Composing Bossa Nova by Evolutionary Computation

Yu-Wei Wen
Department of Computer Science
and Information Engineering
National Chung Cheng University
Chiayi 62102, Taiwan
wyw105p@cs.ccu.edu.tw

Chuan-Kang Ting
Department of Power Mechanical
Engineering
National Tsing Hua University
Hsinchu 30013, Taiwan
ckting@pme.nthu.edu.tw

***Abstract*—Development of artificial intelligence has enabled automation of diverse complicated tasks, such as planning, scheduling, and natural language processing. As one of the complicated tasks, music composition involves various aspects, e.g., melody, rhythm, harmonization, phrasing, and forms. Evolutionary composition systems ordinarily formulate music composition as an optimization problem and adopt evolutionary algorithms to deal with it. This study focuses on evolutionary composition on a special genre—Bossa Nova. The proposed system uses integer-coded genetic algorithm to generate the melody, followed by a postprocess for arranging suitable accompaniment. In particular, the genetic algorithm features evaluation rules derived from music theory for consonant melodies. The postprocess creates a treble part and a bass part which together form the chord accompaniment on the basis of Bossa Nova harmony and rhythmic patterns. This study conducts experiments for quality verification of the generated music. Experimental results show that the proposed rules are able to guide genetic algorithm for composing harmonious melodies, and the accompaniments from postprocess blend well with the generated melodies. By and large, the integration of generated melodies and accompaniments presents satisfactory Bossa Nova music.**

***Index Terms*—Evolutionary computation, music composition, genetic algorithm, Bossa Nova.**

I. Introduction

Bossa Nova is a Brazilian genre which features the fusion of harmony from jazz and rhythm patterns from samba. The melodic and harmonic parts of Bossa Nova music can be sung by vocalists or performed on piano, guitar, saxophone while the percussion parts can be performed on drum sets or traditional samba instruments. Since general Bossa Nova music sounds soothing and relaxing, it is broadly played in fine restaurants, stores, cafés, and banquet venues as background music. Background music is a type of musical works which are composed to create an ambience for affecting humans in emotions or behaviors. Instead of taking the strong focal point, background music tends to be ignorable and undistracting from its listeners. Depending on the intention of composers, background music can be relaxing, soothing, exciting, concentrating, encouraging, and so on.

Evolutionary music composition, especially composing music by genetic algorithm (GA), has been studied for decades and reported with encouraging results [1, 2, 3, 4]. For music composition, evolutionary algorithms (EAs) require designs of several basic components: solution representation, reproductive operator, and fitness function. Most of GA-based composition systems adopt solution representations and reproductive operators based on integer-coded GA, whereas their designs of fitness function vary a lot. The fitness evaluation methods can be roughly categorized into three types: interaction-based, rule-based, and learning-based evaluation. Interaction-based EAs obtain the fitness values of candidates directly from the feedback of human evaluators. Rule-based EAs codify musical knowledge into rules and then compute the fitness values by estimating the compliance level of candidates to the rules. Learning-based EAs use a model pre-trained on existing music corpuses as the fitness function. The computational time and cost are two major factors to be considered for the design of composition systems. Interaction-based and learning-based systems are found less suitable for this matter because the former is generally time-consuming and the latter requires open musical corpuses, which are usually unavailable for some genres, e.g., Bossa Nova.

This study proposes an evolutionary system to compose Bossa Nova music. The proposed system adopts rule-based evaluation, in which a set of evaluation rules are carefully compiled considering music theory and composition convention. The system consists of two parts: evolutionary melody generation and accompanying process. The first part utilizes GA for composition of monophonic melody with reference to the chord progression given by users and the evaluation rules. With the proposed rules, on which the objective of the EA depends, the GA searches for fixed-length melodies that are smooth, pleasant, and harmonic with less drama or tension. After the melody is generated, the postprocess arranges the accompaniment for the melody by fitting the pre-collected samba rhythmic patterns into the chord progression. Instead of working alone, this system allows users to interact with the composition system through assigning preferred chord progressions and accompaniment patterns.

The remainder of this paper is organized as follows: Section II reviews the related studies. Section III elaborates the design of the proposed system in detail. Section IV presents the experimental setting and results. Finally, concluding remarks are given in Section V.

II. Related Work

Researchers in artificial intelligence have been studying on designing intelligent systems for musical tasks. Automatic music generation and composition are popular research topics in computational intelligence. Among the studies of automatic music composition, evolutionary composition has a significant influence and is still growing nowadays. Evolutionary composition systems differ mainly on their designs of fitness evaluation, i.e., quality estimation of candidate music. In the early years, interactive genetic algorithm (IGA) in which fitness is evaluated by human is a rather popular option for music composition tasks. Humans can truthfully deliver their aesthetics of music to GA. However, a considerable amount of evaluation requests can easily fatigue humans and disable human evaluators from staying sensitive and concentrated.

