 3.2 highlights music style transfer methods and posits why some existing methods are not considered as performing music style transfer.

3.1 Music Style Generation

The range of methods for music style generation is notably wide. It includes methods based on rules, grammar, Markov chains, neural networks, evolutionary algorithms and general constrained optimization techniques. Moreover, each of these methods typically handle music style at one of three levels of music representation or at a mixture of levels.

3.1.1 Rules & Grammars

Grammar and rule-based methods use symbolic manipulation to analyze and generate a sequence of music notes according to the grammars or the rules. These grammars or rules represent temporal structures of varying lengths about a music piece, such as chord skeletons, melody and phrasing of individual parts. The atomic items in the grammars or rules are the music notes or chords. Several rules can be combined to form a hierarchical structure to represent long-term structures at different levels. During learning phase, the rules are hand-coded or learned using rule-based learning methods such as learning classified system and association rule learning. Learning can be accomplished using one or more music examples. During generation, the rules or grammar are applied to generate new note combinations, as well as phrases and sections.

Rule-based and grammar-based methods have been used for composition style generation to generate melody and harmony in Palestrina style [4, 26–33] and Bach chorale style [32]. They have also been used to generate harmony against a fixed melody [2]. They may use explicit Boolean functions [26] or fuzzy rules [29, 30] or probabilistic rules [2]. At the performance level, rules have been used to generate performance styles of various music eras [34], as well as novel performance styles [35, 36].

The strength of rule-based and grammar-based methods is their ability to capture the short, medium and long-term structure of the reference music in rules or grammar. Also, by constraining the style generation system with encoded rules and grammar, note sequences of similar style can be easily generated. However, the grammars or rules tend to be not comprehensive enough for generating all the nuances of music that make up an entire piece. Secondly, the rules and grammar

learnt may not generalize well to other types of styles apart from the reference music unless probability is included to make probabilistic or fuzzy rules.

3.1.2 Markov Models

The second class of methods use **Markov models** to generate music style [19, 37–50]. Markov models describe the statistical properties of the change of state, which is the pitch of the musical note or chord over time, and assume that the current state which is dependent only on the preceding state. It can be extended to higher-order Markov models to describe the statistical properties of the current state in terms of more than one past states.

Markov models with observable states are called Markov chains. Since their states are observable, their state change probabilities can be calculated directly from the input data samples and represented in a state transition matrix. Markov models with unobservable states are called Hidden Markov Models. They capture both state transition probabilities and output probabilities which are typically estimated based on maximum likelihood approach via algorithms such as expectation maximization and Markov chain Monte Carlo. In application, random sampling method is used to execute state transition and output production based on the estimated probabilities.

Markov models have been used to generate melodies in rock or arbitrary style [37, 38] and harmony in Palestrina style [40], chorale style [41, 42], Western pop style [19, 44] as well as a blend of styles from classical music [43]. They may also generate short music improvisations from pre-defined short musical patterns [45, 46] and jazz and rock rhythms [47] and rhythms for percussion [48–50]. Higher-order Markov chains are also used to generate chorale melodies of similar time signature and rhythmic structure [51]. At the performance level, Markov models focus on beat-wise structures such as inter-beat interval, inter-onset interval and local beats per minute, while other methods target short-term structures such as metrical strength which jointly models timing and dynamics. They can make tempo or loudness predictions on a single music note [52], monophonic melodies [53, 54] or monophonic streams by decomposing a music piece into its parts [55].

Methods based on Markov chains are simple and efficient whereas methods based on Hidden Markov models are more complex. Both types of methods are natural at introducing variety to the synthesized music due to its probabilistic state transition and output production. Low-order Markov models (e.g., order ≤ 3) can represent local statistical properties, which enable them to capture new short-term musical structure such as the relationship between neighbouring pitches. High-order Markov models can capture medium-term phrase structure in theory, but their space and time complexities are too large to be practically viable. For example, high-order Markov chains produce high-dimensional transition matrices, with many empty entries and they need a lot of training examples to fill the matrix entries. Moreover, there is no guarantee that the same music phrase or chord progression can be repeated in different sections. For the same reason, they are not practically viable for capturing long-term section structure. Markov chains can work on a single training music piece. In this case, they tend to produce state transition matrices with many empty entries.

3.1.3 Neural Networks

The third class of methods use **Neural Networks** to generate music in audio [21, 22, 56, 57], MIDI [58–61] and score formats [62]. Neural networks encode a note or chord as a numeric value. They learn a highly complex nonlinear mapping between the inputs and corresponding training outputs. During training, the weights of the neural networks are learnt using stochastic gradient descent and backpropagation. During the generation phase, the network's output is synthesized from the features extracted from the input and the features learned from training examples by feedforward neuron activation.

Families of neural networks used for composition style generation include recurrent neural networks (RNNs), long-short-term networks (LSTMs) and restricted Boltzmann machines (RBMs). RNNs are designed to represent sequences of data and have cyclical links between neurons that serve as a memory structure. It is used to generate Bach-like minuets and marches [62] and tonal melodies [58]. In contrast, LSTMs are a variant of RNNs which contain additional memory structure in its nodes that can maintain information in memory for longer periods of time. They use piano roll representation containing melody and chords to learn and generate chord sequences in jazz [60] and Celtic style [59]. On the other hand, RBNs are a type of neural network that can learn a probability distribution over its inputs. It is used to learn basic melody, harmony and local temporal coherence to generate polyphonic music [61].

In performance style generation, the neural networks are used to generate stylistic performances in MIDI format. The neural networks are trained on the temporal structure of music to learn a hierarchical non-linear transformation of their input and synthesizes a new sequence of onsets, note durations, loudness values for every musical note in the given score. Feedforward neural networks (FFNNs), [63–65], RNNs [66, 67], LSTMs [68] and RBMs [69, 70] are used to learn performance expressions like note-wise velocities in training data. In timbre style generation, the neural networks learn musical timbre representation that are further mapped to a particular musical instrument or a class of instruments to demonstrate mixtures of timbres [21, 22, 56, 57].

Several authors have identified the potential of RNNs, LSTMs and RBMs in generating new sequences of melodies and harmonies which reflect the style of the preceding sequences and they have been able to create interesting structures that mimicks various styles. However, several limitations exists with such methods. For example, the output of neural networks is derived from a highly non-linear function that needs to be mapped to twelve discrete music notes. When the regression result does not correspond exactly to a note, the nearest note is selected. This may drastically change the overall musical effect since the difference between notes that are one semitone apart are highly dissonant when sounded together. As a result, these neural networks have difficulties producing note combinations that are consistently repeated in medium-term phrases and long-term sections. Moreover, a large number of training music examples are required to train a neural network, especially a deep neural network.

3.1.4 Evolutionary Methods

Evolutionary methods generate music via evolutionary algorithms. These methods maintain a set (population) of possible music and combine different music pieces from this set to form new ones, without training. They repeat a cycle with 3 stages, namely evaluation, selection and reproduction with variation. In the evaluation stage, candidate solutions are generated from user-specified examples or in a random fashion, and are evaluated using a fitness function. In the selection phase, a new set of candidate solutions are probabilistically copied from the old set, in proportional to its fitness. In the subsequent stage, some operators are applied to the set to increase variation. The three-stage evolution cycle repeats until music pieces with high fitness is achieved. For effective evolution, a large population of possible music of a wide range of varieties is needed.

Evolutionary methods differ mainly in the choice of fitness function, whose objective is to minimize the difference between the generated style and the styles of the training examples. In composition style generation, these fitness functions include weighted sum of melodic features or music characteristics [71, 72, 72], distance between target music and training examples [73–75] and statistical properties of n-gram sequences [75]. In performance style generation, the fitness function factors in the timing and energy expressiveness of a performer's interpretation of the performer [76]. In timbre style generation, the fitness function is based on psychoacoustic features such as brightness and roughness that contribute to perceptual dissimilarity [77].

With evolutionary methods, new short-term structures and variations of given music can be produced easily. This is achieved by the re-selection of music phrases from existing musical pieces for modification, thus retaining short-term and possibly medium-term structure and patterns. However, they are not able to deterministically enable musical patterns to evolve into concrete long-term structures with contrasting musical sections because there is no mechanism for constraining musical phrases into a distinct sections.

