# Originality and Diversity in the Artificial Evolution of Melodies

## Digital Entertainment Technologies and Arts

Alan R. R. de Freitas
Departamento de Ciência da Computação
Universidade Federal de Ouro Preto (UFOP)
Ouro Preto, Minas Gerais, Brazil
alandefreitas@gmail.com

Frederico G. Guimarães
Departamento de Engenharia Elétrica
Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte, Minas Gerais, Brazil
fredericoguimaraes@ufmg.br

## ABSTRACT

One of the greatest problems when using genetic algorithms to evolve melodies is creating an aesthetically conscious measure of fitness. In this paper, we describe a new approach with a minimum measure of fitness in which a set of good individuals is returned at the end of the process. Details about the implementation of a population of measures and some genetic operators are described in this work before an implicit way to evaluate fitness is given. We define a Takeover Matrix to measure the relationship between different generations and its compromise between originality and diversity. By means of this Takeover Matrix, the evolutionary process itself can be used as a criterion instead of using only ordinary individual measures of fitness. The results show the implications of using the proposed approach and demonstrate that the proposed algorithm is able to generate good sets of melodies. The algorithm can be used not only for developing new ideas but also to extend earlier created melodies with influence from the initial population.

## Categories and Subject Descriptors

J.5 [**Arts and Humanities**]: Music; I.2 [**Computing Methodologies**]: Artificial Intelligence

## General Terms

Algorithms, Design, Measurement

## Keywords

Evolutionary Music, Algorithmic Composition, Computer Music, Genetic Algorithms

## 1. INTRODUCTION

This paper presents an approach for algorithmic composition, a process in which patterns of composition methods, which are not always algorithmic, are explored in order to automatically produce new music. This is an interesting and stimulating process that includes challenges such as the discovery of new algorithmic standards for composition as well as it raises questions regarding personal concepts about the definition of art and the prospect of making computers to produce art pieces.

The methods for algorithmic composition are usually classified into (i) stochastic; (ii) based on rules; or (iii) based on artificial intelligence, with the latter including evolutionary methods [16]. Evolutionary computation in particular offers potential models and ideas for automatic composition. Actually, composition more than often is done by working on variations of past experiences and variations of existing themes and material. Genetic algorithms in computer music can begin with a data set of existing material and create new melodies, evolving the population by applying variations on them throughout the evolutionary process. In this sense, the evolutionary process shares some similarities with the creative process going on in composition.

However, fitness computation in most evolutionary-based systems for art and music requires aesthetic judgements, which are not easy to model and implement in the form of an algorithm. For this reason, these systems usually employ some level of interaction with the user, which provides feedback to the system about subjective aesthetic judgements, see for instance [6, 12, 17]. When dealing with static images, interactive evaluation is not much of a hindrance, because many alternatives can be presented to the user in parallel (for instance, a grid of images) [13]. Moreover, there are some studies on strategies for minimizing the number of choices presented to the user and for reducing fatigue in interactive evolutionary algorithms [19]. On the other hand, in time-based pieces, such as animation and music, interactive fitness evaluation can require significant attention from the human mentor, who is always liable to getting tired, bored, losing attention and other issues. This aspect is known in the literature as the *fitness bottleneck* [2].

Given this difficulty in basing evolutionary music systems on human evaluation, some authors have studied the development of automatic systems, which would be able to develop art and musical pieces without human intervention. Some ideas involve co-evolution [7], the development of reliable aesthetic metrics [18], and the evolution of adaptive critics [14]. In the context of music, an interesting idea to circumvent the *fitness bottleneck* is presented by Biles in [2], where he eliminates the fitness altogether from the evolu-


*GECCO'11,* July 12–16, 2011, Dublin, Ireland.

tionary system, leading to a fitness-free version of his system. His algorithm starts with a population of measures selected from a database and evolves this population using carefully designed genetic operators for some generations, without any fitness evaluation. The resulting populations represent new original melodies. Although this approach raises questions about the purity of an evolutionary algorithm[1], Biles argues that his system is still a genetic algorithm, see discussion in [2].