To overcome human fatigue, research on automatic fitness evaluation pervades across various types of music composition tasks. Most of automatic fitness evaluation methods can be classified into two groups: rule-based evaluation and learning-based evaluation. Development of rule-based systems usually involves advanced knowledge of music and composition. The evaluation rules are crafted according to composition conventions, music theory, composer's experience, and sometimes personal preferences. Although rule-based systems can effectively guide evolutionary algorithms to produce decent musical works, the barrier of requiring musical knowledge prevents other researchers from taking part in extensive research. On the other hand, learning-based systems attempt to model implicit patterns from existing musical corpuses and use the obtained model as fitness function in GA. The major challenge of building a learning-based system is collecting a large enough corpus that allows learning algorithms to extract knowledge of interest. The high cost of collecting and compiling data limits the generality and applicability of learning-based fitness evaluation for composing different styles of music.

Many evolutionary composition systems have been proposed for genre-specific music, such as four-part harmony, jazz, Chinese music, fusion of genres. McIntyre [1] proposes several musical rules as scoring mechanism for evaluating candidate four-part harmony. Donnelly and Sheppard [5] also propose a set of evaluation rules examining melodic, harmonic, and rhythmic quality for four-part harmonization task. Papadopoulos and Wiggins [6] present a number of weighted rules considering patterns of adjacent notes, suspensions of neighboring chords, stressed beats, rhythmic factors; the weighted evaluation rules enable evolutionary algorithm to generate jazz melodies. Biles uses the GenJam [2], an interactive genetic algorithm, which generates jazz improvisation and further allows real-time interaction to its user. Liu [7] combines pentatonic scales and Chinese music theory which facilitates the rule-based evolutionary system for composing Chinese music. Finally, Liu and Ting [8] also show that the rule-based evolutionary system is capable of fusing different genres. Although Bossa Nova can be found in some other musical tasks, e.g., genre classification problem [9, 10], to the best of our knowledge, only one related case study [11] presents handcrafted fitness functions based on statistical metrics of a small Bossa Nova corpus. Accordingly, research of automatic composition for Bossa Nova is still immature.

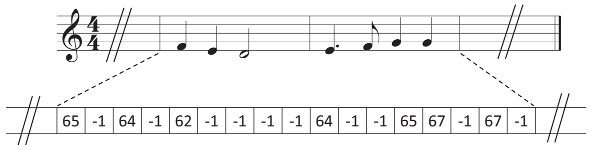


Figure 1: An example mapping between phenotype and genotype.

III. Methodology

The proposed system consists of two major parts, i.e., evolutionary melody generation and accompanying process. The first part formulates melody generation as a combinatorial optimization problem. To solve this problem, the present study adopts integer-coded GA. In addition, we propose using the rule-based fitness function to evaluate the quality of candidate melody generated by GA. The second part focuses on arrangement of appropriate accompaniment derived from common samba rhythmic patterns, for the melody obtained from GA. The following subsections elaborate the details of the GA, design of fitness evaluation, and accompanying process.

A. Genetic Algorithm

As one of the nature-inspired algorithms, GA follows Darwinian theory and mimics evolution of a species. Genetic algorithm manipulates a population of candidate solutions and searches for a solution that is fittest to the fitness function. In this study, we adopt a common representation that encodes a monophonic melody with an integer string. The chromosome length is defined by the desired length of output melody. Each integer value in a solution represents either a pitch value, ranging $[0, 127]$, which conforms musical instrument digital interface (MIDI) format, or a *tenuto*, coded as -1, which extends previous pitch. Due to the shared quality of slow flow in background music, the time period of each integer value is set as long as a quaver, a.k.a. eighth note. Figure 1 illustrates the mapping between genotype and phenotype in representation.

In the beginning of the GA, a population of candidates are sampled randomly from a biased distribution, where tenuto has higher sampling probability than pitches. The degree of bias can be customized by the user for different density of notes in resultant compositions. In the context of composing background music, sparse phrases are preferred than dense ones. As tenuto has higher chance to be sampled, long notes accordingly have higher chances to be sampled. After initialization, the random candidates are evaluated by the fitness function. Then, the GA proceeds to iterate the evolution cycle, including parent

Table I: Proposed evaluation rules.

No.	Rule description
1	Consecutive non-chord notes.
2	A hop bigger than perfect 5th.
3	A hop of dissonance (augmented 4th or minor 7th).
4	A hop of non-scale step.
5	Multiple repetition of monotone.
6	The first note of a phrase is non-chord note.
7	The last note of a phrase is not chord root note.
8	The last note of a phrase is shorter than length of half bar.