3.1.5 General Constrained Optimization

General constrained optimization methods generate music by learning a mapping from input to output music using an objective function with constraints. Typically, they represent a note and a chord as a discrete item, although continuous numerical representation is possible. They include both continuous and discrete optimization methods. During the derivation phase, these methods estimate the mapping subject to constraints that encode music structure such as melody, harmony and rhythm. This can be accomplished by one or more music examples. At the generation phase, the mapping is applied to generate music notes, phrases, and possibly sections.

General constrained optimization methods have been used to generate composition styles using constraint logic programming to harmonize chorales [78], a greedy backtracking algorithm for serial music composition in 20th century style [79] and integer programming with melodic constraints to generate guitar solos [80]. At the performance level, they are used to elicit dynamics, articulation and timing charac-

teristics of notes from a piece or pieces of music by a constrained objective function. In [81], expressive dynamics and timings of score events are selected using a linear evaluative function, allowing its duet system (involving two performers) to generate an expressive piano accompaniment to interact with a human pianist. At timbre level, they aim to minimize the difference in timbre between a given set of instruments' sound samples and the generated timbre. This resultant timbre can sound like voices [82] or electroacoustic music whose dissonance and loudness is manipulated by mixing timbres [83].

Methods based on general constrained optimization can generate a variety of temporal music structures, depending on the type of constraints used. For example, if medium-term structures like chord phrasing are constrained, then the output is likely to contain chord phrase of a similar structure. On the other hand, if the constraints are specific enough to capture interesting patterns in a piece while leaving some aspects of music structure unconstrained for exploration, the method will be able to generate music with similar patterns that exhibit similar levels of sophistication. To date, constraints defined in existing methods are genre-specific and often do not generalize well to other musical genres. As a result, they tend to be more successful at producing genre-specific music structures.

3.2 Mixed Style Generation

Music styles can also be generated across multiple levels by using neural networks. During training, the neural network learns an intermediate representation derived from input pieces that is used for comparing with the target piece. Depending on the control parameters supplied to the network, the learnt representation may not follow the conventional representations of timbre, performance or composition. Therefore during the generation phase, the neural networks could combine the input features and training features at mixed levels of representation.

Some researchers have referred to such neural networks as performing music style transfer [16]. However, it is important to realize that these deep neural networks are not performing music style transfer as defined in Section 3.1. The source of the style information in music style transfer is defined in a specific level of representation, rather than at multiple levels of abstraction. In fact, the authors of [20] have specifically called their neural network task music translation instead of music style transfer to avoid confusion. Therefore, this QE paper categorizes the music translation task of [20] as performing mixed style generation.

In [20], the neural network is an autoencoder architecture where it learns an encoding of a mixture of performance and timbre representation from classical orchestra tracks and synthesizes separate instrumental tracks with multiple decoders. It combines the instrument tracks together to form a single piece, where the performance and timbre style from the training data is combined with the score content of the input music.

3.3 Music Style Transfer

Music style transfer can be approached at three levels, namely sound, performance and score. At the lowest level of sound, the timbre style is changed while performance control and composition content are preserved. At the middle level of performance, performance control style is changed while the composition content is preserved. At the highest level of score, the composition style is changed while the melody is preserved.

Our refined definition of music style transfer also requires that the transferred music style comes from specific music pieces. Methods that generate music style from desired properties that are hand-coded or learned from training examples are not considered as methods for music style transfer in this paper. Examples of such methods include those cited by Dai et al. [16] as performing timbre style transfer [21, 22], performance style transfer and composition style transfer [23–25]. Instead, they are classified as methods for music style generation.

With respect to this categorization scheme, there is no existing work that focus on performance style and timbre style transfer. However, there is an existing method for composition style transfer [5] that is not cited in Dai et al. [16]. This method performs a sub-task of composition style transfer where harmony is transferred across a piece. It models chords in lead sheets of Western popular songs based on music-theoretic knowledge and statistical learning and predicts pop song composers' harmonization decisions for new melodies. In general, it performs style generation in three phases, namely melody note-encoding, chord-tone determination and chord transition. The first phase encodes each melody note with 73 attributes. Some attribute values are automatically determined while other values like phrase index and fragment index of a phrase are manually annotated. The second phase captures the relationship between a melody note and chord using a classification tree. This classification tree is trained using C4.5 algorithm to classify a note as either a chord tone or non-chord tone. In the final phase, chord progression is captured using a predefined transition network whose transition probability is learned using Markov chain method. If only one training music example is used, then the resultant harmony could preserve at least the local structure of the training harmony, consistent with music style transfer. When more training examples are used, then the harmonies of various training examples could be mixed together, resulting in music style generation.

The strength of music style transfer is that it can work with a single reference music piece. Also it synthesizes new music pieces that retain the specific music style of the reference music pieces. In this way, it could allow for the fine control over the nuances of music style, as well as global control over the medium-term and long-term structures.

3.4 Summary

A summary of existing music style transfer and style generation models are presented in Table 1. These models can be classified according to which level of style is targeted for generation, namely composition, performance or timbre.

Table 2 compares properties of various categories of methods. In general, rule and

Table 1: Summary of music style generation and transfer methods.

Task	Style Level	Model	References
Style Transfer	Composition	Mixed	[5]
	Performance	—	—
	Timbre	—	—
Style Generation	Composition	Rule and grammar based	[4, 26–33]
		Markov	[19, 37, 38, 40–50]
		Neural Network	[58–60, 62]
		Evolutionary	[71–75]
		General constrained optimization	[78–80]
		Mixed	[84]
	Performance	Rule and grammar based	[34–36]
		Markov	[52–55, 85–87]
		Neural Network	[63–70]
		Evolutionary	[76]
		General constrained optimization	[81]
	Timbre	Neural Network	[21, 22, 56, 57]
		Evolutionary	[77]
		General constrained optimization	[82, 83]
	Mixed	Neural Network	[20]

grammar-based methods provide an effective symbolic means of generating compositions and performances in a particular style, as they can generate short, medium and long-term structures. However, they cannot generate music patterns that are not described by the rules and grammar. Stochastic neural networks such as Boltzmann machine can produce a variety of outputs give the same input due to its probabilistic nature. In contrast, other deterministic neural networks lack the ability to produce variations given the same input. Although most neural networks have succeeded in generating short-term structures that are evocative of a generalized style of training set, they generally still perform poorly at generating medium-term and long-term structures that describe musical phrasing and sections.

Markov models and evolutionary methods are able to introduce new short-term structures due to their stochastic nature which allow for variations from the same input and the reuse of musical patterns. However, both methods are unable to generate good long-term structures that correspond to music sections. In contrast, general constrained optimization methods can generate various temporal structures when appropriate constraints are incorporated. However, it may be computationally expensive to search for the optimal combination of short, medium and long-term structures that exhibit creative music composition.

Table 2: Comparisons of Various Methods. Short-term, medium-term and long-term structures refer to pitch relationship, music phrases and music sections, respectively. † No, unless probabilistic method is incorporated.

Methods	**Essence**	**Temporal structure**	**Variations**	**Representation of Note**	**Training music**
Rules & Grammars	Symbol manipulation	short, medium, long	no†	discrete	1 or more
Markov & Models	Probabilistic modeling	short	yes	discrete	1 or more
Neural Networks	Nonlinear mapping	short, may be medium	no†	continuous	many
Evolutionary Methods	Gene recombination	short, medium	possible	can be discrete	many
Constrained optimization	Nonlinear mapping	short, medium, possibly long	no†	can be discrete	1 or more

4 Overall Research Problem

This QE paper investigates a first attempt at harmony style transfer via general constrained optimization. The computational representation of a music score is detailed in section 4.1 and the problem of this QE paper is formally defined in section 4.2. Finally, section 4.3 describes the algorithm used for harmony style transfer.