The number of generations required to run a fitness-free evolutionary algorithm that is evolving a melody is arbitrary and depends on the experience of the user. If the number of generations is small, the final melody resembles the initial population, without much originality. It is desirable that the final solution be original in relation to the initial melody/-ies, specially when its author is not the user of the algorithm. If the number of generations is high, a takeover[2] will occur, even with the absence of a fitness function, due to genetic drift. We call this compromise an *originality-diversity trade-off*: during the evolutionary process, originality increases but diversity decreases due to takeover, giving rise to the questions below:

- How to achieve a good compromise between originality and diversity in the artificial evolution of melodies?
- Which criterion or criteria can one use to select a melody among the many candidates generated by the evolutionary process?

In this context, we present in this paper an evolutionary algorithm for evolving melodies represented by a population of measures, without the intervention of a human mentor in the process. However, our approach differentiates from previous work by combining the following aspects:

- The melody generator proposed in this work has a population of measures and a minimum implicit fitness evaluation (see definitions in the next section). The population of each generation represents a single melody;
- The composition process is the evolutionary process itself and all individuals from all generations are candidates to be in the final melody;
- We adapt genetic operators from the literature to the details of our representation scheme;
- We propose a method to monitor takeover, and use this information to select the generation whose population will form the final melody returned by the evolutionary system;

With this approach, we aim at solving the originality-diversity trade-off in an evolutionary music system. We avoid the difficulty of defining the number of generations required to run the algorithm, by simply running it until takeover occurs. We define a Takeover Matrix to quantify the relationship between melodies in different generations. By means of this Takeover Matrix, we can decide which melody from the whole evolutionary process will be selected as the final melody, instead of using only ordinary individual fitness evaluation. This decision is based on the variation of the Takeover Matrix along the process.

We present some experimental results using the melody of the song *One note samba* [3] [9], composed by the Brazilian musicians Tom Jobim and Newton Mendonça, with interpretations available in English. A common saying has originated from the title of this song: the metaphor "too many notes samba", which means that something is exaggerated, confused, or not in harmony. To a certain extent, this captures the ideia of the trade-off we try to solve with the proposed evolutionary-based composition system, whose main goal is to create interesting and original melodies from known material. The results show it is possible to find a diverse set of melodies, being useful not only for developing new ideas but also to extend previous ideas, which are competitive with human created ideas.

## 2. CONCEPTS AND TERMINOLOGY

In this section we present some useful definitions to be used in the rest of the paper. More details about those concepts related to Music Theory can be obtained in [10].

**Measures:** A measure is a segment of time in Western music defined by a given number of beats of a given duration. In formal terms, a measure can be specified by the elements $(\eta_i, t_i, \delta_i)$, where $\eta_i$ represents a note, $t_i$ is its starting time, $\delta_i$ is its duration, and $i = 1, \ldots, n$ is the number of notes in the measure. In this work, each measure is an individual and if there are no notes in an individual, then it is considered a pause.

**Melody:** Melody is a linear succession of notes which is perceived as a single entity. In our system, the population of measures form a melody. Since each individual represents a measure, the main goal is not exactly the search for the best individual, but the evolution of a good population of individuals, for at the end of the process many of them can be chosen in a musically conscious way to form the final result.

**Random fitness:** A random fitness function is equivalent to a fitness-free evolutionary system. An easy way to implement this feature is to use uniform (or unbiased) selection for reproduction, with no selective pressure.

**Minimum implicit fitness:** A minimum implicit fitness function is a function that implements a minimal set of musical rules and/or constraints when analyzing a given individual. Individuals receive penalties or rewards based on the satisfaction of these rules and constraints. The problem becomes similar to a Constraint Satisfaction Problem (CSP) [11]. If too many rules are added to the fitness function, musical creativity is hindered. It is debatable whether we should apply a pre-existing set of rules or try to base algorithms on what composers really do instead [8].

**Takeover:** In evolutionary algorithms, the word takeover refers to the phenomenon in which the population collapses into copies of one or few individuals in the population [4]. Typically, the takeover time is the number

[1]By "pure evolutionary algorithm" we mean an algorithm that presents the following basic ingredients: representation, evaluation, selection and variation operators.

[2]In an evolutionary algorithm, a takeover implies that all individuals are the same, and the population presents no diversity.

[3]*Samba de uma nota só* in Portuguese.

of generations (on average) that a takeover takes to occur, and it has been used to characterize different selection methods used in evolutionary algorithms, see [5]. In this paper, individuals represent measures and the population represents a melody. In this context, takeover occurs when all measures in the population are the same, and the ability of generating new material is limited. Takeover then translates into a melody with repetitive patterns.