Algorithm 1 Pseudocode for rule 1

```
Input: An integer vector $\boldsymbol{x} = (x_1, ..., x_L)$ and chord note sets $C_1, ..., C_L$
 1: $count \leftarrow 0$
 2: $match \leftarrow$ false
 3: for $i \leftarrow 1$ to $L$ do
 4:     if $x_i \notin C_i$ then
 5:         if $match =$ true then
 6:             $count \leftarrow count + 1$
 7:         else
 8:             $match \leftarrow$ true
 9:         end if
10:     else
11:         $match \leftarrow$ false
12:     end if
13: end for
14: return $count$
```

Algorithm 2 Pseudocode for rule 2

```
Input: An integer vector $\boldsymbol{x} = (x_1, ..., x_L)$
1: $count \leftarrow 0$
2: for $i \leftarrow 2$ to $L$ do
3:     if $|x_i - x_{i-1}| > 7$ then        ▷ interval of perfect 5th
4:         $count \leftarrow count + 1$
5:     end if
6: end for
7: return $count$
```

Algorithm 3 Pseudocode for rule 3

```
Input: An integer vector $\boldsymbol{x} = (x_1, ..., x_L)$
1: $count \leftarrow 0$
2: for $i \leftarrow 2$ to $L$ do
3:     if $|x_i - x_{i-1}| = 6$ then   ▷ interval of augmented 4th
4:         $count \leftarrow count + 1$
5:     else if $|x_i - x_{i-1}| = 10$ then ▷ interval of minor 7th
6:         $count \leftarrow count + 1$
7:     end if
8: end for
9: return $count$
```

Algorithm 4 Pseudocode for rule 4

```
Input: An integer vector $\boldsymbol{x} = (x_1, ..., x_L)$
1: $count \leftarrow 0$
2: for $i \leftarrow 2$ to $L$ do
3:     if $|x_i - x_{i-1}| > 2$ then
4:         $count \leftarrow count + 1$
5:     end if
6: end for
7: return $count$
```

Algorithm 5 Pseudocode for rule 5

```
Input: An integer vector $\boldsymbol{x} = (x_1, ..., x_L)$
 1: $count \leftarrow 0$
 2: $repeat \leftarrow 0$
 3: for $i \leftarrow 2$ to $L$ do
 4:     if $x_i = x_{i-1}$ then
 5:         $repeat \leftarrow repeat + 1$
 6:         if $repeat \geq 4$ then
 7:             $count \leftarrow count + 1$
 8:         end if
 9:     else
10:         $repeat \leftarrow 0$
11:     end if
12: end for
13: return $count$
```

selection, crossover, mutation, fitness evaluation, and survival selection, until the termination condition is met.

In order to provide selection pressure for efficient evolution, we take 2-tournament selection as parent selection strategy and $(\mu + \lambda)$ selection as survival selection method for the GA. The 2-tournament selection chooses a candidate as parent from the one with higher fitness value between two randomly sampled candidates. The $(\mu + \lambda)$ selection compares current population and offspring jointly, and filters the better half as the population for the next generation.

On reproduction side, traditional 2-point crossover and random resetting mutation are used in our system. The 2-point crossover exchanges the subsequences of paired parents to produce two children with parental information partially fused. For each child, random resetting mutation probabilistically sets a value of the candidate to a random value to bring slight change to the candidate.

B. Fitness Evaluation

The proposed rule-based evaluation function serves as the core of evolutionary melody generation. This evaluation function is the quality metric which takes a candidate solution as input and yields a corresponding fitness value. For composing melody, we design eight rules derived from music theory. Table I gives the conceptual definitions of the proposed rules. The rules focus on detection of unpleasant melodic patterns with considering given chord progression. In other words, the evaluation function discourages GA from accepting candidates with exciting and tense patterns.

To evaluate a melody, each of the eight rules is tested respectively on the target melody. The number of times that a melody matches the rules implies its discordant level. Hence,

Algorithm 6 Pseudocode for rule 6

```
Input: An integer vector $\boldsymbol{x} = (x_1, ..., x_L)$ , chord note sets $C_1, ..., C_L$, and phrase length $L_{\text{phrase}}$
 1: $count \leftarrow 0$
 2: $i \leftarrow 1$
 3: while $i \leq L$ do
 4:     $j \leftarrow i$
 5:     while $x_j < 0$ do
 6:         $j \leftarrow j + 1$
 7:     end while
 8:     if $j \geq (i + L_{\text{phrase}})$ or $x_j \notin C_j$ then
 9:         $count \leftarrow count + 1$
10:     end if
11:     $i \leftarrow i + L_{\text{phrase}}$
12: end while
13: return $count$
```