4.1 Computational Representation of Music Score

The melody of a music piece consists of a sequence of musical notes or rests u_i, $i = 1, \ldots, m$. A note u_i has three attributes, namely pitch p_i, duration τ_i, and onset time t_i. Duration τ_i encodes relative time in terms of number of beats measured in fractions of 1/16. The onset time t_1 of the first note u_1 is 0. The onset time t_i of note u_i, for $i > 1$, is given by

$$t_i = \sum_{k=1}^{i-1} \tau_k. \tag{1}$$

Pitch is related to sound frequency. Pitch value is encoded as a MIDI note number ranging from 0 to 127, which corresponds to C-1 and G9 respectively. The standard pitch p^* for tuning is pitch number 69 (A4) with a frequency f^* of 440Hz. Pitch number p of any pitch is related to its frequency f by the equation

$$p = p^* + 12 \log_2 \frac{f}{f^*}. \tag{2}$$

A rest is a silent note, which is denoted using the pitch value of -1. The relationship

Table 3: Pitch dissonance ratings based on Huron's findings [88].

Pitch Interval θ_p	**Pitch Dissonance** δ_p	**Example:** $p = C$
0	0	C
1, 11	1	C♯,B
2, 10	0.8	D,A♯
3, 9	0.5	D♯, A
4, 8	0.4	E, G♯
5, 7	0.3	F,G
6	0.7	F♯

between two pitches can be described and measured in several ways. The absolute signed difference between two pitches p and q is called **pitch difference** d_p:

$$d_p(p, q) = q - p. \tag{3}$$

With p and q in MIDI note numbers, this difference measures the number of semitones between p and q. The relative difference between pitches p and q is called the **pitch interval** θ_p, which is defined as

$$\theta_p(p, q) = d_p(p, q) \bmod 12 = (q - p) \bmod 12. \tag{4}$$

Two pitches p and q played simultaneously creates a perceptual dissonance called **pitch dissonance** δ_p. The value of pitch dissonance (Table 3) is obtained based on [88], and it is related to pitch interval. Pitches with 0 pitch interval are said to belong to the same **pitch class** and their pitch dissonance is 0.

The accompaniment of a music piece consists of a sequence of chords c_j, $j = 1, \ldots, n$. A chord c_j consists of two or more notes played simultaneously or in succession. As a first attempt, this QE paper considers only chords with simultaneous notes. The onset time of a chord can be calculated as for the onset time of a note.

The concept of dissonance can be applied to chords as well. The dissonance δ_{pc} between a pitch p and a chord c is a summary of the pitch dissonance δ_p between pitch p and every pitch in c:

$$\delta_{pc}(p, c) = \frac{1}{|C|^2} \sum_{q \in C} \sum_{r \in C} \delta_p(q, r), \tag{5}$$

where $C = p \cup c$ and $|C|$ denotes the number of notes (or pitches) in the set containing chord c and pitch p. Similarly, the dissonance δ_c between two chords c and c', called **chord dissonance**, is a summary of the dissonance δ_p between all pairs of pitches in the two chords:

$$\delta_c(c, c') = \frac{1}{|c||c'|} \sum_{p \in c} \sum_{q \in c'} \delta_p(p, q). \tag{6}$$

The relationship between chords can also be measured based on the number of pitches they do not have in common, called **chord dissimilarity**. The chord

dissimilarity $d_c(c, c')$ between a reference chord c and another chord c' is:

$$d_c(c, c') = \frac{|c| - |(c \cap c')|}{|c|}, \tag{7}$$

where $|c|$ denotes the number of notes (or pitches) in a chord c.

4.2 Problem Formulation

The inputs of the harmony style transfer problem consist of a reference music, with both melody and accompaniment, and a target music with melody. For now, this QE paper considers the simplified case where the reference music and the target music have the same time signature and number of bars. If the reference music and the target music have different key signatures, then the reference music is transposed to the key signature of the target music using standard key transposition method [89]. Therefore, in the following problem formulation, the reference and the target are considered to have the same key signature as well.

The output of the problem is a resultant music that contains the target melody and the synthesized accompaniment. The chords of the resultant accompaniment follow the onset time of the reference chords. This allows for the harmony style in the entire length of the reference music to be transferred across to the resultant music.

The reference melody is different from the resultant melody, which is the target melody. The reference chords which are harmonized to the reference melody may not harmonize with the resultant melody. So, in transferring the reference chords to the target music, the reference chords need to be modified to harmonize with the resultant melody. Then, the resultant harmony would not be the same as the reference harmony, unless something about the reference style is preserved. A possible style element to be preserved while allowing for the chords to change is the harmonic tension.

Harmonic tension refers to the tension created by chords and melody sounding together. It is determined by the pitch intervals formed by the notes in the chords and the melody at the same onset time. It can be measured in terms of the dissonance δ_{pc} between a melody pitch p and the accompanying chord c. To preserve the harmonic tension of the reference harmony in the resultant music, the harmonic tension of the chords in the resultant should be as similar as possible to the harmonic tension of the corresponding chords in the reference. Let p_j denote the reference melody pitch at the same onset time as the reference chord c_j, and p'_j denote the resultant melody pitch at the same onset time as the resultant chord c'_j. Then, preservation of harmonic tension requires the minimization of the difference in the pitch-chord dissonance between the resultant and the reference:

$$\text{minimize } (\delta_{pc}(p'_j, c'_j) - \delta_{pc}(p_j, c_j))^2, \text{ for each } j = 1, ..., n. \tag{8}$$

Another component about harmonic tension is the chord progression. In this case, the chord progression of the resultant should be as similar as possible to the chord progression of the reference. This is achieved by minimizing the difference in

chord dissonance over time between the resultant and the reference:

$$\text{minimize } (\delta_c(c'_j, c'_{j-1}) - \delta_c(c_j, c_{j-1}))^2, \text{ for each } j = 1, ..., n. \tag{9}$$

Finally, the chords pitches in the resultant should be as similar as the reference chords as possible. This is achieved by minimizing the chord dissimilarity between the resultant and the reference:

$$\text{minimize } d_c(c_j, c'_j), \text{ for each } j = 1, ..., n. \tag{10}$$

The problem definition can be summarized as follows:

Problem Definition

Inputs

- Reference music F, with a sequence of notes $u_i, i = 1, \ldots, m$, and a sequence of chords $c_j, j = 1, \ldots, n$.
- Target music T, with a sequence of notes $v_k, k = 1, \ldots, m'$.
- Reference music F and target music T have the same key signature, time signature and number of bars.

Output

- Resultant music R whose melody notes are the same as the target's melody notes v_k and whose chords are to be generated as $c'_j, j = 1, \ldots, n$.

Constraints

0. Hard constraint.
   R's chords c'_j and F's chords c_j, for each j, have the same onset time.

1. Main constraint. Matching of harmonic tension.
   R's harmonic tension and F's harmonic tension are as similar as possible. This is achieved by minimizing the difference in pitch-chord dissonance between R and F (Eq. 8).

2. Matching chord progression.
   R's chord progression and F's chord progressions are as similar as possible. This is achieved by minimizing the difference in chord dissonances between R and F over time (Eq. 9).

3. Matching chords.
   R's chords and F's chords are as similar as possible. This is achieved by minimizing the chord dissimilarity between R and F (Eq. 10).

4.3 Algorithm

The harmony style transfer problem defined in Section 4.2 takes the reference music F and target music T as inputs and computes a new set of chords for the resultant music R. It is a complex problem that needs to satisfy three constraints. The essence of this problem is to find target chords that match the target melody notes and at the same time retain the harmonic tension of the reference music. Since the pitches and chords are discrete items, this problem is a discrete matching problem. As a first attempt, a greedy matching algorithm is proposed, which has the structure of Algorithm 1 [90].

The greedy algorithm iteratively finds a resultant chord that matches the constraints. Constraint #0 is trivially satisfied by having the resultant chord c'_j timed to the onset time of the corresponding reference chord c_j. Next, a small number of chords are selected in an attempt to satisfy Constraint #1, the main constraint. In other words, find some chords with the smallest difference in harmonic tension H_j:

$$H_j = (\delta_{pc}(p'_j, c'_j) - \delta_{pc}(p_j, c_j))^2. \tag{11}$$

Among these chords, the one that best matches the Constraint #2 and #3 is selected as the resultant chord c'_j. The combined constraint is defined as:

$$C_j = \alpha(\delta_c(c'_j, c'_{j+1}) - \delta_c(c_,c_{j+1}))^2 + d_c(c_j, c'_j). \tag{12}$$

where positive parameter α is used to balance the various terms. It is acknowledged that this greedy algorithm may not produce the optimal result but it is a simple algorithm to begin with.