# 3. THE MELODY GENERATOR

The melody generator proposed in this work has a population of measures and a minimum implicit fitness evaluation. The composition process is the evolutionary process itself and all individuals from all generations are candidates to be in the final melody. This section describes the melody generator implemented in this work. We first present the representation scheme for measures and melodies, and the genetic operators adapted for this representation. The section concludes with the definition of the Takeover Matrix, used to select the final melody returned by the algorithm.

## 3.1 Representation

The delineation of what might be considered art occurs during the design of the algorithm, since a structure not foreseen by the representation scheme will not be generated. If this characterization is made by specifying rules, composition in particular genres of music becomes easier, since the search space can be greatly restricted in a convenient way.

The size of the representation of a solution may be fixed or variable, depending on its needs. Sometimes, the representation may be obvious but that does not happen often in the composition process and a poorly designed representation scheme might lead to search spaces that are hard to be explored and useless automatic composition algorithms.

In this work, an approach based on the order of the notes as shown in Table 1, with absolutely represented pitches, is used. Each row represents an event and the matrix can be easily converted into a MIDI file.

**Table 1: Representation of a solution.**

| Track | Channel | Note | Vel. | Note on | Note off |
|---|---|---|---|---|---|
| 1 | 1 | 60 | 90 | 0.0 | 0.5 |
| 1 | 1 | 62 | 95 | 0.5 | 1.0 |
| ... | ... | ... | ... | ... | ... |
| 1 | 1 | 74 | 93 | 10.5 | 11.0 |

The first two columns represent the track and channel used to perform the respective notes. All notes performed by the same instrument must use the same track and all notes performed with the same timbre must use the same channel. At the column note, a non-negative number represents a note, with 60 being middle C. Since an equal temperament scale is used, any multiple of 12 represents a C. Similarly, any multiple of 12 plus 2 represents a D. Since the notes are represented with absolute values, the genetic operators will have freedom to generate any note, contrasting with relative representation schemes based on scales. The values of velocity in column four can have 7 bits to control the intensity to perform the notes.

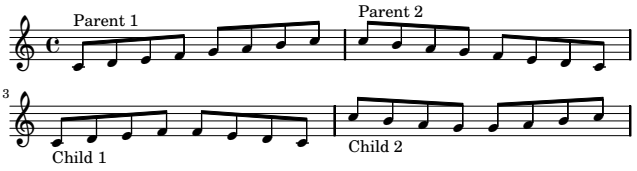


**Figure 1: Simple Musical Crossover limited to the pulses.**

Finally, it is necessary to specify when a note is performed and when it is released. The columns note on and note off use relative units of time. Considering a measure with 4 pulses and that the size of a measure is 2, there is pulse 1 beginning at time 0.0, pulse 2 at 0.5, pulse 3 at 1.0 and so on. In this work, each measure is an individual and if there are no notes in a individual, then it is considered a pause. The operators mainly work on the values in columns 3, 5 and 6, which are related to the parameters $(\eta_i, t_i, \delta_i)$ described before, with $\eta_i \in \{0, \ldots, 127\}$, $t_i \in \mathbb{R}_+$, and $\delta_i \in \mathbb{R}_+^*$. The other values are useful for having a representation scheme that can be easily exported to MIDI files. Besides, the values of velocity in the initial population can be inherited through the generations.

In order to represent the timing and durations of each note, an order-based representation scheme is used. In this way, melodies are represented by a list of pitch-timing elements $(\eta_i, t_i, \delta_i)$, creating the possibility of any value of time for $t_i$ and $\delta_i$. This model has features that are different from the ones in a position-based chromosome structure, such as in [3]. In position-based approaches, each gene represents a fraction of time and individuals have a fixed length. Each gene has a value of $\eta_i$ while the position of this gene itself defines $t_i$. Besides the notes, some additional values are needed to represent rest or hold events in order to stop or keep notes sounding. Thus, the value of $\delta_i$ can be defined.

A melody with any rhythm structure can be represented in an order-based scheme. Thus, this option seemed more useful to the authors since one of the purposes of the algorithm is to generate results with some aesthetic relation to the initial population. Besides, other implications of this choice will happen in the genetic operators, as described in the following subsection.

## 3.2 Genetic Operators

The genetic operators are applied to the initial population and lead them to new material, with an intrinsic relationship that can be perceived by the listener with the music style of their ascendents.