Algorithm 7 Pseudocode for rule 7

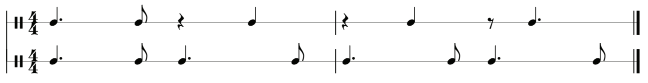
```
Input: An integer vector $\boldsymbol{x} = (x_1, ..., x_L)$ , chord roots sets $C_1^{\text{root}}, ..., C_L^{\text{root}}$, and phrase length $L_{\text{phrase}}$
 1: $count \leftarrow 0$
 2: $i \leftarrow L_{\text{phrase}}$
 3: while $i > 0$ do
 4:     $j \leftarrow i$
 5:     while $x_j < 0$ do
 6:         $j \leftarrow j - 1$
 7:     end while
 8:     if $j \leq (i - L_{\text{phrase}})$ or $x_j \notin C_j^{\text{root}}$ then
 9:         $count \leftarrow count + 1$
10:     end if
11:     $i \leftarrow i + L_{\text{phrase}}$
12: end while
13: return $count$
```

the fitness value is defined as the arithmetic negation of this number for maximization of fitness.

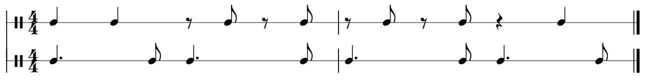
The related parameters are determined as follows: Let $\boldsymbol{X}$ be the population in genetic algorithm. The population $\boldsymbol{X}$ consists of chromosomes in which each chromosome $\boldsymbol{x}$ is a piece of melody represented by an integer vector, i.e., $x_i \in \mathbb{Z}$ $(i = 1, .., L)$, where L is the length of chromosome. Nonnegative value of x_i indicates a pitch to be played at the certain moment, while -1 stands for tenuto. A chromosome $\boldsymbol{x}$ can be segmented into n_{phrase} phrases, where $n_{\text{phrase}} = \lceil L/L_{\text{phrase}} \rceil$, and L_{phrase} is a user-defined parameter indicating the length of phrase in terms of bars. The length of a bar L_{bar} is determined by the length of atomic note and time signature. The evaluation process requires chord progression $C_1, ..., C_L$ as a part of input. A chord note set C_i consists of the MIDI numbers of all chord notes in the i^{th} moment. Algorithms 1–8 present the exact matching procedure of the proposed rules.

Algorithm 8 Pseudocode for rule 8

```
Input: An integer vector $\boldsymbol{x} = (x_1, ..., x_L)$ , phrase length $L_{\text{phrase}}$, and bar length $L_{\text{bar}}$
 1: $count \leftarrow 0$
 2: $i \leftarrow L_{\text{phrase}}$
 3: while $i > 0$ do
 4:     $j \leftarrow i$
 5:     while $x_j < 0$ do
 6:         $j \leftarrow j - 1$
 7:     end while
 8:     if $(i - j) < (L_{\text{bar}} / 2)$ then
 9:         $count \leftarrow count + 1$
10:     end if
11:     $i \leftarrow i + L_{\text{phrase}}$
12: end while
13: return $count$
```



(a) An example of Bossa Nova pattern.



(b) Another example of Bossa Nova pattern.

Figure 2: Examples of Bossa Nova rhythmic patterns.

C. Accompaniment

The accompanying process takes over the generated melody from GA and accompanies the melody by arranging chords into samba rhythm and bass patterns. Most of Bossa Nova music features the possession of a bass accompaniment which plays only the root and the fifth tones of current chord, a rhythmic pattern of samba, and a chord accompaniment of jazz harmony. To build up a chord in Bossa Nova style, the bass part switches between the root tone and the fifth tone of the chord while the treble part completes the remainder of the chord. This procedure is performed on each chord of the given chord progression to obtain the constituents of treble and bass. Then, the chords are fitted into Bossa Nova or samba rhythmic patterns forming the final accompaniment of a melody. Various conventional Bossa Nova and samba rhythmic patterns are publicly available. Figure 2 shows two patterns for examples.

Given a single chord progression, repetition of the GA can produce several different melodies. Since the chord progression remains unchanged, the accompaniment process is required only once for those melodies.

Table II: Parameter setting for GA.