Algorithm 1: Greedy Algorithm

Input: Reference music F, with a sequence of notes and chords, Target music T, with a sequence of notes.

Output: Resultant music R.

1 Copy target melody notes $v_k, k = 1, ..., m'$, to resultant music.
2 **for** *each reference chord* $c_j, j = 1, ..., n$ **do**
3 Identify the reference melody pitch p_j and the resultant melody pitch p'_j at the same onset time as the reference chord c_j.
4 **if** $j = 1$ **then**
5 Select resultant chord c'_j with the smallest chord dissimilarity $d_c(c_j, c'_j)$ to c_j.
6 **else**
7 Identify K chords with the smallest difference in harmonic tension H_j (Eq. 11), given the pitches and the chords identified in Step 3.
8 Among these K chords, select the chord with the smallest constraint value C_j (Eq. 12).
9 Set resultant chord c'_j as the chord selected in Step 8.

For the first chord, the chord with the smallest chord dissimilarity with respect to the reference chord is chosen. Step 7 and 8 are redundant for the first chord as

Table 4: Pitch-chord dissonance table. This table lists pitch p and major chords c with the corresponding pitch-chord dissonance. The dissonance values between other pitches and chords are the transposed version of these values.

Pitch p	**Chord** c	**Pitch-Chord Dissonance** δ_{pc}
C	C-E-G	0.24
C	D♭-F-A♭	0.36
C	D-F♯-A	0.4
C	E♭-G-B♭	0.35
C	E-G♯-B	0.38
C	F-A-C	0.25
C	F♯-A♯-C♯	0.46
C	G-B-D	0.41
C	A♭-C-E♭	0.26
C	A-C♯-E	0.39
C	B-♭-D-F	0.39
C	B-D♯-F♯	0.42

there is no previous chord for computing the chord progression. Instead they apply to subsequent chords.

Step 7 first computes the harmonic tension of all possible chords, and then selects K chords with the smallest difference in harmonic tension. The current implementation considers only triads, i.e., chords with three notes. There is a total of 24 triads: 12 in major and 12 in minor mode. The major triads are C-E-G, D♭-F-A♭, D-F♯-A, E♭-G-B♭, E-G♯-B, F-A-C, F♯-A♯-C♯,G-B-D, A♭-C-E♭, A-C♯-E, B-♭-D-F, B-D♯-F♯ and the minor triads are C-E♭-G, D♭-E-A♭, D-F-A, E♭-G♭-B♭, E-G-B, F-A♭-C, F♯-A-C♯,G-B♭-D, G♯-B-D♯, A-C-E, B♭-D♭-F, B-D-F♯. The root note of the chords considered lies in the same octave as the root note of the reference chord c_j.

Step 8 computes the chord progression difference and chord dissimilarity between the reference and K chords from Step 7 the values are combined in constraint C_j. In the implementation of C_j, α is set to 2 to favour the minimization of chord progression difference over chord dissimilarity. Finally, the chord with the smallest C_j is selected as the resultant chord.

In implementation, the pitch-chord dissonance between a pitch C and every major triad are considered. Table 4 shows an example of the pitch-chord dissonance between pitch C and each of the 24 triads. The pitch-chord dissonance values using the remaining melody pitches (C♯, D, D♯, etc.) can be derived by transposing the pitches and chords in Table 4 by half steps. The same applies to pitch-chord dissonances involving minor triads. These values can be pre-computed as a 12x24 table. Similarly, chord dissonance can be pre-computed as a 24x24 table. For now, the set of 24 major and minor triads are selected as most popular music use these chords. This chord set can be expanded subsequently to include more complex chords found in classical and jazz music.

Since there are 3 chords with least dissonance compared to the pitch, K is set to

3. This value of K balances the optimization constraints in the algorithm If $K = 1$, the algorithm will match the resultant harmonic tension to the reference harmonic tension well, but the reference chord progression and chord will not be retained. If $K = 24$, the algorithm will match the reference chord and chord progression well, but the reference harmonic tension may not be retained. Among the 3 possible chords returned from Step 7, the chord that matches the reference chord and chord progression better is set as the resultant chord.

5 Experiments and Discussion

5.1 Harmony Transfer Test

The objective of the harmony transfer test is to assess whether the harmony from reference music is transferred to the target music successfully, given a pair of target and reference music piece.

The test data is a set of nursery rhymes and popular music, where the chords of the dataset are major or minor triads. Each pair of reference and target music pieces are selected such that they contain the same number of bars and with the same time signature, so that both music pieces are temporally aligned. The reference music is transposed to the key of the target music prior to the test. This ensures that the reference harmony harmonic tension and chord progressions are computed in the context of the same key as the target music.

In the harmony transfer experiment, the reference chords were mapped to either 0 or 1 target melody note at same onset time, for every pair of target and reference music piece. For the reference chords without a corresponding target melody note (i.e. rest), the melody pitch is set to the root of the chord, so that the pitch-chord dissonance correspond to the reference chord.

As a control, a baseline algorithm was designed to generate the chords for the target melody without any reference music chord. It has a similar structure to the greedy algorithm 1, except that parts referring to reference music are ignored. Steps 2 and 3 are modified by using the target chords to get the corresponding onset times and melody to compute the resultant chord. In steps 7 and 8 of Algorithm 1, the parts involving pitch-chord dissonances of reference music and chord dissonance of reference chords in Equation 11 and 12 are ignored. The resultant chords serves as the baseline accompaniment.

The test procedure is as follows:

1. For each pair of target and reference music piece,

2. Apply the greedy algorithm and the base algorithm to compute the resultant chords.

3. Compute the pitch-chord dissonances and chord dissonances between resultant and reference chords for assessment of performance.

To evaluate the quality of the harmony transfer accompaniment for the target melody, we use the reference music accompaniment as ground truth and baseline