Genetic operators are the main tools for the creation of new individuals and they may be guided considering knowledge about the problem [15]. Guided operators correct the genotype in order not to generate absurd solutions.

Usually, the crossover point is randomly selected and that is good for exploring the potential of the population in less artistic musical tasks that are reducible to regular optimization problems. In a fitness-free generation of melodies, the crossover point should be limited to musically advantageous points, according to analyses of the phenotype. The crossover was defined with concern to the pulses of the measures, which cannot be broken, as shown in Figure 1.

Mutation is usually done by giving a new value to a gene

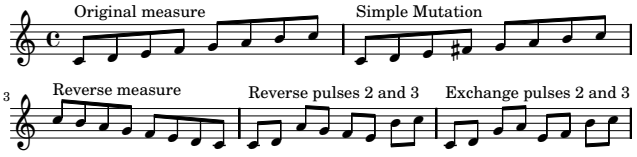


**Figure 2: Genetic Operators used in this work.**

but this simple non-guided bit-flop operation rarely works for non-optimization musical tasks. It was defined that the mutation only changes the value of a note by a difference of 2 semitones, at most, to avoid that too many large vertical intervals are randomly generated, which would be a serious problem with the simple bit-flop mutation. As in any complex problem, it makes sense to have several mutation operators occurring in parallel. Figure 2 shows all operators used in this work.

It is also common to define some different rates to apply each operator, for they may lead to undesirable results when inconvenient rates are used [1]. Those rates can be even adjusted during the process or adjusted by another evolutionary algorithm. Since no operators that change the size of the measures have been used in this work (such as copying parts or erasing notes), all operators have the same probability of being applied.

The use of an order-based representation of the melodies imply some differences in the results from those operators. In a position-based scheme, genes represent notes but also holds and rests and those events are also susceptible to change by these operators. In these cases, the application of the operators may change the genes in ways that can generate new rests and holds, modifying the number and duration of the notes. In contrast, in an order-based scheme, the timing of the notes can be more easily inherited from the parents since the elements $(\eta_i, t_i, \delta_i)$ are explicitly defined in chromosomes.

## 3.3 Minimum Implicit Fitness

One of the biggest problems for the generation of music is the definition of an adequate fitness function. Initially, one may think about basic necessities of the solution, making processes that compose music in a particular style to be more efficient but our intention is to base our process on what composers actually do [20].

Models based on rules may oversimplify the essence of music in order to make things simpler and help us define good and bad individuals. The idea is that some songs may be defined as good when they break many "rules" and the same thing has to be algorithmically foreseen in creative compositional systems. However, these models may not represent how music is usually composed.

Another possible approach is the similarity to a target song, which is useful for the creation of songs with some relationship between the initial population and the target music. In this work, the opposite idea will be employed, which is to create a new melody having the previously known melody as the initial population instead.

An approach of selection with a minimum fitness is employed in a population that comes from a predefined melody, allowing creativity to emerge from the application of genetic operators. A minimum fitness can allow us to eliminate at

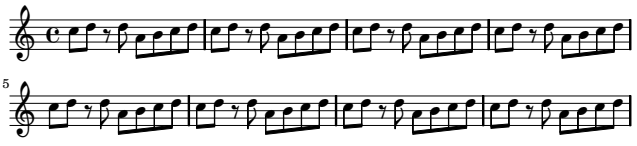

**Figure 3: Population in which a takeover has occurred.**

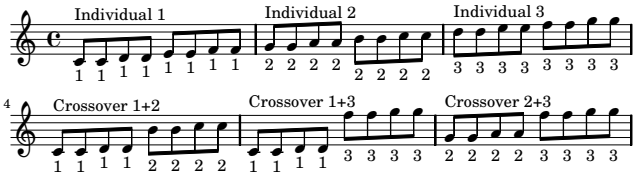


**Figure 4: Population with the initial population markers.**

least very bad solutions, which would be inaudible solutions according to the minimal set of rules embedded in the fitness function. The genetic operators must be very efficient and robust for the melodies because otherwise there is no selective pressure to force the melodies to converge to something better again.

In our experiments, two approaches of fitness were tried: (i) fitness-free or unbiased selection and (ii) a minimum fitness function that maintains the notes in the major scale of C. In the second fitness function, a penalization is given to any note out of the scale. Either way, all individuals have the same value of fitness in most cases and the selection of individuals is unbiased.