Parameter	Value
Representation	Integer (ranging 2 octaves)
Chromosome length	128 (16 bars)
Population size	500
Generation	1000
Parent selection	2-tournament
Survivor selection	$\mu + \lambda$
Crossover	2-point
Crossover rate	1.0
Mutation	Random resetting
Mutation rate	1/128
Phrase length L_{phrase}	2 bars
Bar length L_{bar}	8

IV. Experimental Results

This study conducts experiments to examine the effectiveness and efficiency of the proposed system. The composition system consists of two stages which are GA for melody generation and accompanying process. The GA part is found much more time-consuming comparing to accompanying process. Accompanying process can be accomplished almost instantaneously, whereas the GA typically requires about 10 minutes. The specification of testing platform is a personal computer equipped with i7-8700 CPU, 16 GB RAM, and Windows 10. The whole composition system is programmed in Python and run with Python 3.7.3.

Chord progression plays an important role for the rule-based fitness function of GA. As Bossa Nova being the target genre of composition, the common jazz chord progressions including, but not limited to, II–V–I and I–VI–II–V are used in the experiments. The rest of parameters for GA are summarized in Table II. The resultant compositions are synthesized to audio format by Logic Pro X. Nonetheless, audio wave synthesis is also possible by other digital audio workstation (DAW) software. Some sample audio files of the results can be listened via https://soundcloud.com/user-722647244/sets/evolutionary-composition-for-bossa-nova.

Regarding the convergence of GA, Fig. 3 presents the mean best fitness value over 30 independent runs. The course of increasing fitness value shows that GA is capable of improving melodies with regarding the evaluation rules. As the evolution goes on, the melodies become more and more harmonious with the chord progression and retain gradually less discordant components. An actual example of initial melody and evolved melody are presented in Fig. 4, where the initial melody is drawn from random and thus, does not consider consonant relationship between pitch and chord, arrangement of adjacent notes, and phrasing of melody. The initial melody receives poor fitness value according to the proposed fitness evaluation. On the other hand, the evolved melody maintains harmonious concordance with the chords and also arranges neighbor tones and passing tones as connection between chord notes. In addition, the structure of the evolved melody is well formed in phrases every two bars.

The postprocess organizes the bass line and rhythmic chord accompaniment for the melody obtained from GA. Figure 5

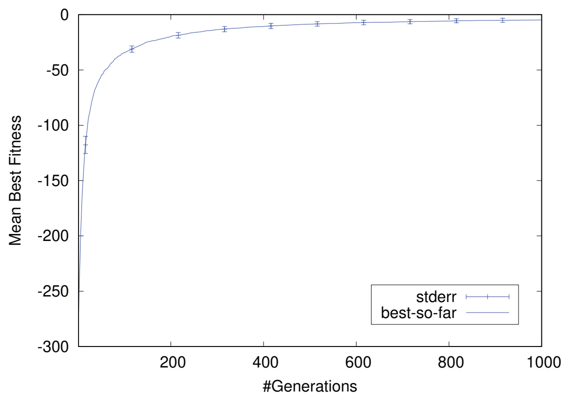
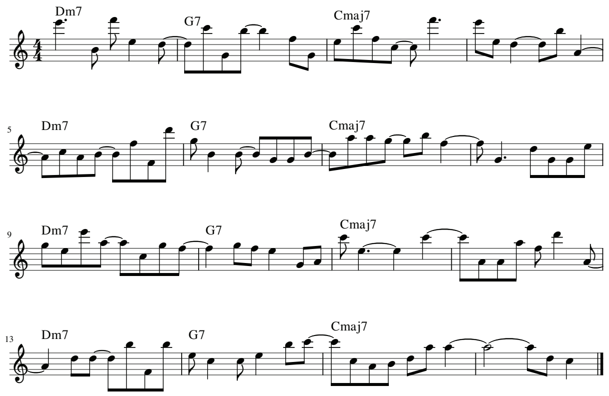
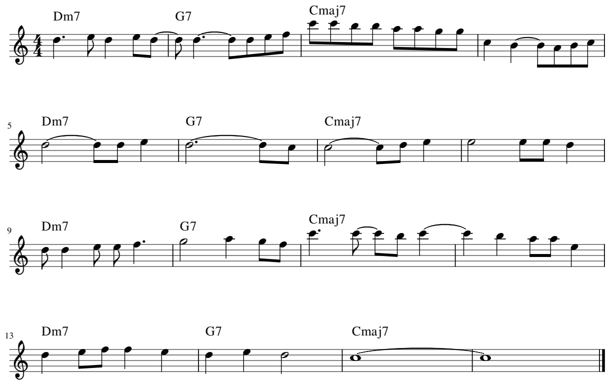


Figure 3: Convergence of mean best fitness against number of generations for the GA over 30 runs.



(a) An example of initial melody.



(b) An example of evolved melody.