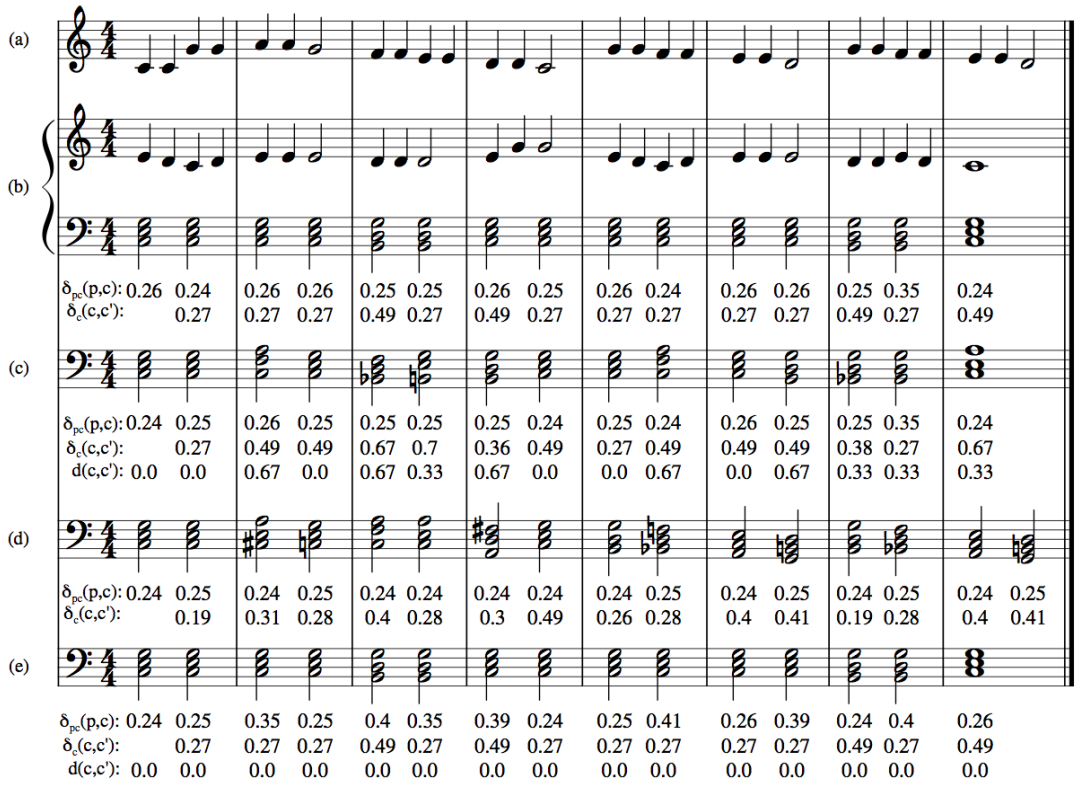


Figure 4: Score of various accompaniment for *Twinkle, Twinkle, Little Star*. Original music: (a) Target melody *Twinkle, Twinkle, Little Star* (TTLS), (b) reference music *Mary Had A Little Lamb* (MHLL). Generated accompaniment: (c) Algorithm 1, $K = 3$, (d) baseline algorithm, $K = 3$ and (e) Algorithm 1, $K = 24$.

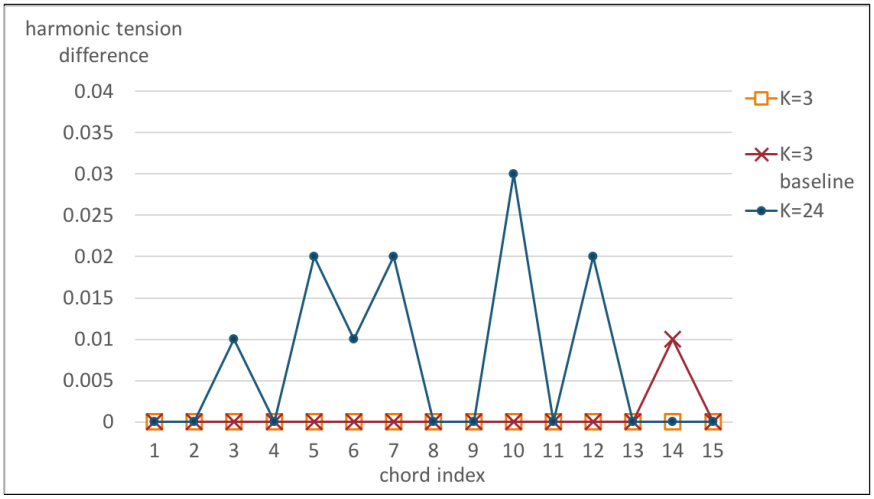
accompaniment for comparison. To compare two accompaniments for the target melody, we measure their difference in the harmonic tension (Eq. 11), difference in chord dissonance over time (Eq. 9) and chord similarity (Eq. 10).

5.2 Results and Discussion

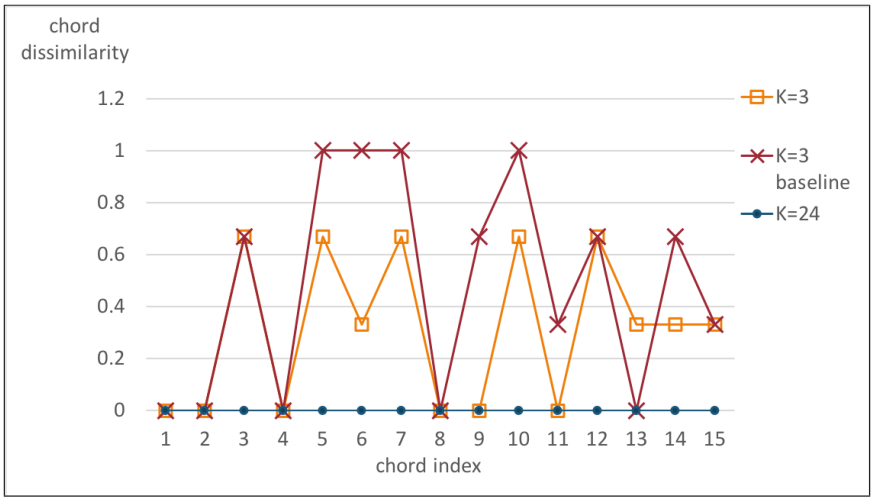
Figure 4 shows an example of the accompaniment generated by Algorithm 1 and baseline accompaniment generated by the baseline algorithm[2]. For Algorithm 1, *Twinkle, Twinkle, Little Star* is used as the target music piece and *Mary Had A Little Lamb* is used as the reference music piece. For the base algorithm, only *Twinkle, Twinkle, Little Star* is used as the input music. Figure 5 summarizes harmonic tension difference and chord dissimilarity of the results from the reference music.

When $K = 3$, the harmonic tension difference between the reference and generated accompaniment by harmony transfer is much smaller as compared to the harmonic tension difference between the reference and baseline accompaniment. Similarly, the chord dissimilarity values are smaller for the generated accompaniment than that of the baseline accompaniment. This shows that some degree of harmony

[2]Refer to the web page `https://www.comp.nus.edu.sg/~slewyh/QE-music-examples.html` for results.



(a)



(b)

Figure 5: Results based on the harmonic measures. (a) Harmonic tension difference and (b) chord dissimilarity.

style (i.e. harmonic tension distribution and chord dissonance progression) in the reference music is present in the generated accompaniment. In contrast, the baseline accompaniment differs from the harmony style of the reference music in terms of its chord progression and harmonic tension since no reference music piece was used.

When $K = 24$, Algorithm 1 selected chords that are most similar reference chords to match the target melody. The example in Figure 4 shows that only reference chords were selected. However, it did not retain much of the reference harmonic tension.

Figure 6 and 7 shows the accompaniment generated for the 12-bar target piece *Twinkle, Twinkle, Little Star* using various reference music excerpts such as *Mary Had A Little Lamb*, *Do-Re-Mi* and *Canon in D* that span 12 bars[3]. It shows that

[3]Refer to the web page `https://www.comp.nus.edu.sg/~slewyh/QE-music-examples.html` for results.