## 3.4 The Takeover Matrix

As most individuals will have the same fitness most of the time, the best individuals are not necessarily at the last generations of the algorithm. In fact, these solutions are not even desirable, since these last generations are characterized by takeover (even without the selective pressure caused by fitness functions, takeover occurs due to genetic drift). The diversity is therefore small and there are many similar chunks of melody in the population. Since the population represents the melody, takeover translates into a repetitive and boring melody, as shown in Figure 3.

The most interesting set of measures will neither be in latter generations, since there will be a convergence to takeover there, nor in initial generations, where the population is still very related to the initial melody. We call this compromise an originality-diversity trade-off. Since the advancement of the solutions from diverse to original does not happen in a linear scale, selecting a generation halfway from these extrema is also not the best choice. We need a better criterion to select a melody among the many candidates generated by the evolutionary composition process.

Originality and diversity play an important role in the definition of a good melody to be returned by the algorithm. The originality-diversity compromise between the melodies in different generations can be an alternative for quantifying how good a set of individuals is.

Given the existence of this trade-off, our approach is to halt the algorithm when the takeover happens. In order to

maintain the population far from takeover for a little longer, a holdover approach is used. After a generation, one of its individuals is chosen to be held over, which means that this individual will be also a candidate to breed in a later generation. One individual of each generation is randomly chosen to be a potential parent in the following generation. This strategy avoids a quick convergence of the algorithm.

At the end of the algorithm, we shall identify the best set of melodies but first we need to identify a takeover to do it. In order to find a takeover, each event of the population is marked with a number which identifies from which measure of the initial population that event comes from, no matter if it has been mutated. An example of the application of a crossover on a population with the initial population markers is shown in Figure 4. In this example, the 3 initial measures are the initial population and the others are results of crossover. It is interesting to note that some genetic material is lost in this process (pulses 3 and 4 from individual 1 and pulses 1 and 2 from individual 3).

A case in which it is easy to identify a takeover is when all events of the last generation come from only one individual of the first generation. Nevertheless, that does not always happen because individuals with genetic material from many initial measures may occur repeatedly during the process, forming also a takeover that has notes from many individuals while no crossover can make the next generations have notes from only one individual.

In those cases, a more general solution can be used to identify a takeover. It is to verify the origin of each event in each pulse of the measure, since those are the limited possible crossover points defined. For instance, if all notes in pulse 1 and 2 of all individuals come from the initial individual $x$ and all notes in pulse 3 and 4 come from the initial measure $y$, no crossover combination can lead to new individuals and a takeover can be equally declared. For this reason, we propose a Takeover Matrix, defined next.

**Definition 1** (Takeover Matrix)**.** *For each generation, a matrix* $\mathbf{T}$*, with dimension* $n \times p$ *is generated, where* $n$ *is the number of pulses in each measure and* $p$ *is the number of individuals. Each element* $\mathbf{T}_{ij}$ *of this matrix gives the percentage values of the origin of each event, according to the pulses and regarding individuals of the initial population.*

Table 2 shows examples of the Takeover Matrix. Table 2(a) shows $\mathbf{T}$ for an initial generation, where 25% of the notes in each pulse come from the same individual.

As the population gets close to a takeover, those percentages get close to 100%, which means all notes of that pulse come from the same individual of the initial generation. In Table 2(b), all notes from the supposed population in pulse 3 come from the individual 2 in the initial generation. In this example, the genetic material from individual 1 has been completely lost.

Mutated notes do not change the origin of a note because it would make it more difficult to identify a takeover while less musical solutions could impede the algorithm to halt. When a takeover occurs, an individual dominates each pulse of the population, and the values of the matrix do not change from one generation to another.

As it is important to have a high diversity final solution as much as they must be original in relation to the initial melody, specially when its author is not the user of the algorithm, we need now to evaluate the compromise between those objectives in the generations.