Figure 4: Comparison of initial melody (top) and evolved melody (bottom).

shows a sample composition associated with the melody generated by GA and the accompaniment finalized by proposed process. The bass and treble lines form a classic Bossa Nova rhythm, which stresses the 1st, 4th, 7th, 11th, and 14th beats in every two bars. Figure 6 presents another composition obtained from the system. The compositions in Figs. 5 and 6 use II–V–I and I–VI–II–V chord progressions, respectively.

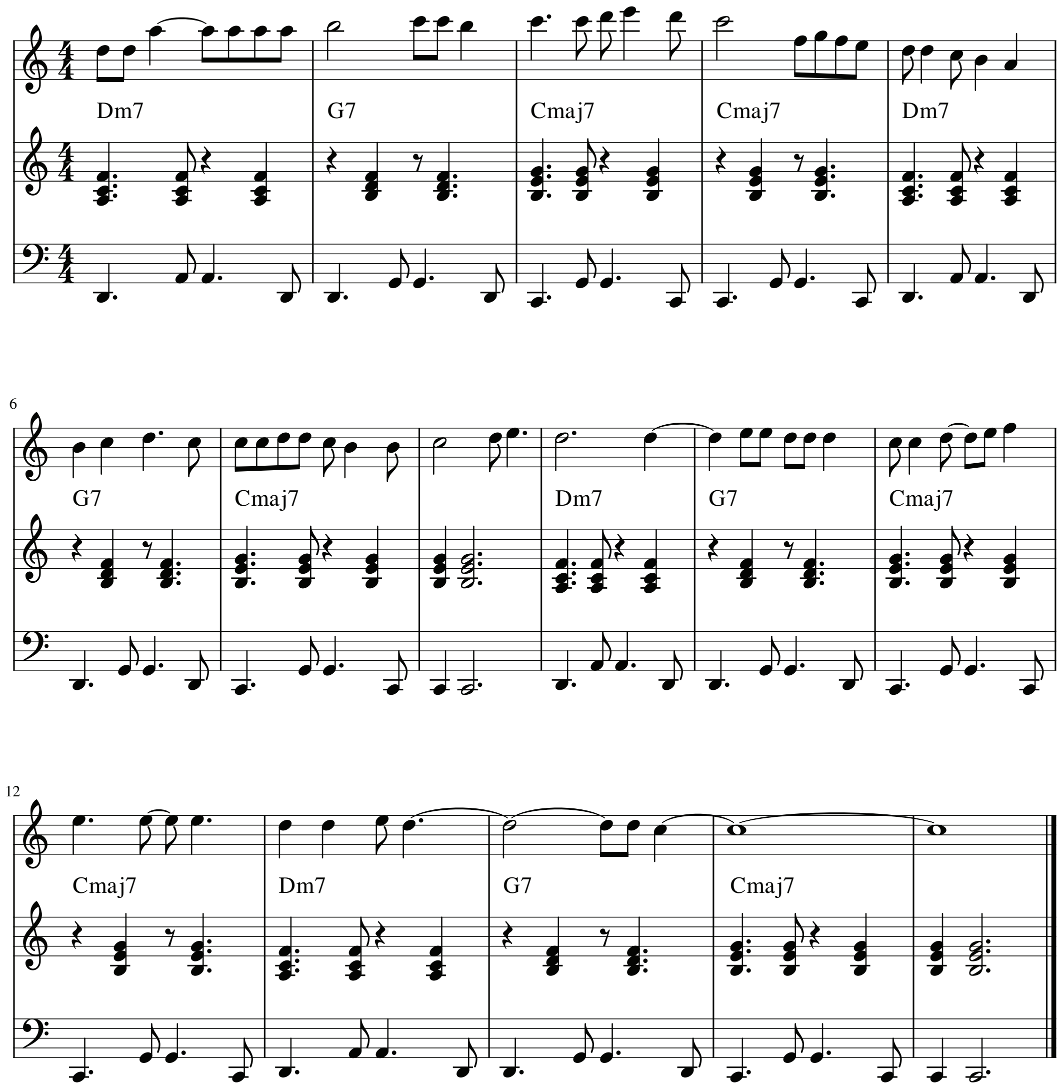


Figure 5: An example result using II–V–I chord progression.

Both results show that GA can search melodies for different chord progressions. Moreover, the different accompaniment patterns also produce different atmospheric colors for the resultant music.

V. Conclusions

The study presents a two-stage composition system that integrates GA and accompanying process for composing Bossa Nova music. Experimental results show that the proposed rule-based fitness function can effectively assist GA in generating harmonious melody over jazz chord progressions, while the accompanying process brings Bossa Nova into the compositions. In terms of computational cost, the GA is the computationally most expensive part, but the proposed composition system shows acceptable time efficiency in music generation on personal computer. The design further allows users to interact with the system by providing their favored chord progressions or accompanying rhythms.