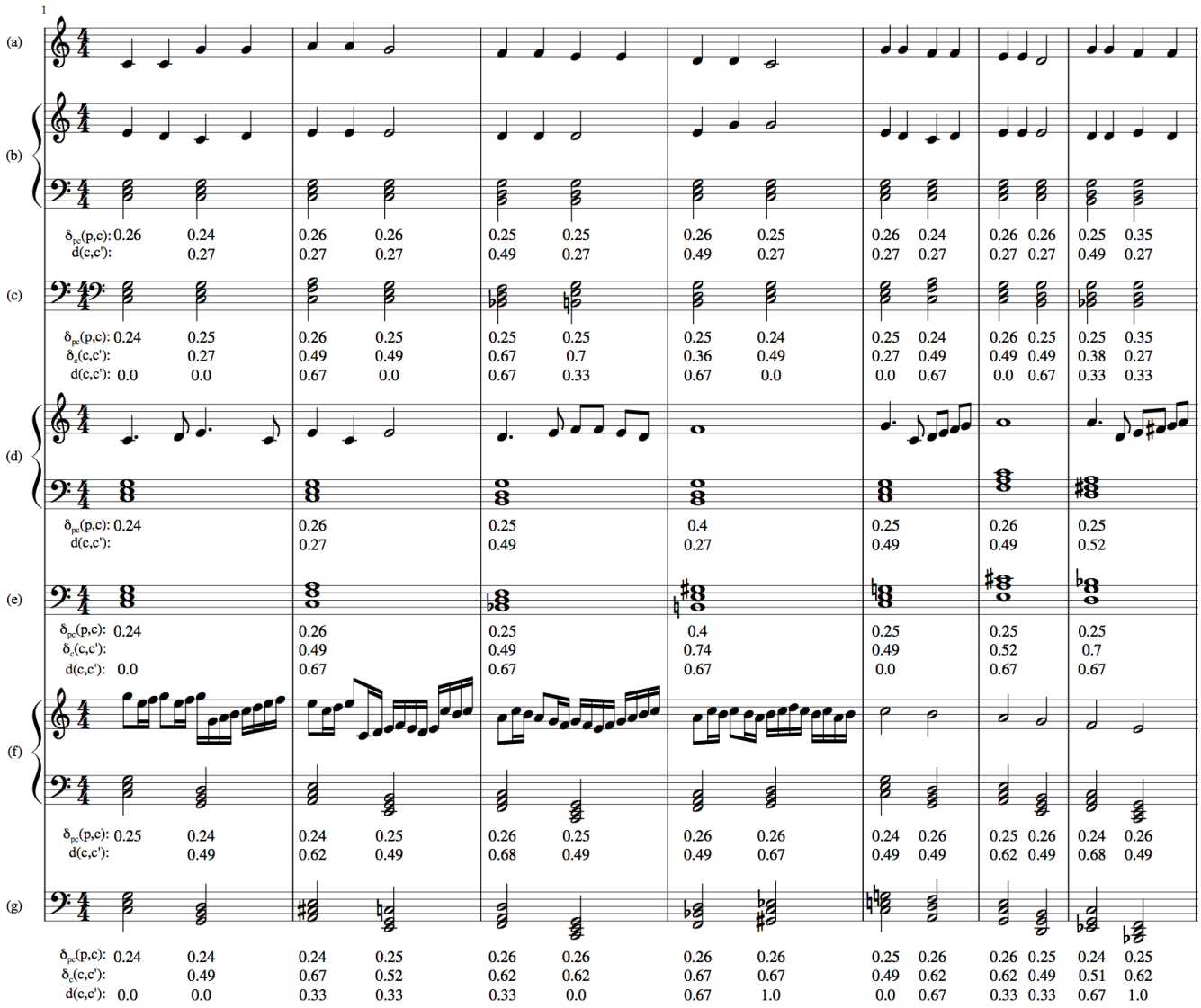


Figure 6: Bars 1-7 of various accompaniment for *Twinkle, Twinkle, Little Star*. (a) Target melody *Twinkle, Twinkle, Little Star* (TTLS). Reference music are (b) *Mary Had A Little Lamb* (MHLL), (d) *Do-Re-Mi* and (e) *Canon in D* transposed to the C major key. Their respective accompaniment using Algorithm 1 in (c),(e) and (g).

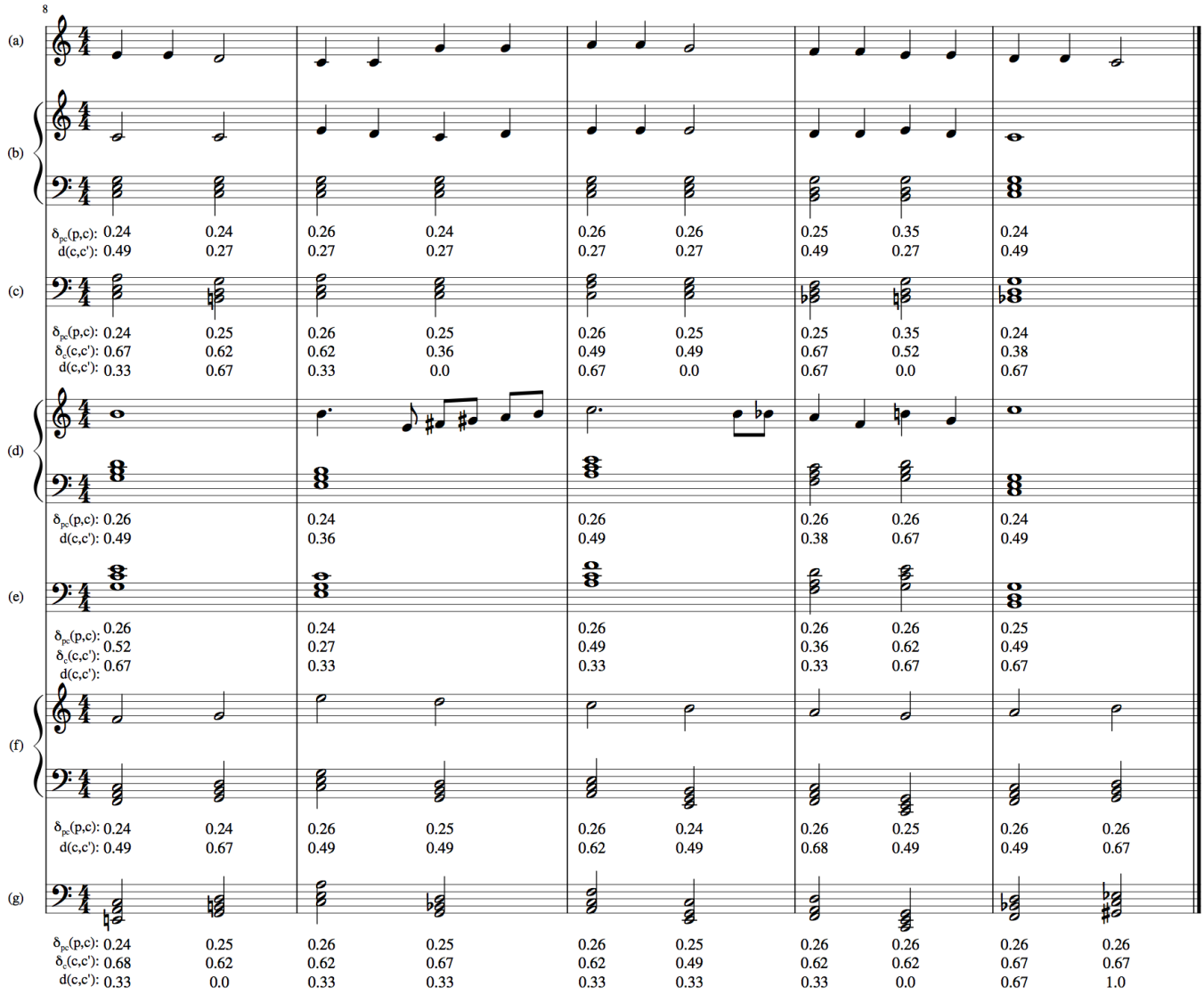


Figure 7: Bars 8-12 of various accompaniment for *Twinkle,Twinkle, Little Star*. (a) Target melody *Twinkle,Twinkle, Little Star* (TTLS). Reference music are (b) *Mary Had A Little Lamb* (MHLL), (d)*Do-Re-Mi* and (e)*Canon in D* transposed to the C major key. Their respective accompaniment using Algorithm 1 in (c),(e) and (g).

the algorithm can consistently generate chords with harmonic tension similar to the reference music harmony. In the case of the accompaniment Figure 6(e) with reference from *Canon in D*, the result even retained the short-term and medium-term music structure of the piece.

6 Future Work

The current greedy algorithm does not consider the case where the target melody pitch is not aligned to the onset time of the reference chord. One way to handle this situation is to look for neighbouring pitches with onset time close to that of the reference chord.

It also does not consider the case with reference music that does not have the same number of bars and time signature. By using a temporal matching algorithm, the sections of both pieces can be aligned. More bars can be repeated in the shorter piece to make up the same number of bars. Also, the duration of a reference bar with different number of beats can be shortened or extended to match that of the target music piece.

The current implementation considers only 24 major and minor chords. This is not enough to handle complex music of other styles such as classical and jazz music. A simple way to handle this is to include other chords commonly found in classical and jazz music to increase the stylistic diversity of the resultant music.

7 Conclusion

This QE paper has proposed a new algorithm of harmony style transfer where the music harmony of a reference music was transferred to a target music piece. The algorithm generated piece a matching accompaniment for the target melody by exploiting the harmonic relationship between a melody note and chord called harmonic tension. Harmonic tension was used as the main measure instead of the reference chord properties since the resultant chord should match the target melody instead of the reference melody.

Based on the test results, the algorithm has been found to work well when $K = 3$, where a balance of reference harmonic tension and chords are retained in the result. Also, its resultant accompaniment matches the target melody. Similar performance was obtained when other reference music pieces were used.