**Table 2: Takeover Matrix in Two Different Cases**

a)

| Pulse | Ind.1 | Ind.2 | Ind.3 | Ind.4 |
|---|---|---|---|---|
| 1 | 25% | 25% | 25% | 25% |
| 2 | 25% | 25% | 25% | 25% |
| 3 | 25% | 25% | 25% | 25% |
| 4 | 25% | 25% | 25% | 25% |

b)

| Pulse | Ind.1 | Ind.2 | Ind.3 | Ind.4 |
|---|---|---|---|---|
| 1 | 0% | 0% | 100% | 0% |
| 2 | 0% | 100% | 0% | 0% |
| 3 | 0% | 100% | 0% | 0% |
| 4 | 0% | 0% | 0% | 100% |

**Table 3: Calculation of a compromise value.**

| Pulse | Ind.1 | Ind.2 | Ind.3 | Ind.4 | $\sigma_i$ |
|---|---|---|---|---|---|
| 1 | 0.50 | 0.25 | 0.15 | 0.10 | 0.18 |
| 2 | 0 | 1.00 | 0 | 0 | 0.50 |
| 3 | 0.25 | 0.25 | 0.25 | 0.25 | 0 |
| 4 | 0.20 | 0.30 | 0.50 | 0.10 | 0.17 |
| Total | | | | | **0.85** |

In order to do that, during the evolutionary process, each generation (from the first one until the one in which a takeover was declared) gets a value defined as compromise value. This compromise value is defined as:

1. The standard deviation of the values of each pulse (each row in the Takeover matrix) are calculated. A high standard deviation in a pulse means that the origin of the notes of this pulse are less equally distributed than it was in the beginning of the process.

2. The standard deviations of all pulses are summed up to create the compromise value which is assigned to the current generation.

Mathematically, the compromise value $c$ is given by:

$$c = \sum_{i=1}^{n} \sigma_i = \sum_{i=1}^{n} \left[ \sqrt{\frac{1}{p} \sum_{j=1}^{p} (\mathbf{T}_{ij} - \mu)^2} \right] \qquad (1)$$

Table 3 shows the calculation of a compromise value of 0.85. The compromise value is used to monitor the process, giving an idea of the distribution of the origin of all notes in a specific generation. If the value of the accumulated deviation is close to 0, the notes are equally distributed and the current generation is probably close to the initial melody. On the other hand, a high value of accumulated deviation may indicate a takeover about to happen. Thus, it is possible to monitor the advancement of a diverse melody towards a more original melody, as this transformation does not happen linearly.

At the end, as explained above, the trade-off between diversity and originality must be dealt. With the compromise values given by the Takeover matrix, it is now possible to find a melody which is neither too close to a takeover nor too similar to the original melody not to be considered a new melody. In order to do that, the median of all compromise

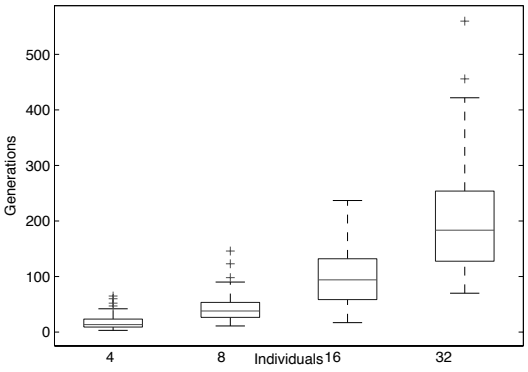


**Figure 5: Number of generations to reach takeover.**

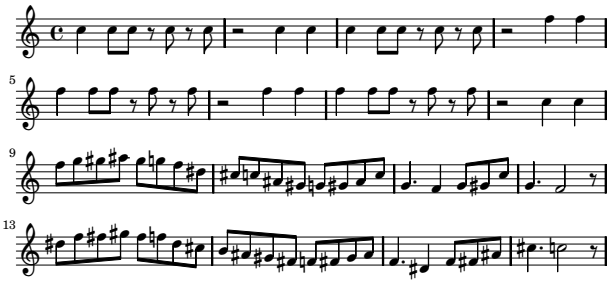

**Figure 6: 16 measures from *One Note Samba*.**

values can be used to select the generation from which the final measures will be returned as the result. Generations whose compromise values are close to the median form good candidates to be the result returned by the algorithm. We arbitrarily select the individuals in the generation with the closest value to that median to be returned as a new melody.

# 4. EXPERIMENTS AND DISCUSSION

In this section, we describe some experiments held in order to better understand the functioning of the algorithm, its compromise values, takeovers and give some examples of possible results. Different melodies were used to analyze some possibilities of the method.

As the algorithm stops running as soon as the takeover is found, the number of generations of the algorithm varies in different executions. The boxplot in Figure 5 shows the number of generations needed until takeover occurred for several different sizes of population. The algorithm was run 100 times with each population size.