The results demonstrate the practicability of the proposed evolutionary composition for Bossa Nova. Some potential improvements found from the experiments give the directions for future study. First, the system has no guarantee of performability on real instruments. Second, time efficiency of the system can be further increased for real-time application.

References

[1] R. A. McIntyre, "Bach in a box: the evolution of four part baroque harmony using the genetic algorithm," in *Proceedings of IEEE Conference on Evolutionary Computation (CEC)*. IEEE, 1994, pp. 852–857.

[2] J. A. Biles, "GenJam: evolution of a jazz improviser," in *Creative evolutionary systems*. Elsevier, 2002, pp. 165–187.

[3] C.-H. Liu and C.-K. Ting, "Evolutionary composition using music theory and charts," in *Proceedings of IEEE Symposium on Computational Intelligence for Creativity and Affective Computing*, 2013, pp. 63–70.

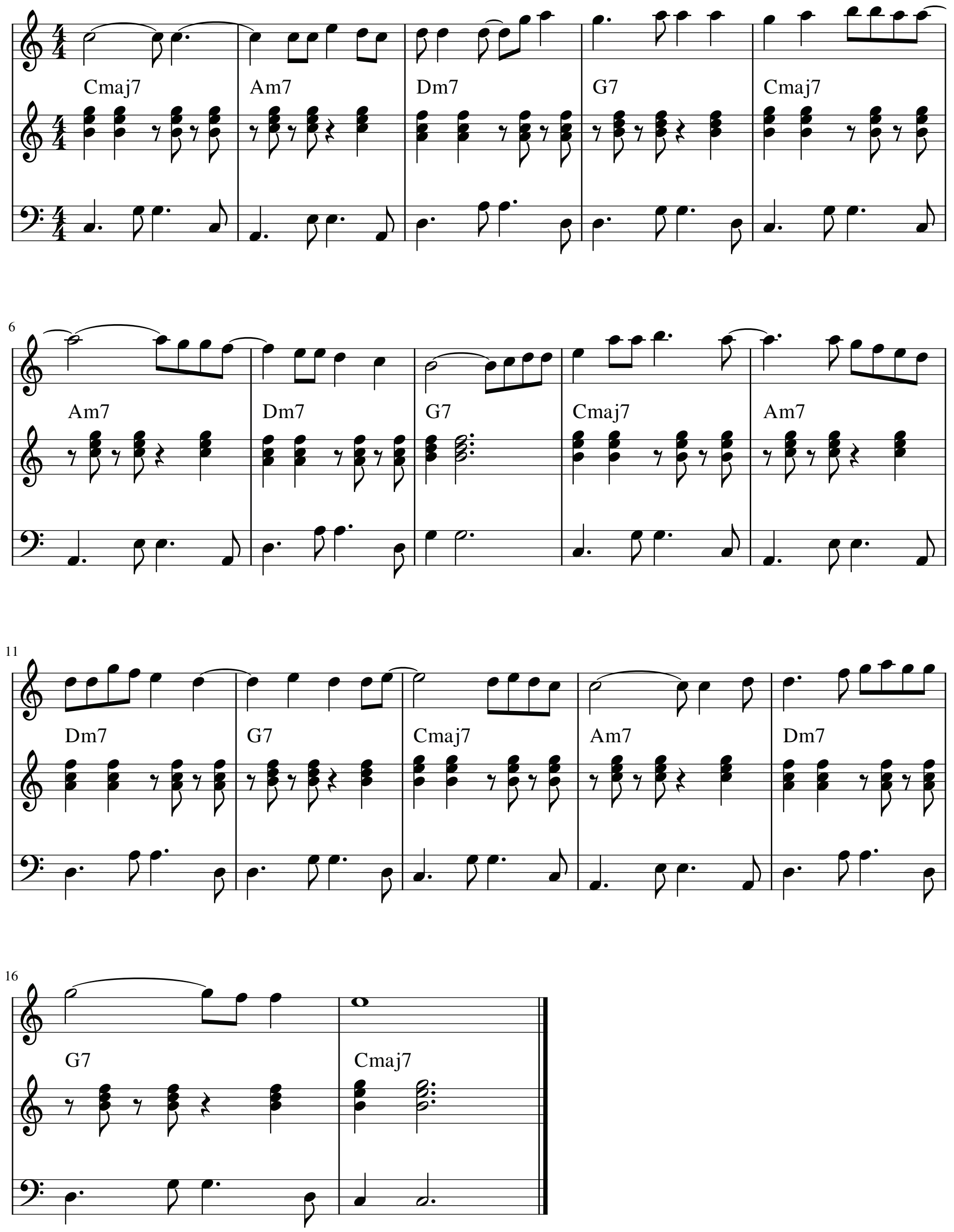


Figure 6: An example result using I–VI–II–V chord progression.