This algorithm has the potential to generate convincing music with well-crafted medium-term and long-term music structures since it tries to preserve the overall harmonic tension of the reference music piece across all bars. However, it does not generate new melodies, unlike other music style generation methods that model melody styles. Nevertheless, the resultant music can practically serve as composition material for games, film music and music therapy.

References

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

[2] G. Aguilera, J. L. Galán, R. Madrid, A. M. Martínez, Y. Padilla, and P. Rodríguez, "Automated generation of contrapuntal musical compositions using probabilistic logic in derive," *Mathematics and Computers in Simulation*, vol. 80, no. 6, pp. 1200–1211, 2010.

[3] M. Chemillier, "Improvising jazz chord sequences by means of formal grammars," *Journées d'informatique musicale*, p. 121–126, 2001.

[4] D. Quick, "Generating music using concepts from Schenkerian analysis and chord spaces," tech. rep., Yale University, 2010.

[5] 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.

[6] L. Bigo and D. Conklin, "A viewpoint approach to symbolic music transformation," in *International Symposium on Computer Music Multidisciplinary Research*, pp. 213–227, Springer, 2015.

[7] N. Cunha, A. Subramanian, and D. Herremans, "Generating guitar solos by integer programming," *Journal of the Operational Research Society*, vol. 69, pp. 971–985, 2018.

[8] N. Tokui and H. Iba, "Music composition with interactive evolutionary computation," in *International Conference on Generative Art*, vol. 17, pp. 215—-226, Springer, 2000.

[9] M. Towsey, A. Brown, S. Wright, and J. Diederich, "Music composition with interactive evolutionary computation," *Educational Technology Society*, vol. 4, no. 2, pp. 54—-65, 2001.

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

[11] F. Liang, "Bachbot: Automatic composition in the style of Bach chorales," Master's thesis, University of Cambridge, 2016.

[12] G. Hadjeres, F. Pachet, and F. Nielsen, "DeepBach: a steerable model for Bach chorales generation," in *Proceedings of the Conference on Machine Learning*, pp. 1362–1371, 2017.

[13] D. Herremans, C.-H. Chuan, and E. Chew, "A functional taxonomy of music generation systems," *ACM Computing Surveys*, vol. 50, pp. 69–101, 2017.

[14] J.-P. Briot, G. Hadjeres, and F. Pachet, "Deep learning techniques for music generation-a survey," 2017, arXiv:1709.01620.

[15] G. Widmer, "Getting closer to the essence of music: The *Con Espressione* Manifesto," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 8, no. 2, pp. 19–32, 2017.

[16] S. Dai, Z. Zhang, and G. G. Xia, "Music style transfer: A position paper," in *Proceedings of the International Workshop on Musical Metacreation*, 2018, arXiv:1803.06841.

[17] R. Felts, *Reharmonization Techniques*. Berklee Press, 2002.

[18] A. Schoenberg, P. Carpenter, and S. Neff, *The Musical Idea and the Logic, Technique and Art of Its Presentation*. Indiana University Press, 2006.

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

[20] N. Mor, L. Wolf, A. Polyak, and Y. Taigman, "A universal music translation network," 2018, arXiv:1805.07848.

[21] J. Engel, C. Resnick, A. Roberts, S. Dieleman, M. Norouzi, D. Eck, and K. Simonyan, "Neural audio synthesis of musical notes with WaveNet autoencoders," in *Proceedings of the International Conference on Machine Learning*, vol. 70.

[22] P. Verma and J. O. Smith, "Neural style transfer for audio spectograms," 2018, arXiv:1801.01589.

[23] F. Zalkow, S. Brand, and B. Graf, "Musical style modification as an optimization problem," in *Proceedings of International Computer Music Conference*, p. 206–211, 2016.

[24] M. Kaliakatsos-Papakostas, M. Queiroz, C. Tsougras, and E. Cambouropoulos, "Conceptual blending of harmonic spaces for creative melodic harmonisation," *Journal of New Music Research*, vol. 46, no. 4, pp. 305–328, 2017.

[25] S. Lattner, M. Grachten, and G. Widmer, "Imposing higher-level structure in polyphonic music generation using convolutional restricted boltzmann machines and constraints," 2016, arXiv:1612.04742.

[26] K. Ebcioğlu, "An expert system for harmonizing four-part chorales," *Computer Music Journal*, vol. 12, no. 3, pp. 43–51, 1988.

[27] D. Herremans and K. Sörensen, "Fux, an android app that generates counterpoint," in *Proceedings of IEEE Symposium on Computational Intelligence for Creativity and Affective Computing (CI- CAC)*, pp. 48–55, 2013.

[28] T. Tanaka, B. Bemman, and D. Meredith, "Constraint programming formulation of the problem of generating milton babbitt's all-partition arrays," in *Proceedings of the International Conference on Principles and Practice of Constraint Programming* (F. Toulouse, ed.), 2016.

[29] A. Yilmaz and Z. Telatar, "Note-against-note two-voice counterpoint by means of fuzzy logic," *Knowledge-Based Systems*, vol. 23, no. 3, pp. 256–266, 2010.

[30] N. Gwee, *Complexity and Heuristics in Rule-Based Algorithmic Music Composition, Ph.* PhD thesis, Louisiana State University, 2002.

[31] C. Z. A. Huang and E. Chew, "Palestrina pal: a grammar checker for music compositions in the style of palestrina," in *Proceedings of Conference on Understanding and Creating Music*, 2005.

[32] E. Gilbert and D. Conklin, "A probabilistic context-free grammar for melodic reduction," in *Proceedings of International Workshop on Artificial Intelligence and Music at International Joint Conference on Artificial Intelligence*, (Hyderabad, India), 2007.

[33] M. Gogins, "Score generation in voice-leading and chord spaces," in *Proceedings of the International Computer Music Conference*, 2006.

[34] A. Friberg and E. Bisesi, "Using computational models of music performance to model stylistic variations," in *Expressiveness in Music Performance: Empirical Approaches Across Styles and Cultures*, pp. 240–259, Oxford: Oxford University Press, 2014.

[35] A. Friberg, "pDM: an expressive sequencer with real-time control of the KTH music-performance rules," *Computer Music Journal*, vol. 30, pp. 37–48, 2006.

[36] E. Bisesi, R. Parncutt, and A. Friberg, "An accent-based approach to performance rendering: music theory meets music psychology," in *Proceedings of the International Symposium on Performance Science*, pp. 27–32, 2011.

[37] F. Pachet, P. Roy, and G. Barbieri, "Finite-length markov processes with constraints," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2011.

[38] M. McVicar, S. Fukayama, and M. Goto, "AutoLeadGuitar: Automatic generation of guitar solo phrases in the tablature space," in *Proceedings of International Conference on Singal Processing*, no. 12, pp. 599–604, IEEE, 2014.

[39] R. Wooller and A. R. Brown, "Investigating morphing algorithms for generative music," in *Proceedings of the International Conference on Generative Systems in the Electronic Arts*, 2005.

[40] M. Farbood and B. Schoner, "Analysis and synthesis of palestrina-style counterpoint using Markov chains," in *Proceedings of the International Computer Music Conference*, pp. 471–474, 2001.

[41] M. Allan and C. Williams, "Harmonising chorales by probabilistic inference," *Advances in neural information processing systems*, pp. 25–32, 2005.

[42] L. Yi and J. Goldsmith, "Automatic generation of four-part harmony," in *Proceedings of CEUR Workshop In Bayesian Modeling Applications*, 2007.

[43] M. Kaliakatsos-Papakostas and M. Queiroz, "Conceptual blending of harmonic spaces for creative melodic harmonisation," *Journal of New Music Research*, vol. 46, no. 4, pp. 305–328, 2017.

[44] A. Papadopoulos, P. Roy, and F. Pachet, "Avoiding plagiarism in Markov sequence generation," in *Proceedings of AAAI Conference on Artificial Intelligence*, (Quebec), 2014.

[45] M. Grachten, "JIG : jazz improvisation generator," in *Proceedings of the Workshop on Current Research Directions in Computer Music*, pp. 1–6, 2001.