[4] ——, "Computational intelligence in music composition: a survey," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 1, no. 1, pp. 2–15, 2017.

[5] P. Donnelly and J. Sheppard, "Evolving four-part harmony using genetic algorithms," in *Proceedings of European Conference on the Applications of Evolutionary Computation*, 2011, pp. 273–282.

[6] G. Papadopoulos and G. Wiggins, "A genetic algorithm for the generation of jazz melodies," *Proceedings of STEP*, vol. 98, 1998.

[7] C.-H. Liu, "Automatic music composition using evolutionary algorithm based on music theory," Ph.D. dissertation, National Chung Cheng University, 2017.

[8] C.-H. Liu and C.-K. Ting, "Fusing Flamenco and Argentine Tango by evolutionary composition," in *Proceedings of IEEE Congress on Evolutionary Computation (CEC)*, 2017, pp. 2645–2652.

[9] A. J. H. Goulart, C. D. Maciel, R. C. Guido, K. C. S. Paulo, and I. N. da Silva, "Music genre classification based on entropy and

fractal lacunarity," in *Proceedings of IEEE International Symposium on Multimedia*, 2011, pp. 533–536.

[10] T. Völkel, J. Abeßer, C. Dittmar, and H. Großmann, "Automatic genre classification of latin american music using characteristic rhythmic patterns," in *Proceedings of the 5th Audio Mostly Conference: A Conference on Interaction with Sound*, 2010, p. 16.

[11] A. Freitas, F. GuimarÃ£es, and R. Barbosa, "Automatic evaluation methods in evolutionary music: an example with bossa melodies," 2012, pp. 458–467.

[12] C.-L. Wu, C.-H. Liu, and C.-K. Ting, "A novel genetic algorithm considering measures and phrases for generating melody," in *Proceedings of IEEE Congress on Evolutionary Computation (CEC)*, 2014, pp. 2101–2107.

[13] J. D. Fernández and F. Vico, "AI methods in algorithmic composition: a comprehensive survey," *Journal of Artificial Intelligence Research*, vol. 48, pp. 513–582, 2013.

[14] P. Casella and A. Paiva, "Magenta: an architecture for real time automatic composition of background music," in *Proceedings of International Workshop on Intelligent Virtual Agents*, 2001, pp. 224–232.

[15] 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.

[16] C.-Y. Chang and Y.-P. Chen, "Fusing creative operations into evolutionary computation for composition: from a composer's perspective," in *Proceedings of IEEE Congress on Evolutionary Computation (CEC)*, 2019, pp. 2113–2120.

[17] J. A. Biles, "Evolutionary computation for musical tasks," in *Evolutionary computer music*. Springer, 2007, pp. 28–51.

[18] S.-C. Chiu and M.-K. Shan, "Computer music composition based on discovered music patterns," in *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, vol. 5, 2006, pp. 4401–4406.

[19] C.-H. Liu and C.-K. Ting, "Polyphonic accompaniment using genetic algorithm with music theory," in *Proceedings of IEEE Congress on Evolutionary Computation (CEC)*, 2012, pp. 1–7.

[20] H. B. Lopes, F. V. C. Martins, R. T. N. Cardoso, and V. F. dos Santos, "Combining rules and proportions: a multiobjective approach to algorithmic composition," in *Proceedings of IEEE Congress on Evolutionary Computation (CEC)*, 2017, pp. 2282–2289.

[21] A. Freitas and F. Guimaraes, "Melody harmonization in evolutionary music using multiobjective genetic algorithms," in *Proceedings of the Sound and Music Computing Conference*, 2011.

[22] G. A. Wiggins, G. Papadopoulos, S. Phon-Amnuaisuk, and A. Tuson, *Evolutionary methods for musical composition*. University of Edinburgh, Department of Artificial Intelligence, 1998.

[23] S. Phon-Amnuaisuk, A. Tuson, and G. Wiggins, "Evolving musical harmonisation," in *Artificial Neural Nets and Genetic Algorithms*, 1999, pp. 229–234.

[24] M. Marques, V. Oliveira, S. Vieira, and A. C. Rosa, "Music composition using genetic evolutionary algorithms," in *Proceedings of IEEE Congress on Evolutionary Computation (CEC)*, vol. 1, 2000, pp. 714–719.

[25] R. L. De Mantaras and J. L. Arcos, "AI and music: from composition to expressive performance," *AI magazine*, vol. 23, no. 3, pp. 43–43, 2002.