[46] B. Manaris, D. Hughes, and Y. Vassilandonakis, "Monterey mirror: Combining Markov models," in *Proceedings of the IEEE Conference on Evolutionary Computation*, 2011.

[47] C. Ames and M. Domino, "Cybernetic composer: an overview," in *Understanding music with AI*, pp. 186–205, Cambridge: MIT Press, 1992.

[48] A. Tidemann and Y. Demiris, "A drum machine that learns to groove," in *Proceedings of the Annual Conference on Artificial Intelligence*, pp. 144–151, Springer, 2008.

[49] M. Marchini and H. Purwins, "Unsupervised generation of percussion sound sequences from a sound example," in *Proceedings of the Sound and Music Computing Conference*, pp. 205–218, 2010.

[50] A. Hawryshkewich, P. Pasquier, and A. Eigenfeldt, "Beatback: A real-time interactive percussion system for rhythmic practise and exploration," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, pp. 100–105, 2011.

[51] F. Brooks Jr, A. Hopkins Jr, P. Neumann, and W. Wright, *An experiment in musical composition*, pp. 23–40. 1992.

[52] K. Teramura, H. Okuma, Y. Taniguchi, S. Makimoto, and S. Maeda, "Gaussian process regression for rendering music performance," in *Proceedings of the International Conference on Music Perception and Cognition*, 2008.

[53] G. Grindlay and D. Helmbold, "Modeling, analyzing, and synthesizing expressive piano performance with graphical models," *Machine Learning*, vol. 65, no. 2, pp. 361–387, 2006.

[54] Y. Gu and C. Raphael, "Modeling piano interpretation using switching kalman filter," in *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 145–150, 2012.

[55] K. Okumura, S. Sako, and T. Kitamura, "Laminae: a stochastic modeling-based autonomous performance rendering system that elucidates performer characteristics," in *Joint Proceedings of the International Computer Music Conference and the Sound and Music Computing Conference*, pp. 1271–1276, 2014.

[56] J. W. Kim *et al.*, "Neural music synthesis for flexible timbre control," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2019–2019, 2019.

[57] S. Huang, Q. Li, C. Anil, X. Bao, S. Oore, and R. B. Grosse, "Timbretron: A WaveNet(CycleGAN(CQT(Audio))) pipeline for musical timbre transfer," in *International Conference on Learning Representations*, 2019.

[58] K. Agres, J. E. DeLong, and M. Spivey, "The sparsity of simple recurrent networks in musical structure learning," in *Proceedings of the Annual Conference of the Cognitive Science Society*, pp. 3099–3104, 2009.

[59] B. L. Sturm, J. F. Santos, O. Ben-Tal, and I. Korshunova, "Music transcription modelling and composition using deep learning," in *Proceedings of the Conference on Computer Simulation of Musical Creativity*, 2016.

[60] D. Eck and J. Schmidhuber, "A first look at music composition using LSTM recurrent neural networks," tech. rep., Istituto Dalle Molle Di Studi sull'Intelligenza Artificiale, 2002.

[61] 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 International Conference on Machine Learning*, (New York, NY, USA), ACM, 2012.

[62] M. C. Mozer and T. Soukup, "Connectionist Music Composition Based on Melodic, Stylistic, and Psychophysical Constraints," in *Music and Connectionism*, The MIT Press, 2003.

[63] R. Bresin, "Artificial neural networks based models for automatic performance of musical scores," *Journal of New Music Research*, vol. 27, pp. 239–270, 1998.

[64] C. Chacón, C. E., and M. Grachten, "An evaluation of score descriptors combined with non-linear models of expressive dynamics in music," in *Proceedings of the International Conference on Discovery Science*, pp. 48–62, 2015.

[65] S. I. Giraldo and R. Ramirez, "A machine learning approach to discover rules for expressive performance actions in jazz guitar music," *Frontiers in Psychology*, vol. 7, p. 1965, 2016.

[66] C. Chacón, C. E., and M. Grachten, "The basis mixer: a computational romantic pianist," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2016.

[67] C. Chacón, T. Gadermaier, G. Widmer, and M. Grachten, "An evaluation of linear and non-linear models of expressive dynamics in classical piano and symphonic music," *Machine Learning*, vol. 106, pp. 887–909, 2017.

[68] I. Malik and C. H. Ek, "Neural translation of musical style," 2017, arXiv:1708.03535.

[69] M. Grachten and F. Krebs, "An assessment of learned score features for modeling expressive dynamics in music," *IEEE Transactions on Multimedia*, vol. 16, pp. 1211–1218, 2014.

[70] S. van Herwaarden, M. Grachten, and W. B. de Haas, "Predicting expressive dynamics in piano performances using neural networks," *Proceedings of the International Society for Music Information Retrieval Conference*, vol. 15, pp. 47–52, 2014.

[71] V. M. Marques, V. Oliveira, S. Vieira, and A. C. Rosa, "Music composition using ge- netic evolutionary algorithms," in *Proceedings of the IEEE Conference on Evolutionary Computation*, pp. 714–719, 2000.

[72] M. Johnson, D. R. Tauritz, and R. Wilkerson, "Evolutionary computation applied to melody generation," in *Proceedings of the Artificial Neural Networks in Engineering (ANNIE) Conference*, 2004.

[73] A. Gartland-Jones, "Can a genetic algorithm think like a composer?," in *Proceedings of the Generative Art Conference*, 2002.

[74] E. Özcan and T. Erçal, "A genetic algorithm for generating improvised music," in *Proceedings of the International Conference on Artificial Evolution*, pp. 266–277, 2008.

[75] J. Wolkowicz, M. Heywood, and V. Keselj, "Evolving indirectly represented melodies with corpus-based fitness evaluation," in *Proceedings of the Conference on Applications of Evolutionary Computation*, pp. 603–608, 2009.

[76] R. Ramirez and A. Hazan, "Inducing a generative expressive performance model using a sequential-covering genetic algorithm," in *Proceedings of the Annual Conference on Genetic and evolutionary computation*, pp. 2159–2166, ACM, 2007.

[77] 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.

[78] C. P. Tsang and M. Aitken, "Harmonizing music as a discipline of constraint logic programming," in *Proceedings of the International Computer Music Conference*, 1999.

[79] B. Bemman and D. Meredith, "Generating Milton Babbitt's all-partition arrays," *Journal of New Music Research*, vol. 45, no. 2, p. 2016, 2016.

[80] "Uma abordagem baseada em programação linear inteira para a geração de solos de guitarra," in *Proceedings of the Simpósio Brasileiro de Pesquisa Operacional*, 2016.

[81] G. Xia and Dannenberg, "Duet interaction: learning musicianship for automatic accompaniment," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, 2015.

[82] T. A. Hummel, "Simulation of human voice timbre by orchestration of acoustic music instruments," in *Proceedings of the International Computer Music Conference*, p. 185, 2005.

[83] N. Collins, "Automatic composition of electroacoustic art music utilizing machine listening," *Computer Music Journal*, vol. 36, no. 3, p. 2012, 2012.

[84] 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.

[85] G. Widmer, S. Flossmann, and M. Grachten, "YQX plays Chopin," *AI Magazine*, vol. 30, pp. 35–48, 2009.

[86] S. Flossmann, M. Grachten, and G. Widmer, "Expressive performance with Bayesian networks and linear basis models," in *Proceedings of the Rencon Workshop Musical Performance Rendering Competition for Computer Systems(SMC-Rencon)*, 2011.

[87] S. Flossmann, M. Grachten, and G. Widmer, "Expressive performance rendering with probabilistic models," in *Guide to Computing for Expressive Music Performance*, pp. 75–98, London: Springer, 2013.

[88] D. Huron, "Interval-class content in equally tempered pitch-class sets: Common scales exhibit optimum tonal consonance," *Music Perception: An Interdisciplinary Journal*, vol. 11, no. 3, pp. 289–305, 1994.

[89] M. Schuijer, *Analyzing Atonal Music*, pp. 52–54. 2008.

[90] W. K. Leow, "NUS GET1031, Lecture 6: Algorithm design 2, part 3: Balancing optimization criteria," 2019.