Our first experiment involves the development of a new melody having 16 measures taken from *One Note Samba* as a reference for the algorithm. Those measures are shown in Figure 6. Measures 1-8 are taken from the verse while measures 9-16 are taken from the chorus. This can be considered a good song to analyze the algorithm, since the verse has a very simple melody, with only two notes and a very repetitive rhythm pattern, while the chorus, more sophisticated, has 9 out of the 12 possible notes.

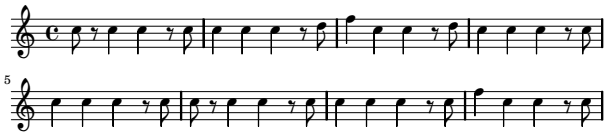

**Figure 7: A result using the verse of the song.**

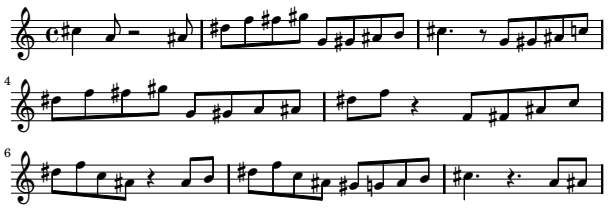

**Figure 8: A result using the chorus of the song.**

Three different experiments were held with this melody, using the (i) verse, (ii) the chorus and (iii) both of them as the initial population. Beginning with the melody here defined as the simpler one, Figure 7 shows a possible result from the algorithm which uses the first 8 measures of the melody in Figure 6.

The resulting melody shares many similarities with the initial population of measures, which are rhythmically repetitive. All notes are executed at the beginning of pulse, apart from the 4th pulse, which always have first a rest and then a note. The result has also as few notes as the initial population.

Although this first result resembles the style of the initial solution, it does not simply copy chunks of melody from the initial population and does not lead to a melody that cannot be considered a new one.

To contrast the first result with a more complex one, Figure 8 shows results that used measures 9-16 as initial population. Once more, the final result shares characteristics with the initial population such as a greater variety of notes, being many of them with the same duration (a 8th of a measure). Also the range of the notes is very similar, from D# to A# (using 19 half tones) in the initial population and from F to G# (using 15 half tones) in the resulting melody.

Comparing those first two experiments, some evidence that the final results really take after the initial population can be seen. It shows that this may be a feasible way to control the results of the algorithm, instead of defining strict rules for each particular style of music.

The influence of the rhythmic structure of the initial population can be perceived as the simplicity or complexity of the rhythm of the final result is maintained in both cases. Mutations which strongly alter the rhythmic structure of the melody may easily lead to inaudible results, specially when many subdivisions of a pulse are allowed as crossover points. Allowing only some musically relevant crossover points is one of the reasons the musicality of the results is kept.

A new song could be built up mimicking the style of the original song and parts of this new song could result from the melody generator with different parts of the original song. But something else can be done. All parts of the song can be used, forming results such as the one shown in Figure 9.

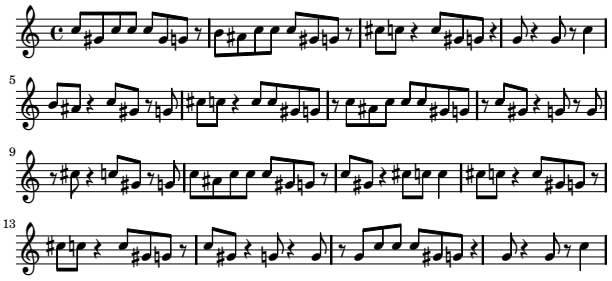

**Figure 9: A result using all measures as initial population.**

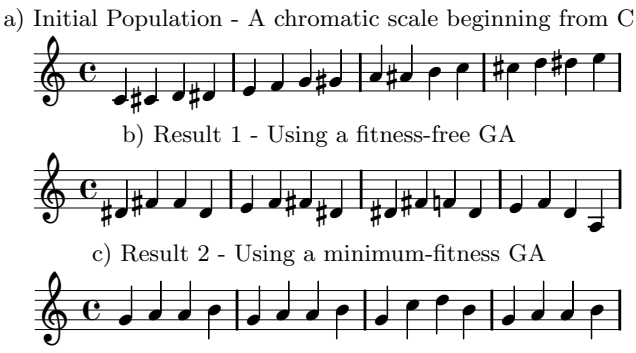


**Figure 10: Using a chromatic scale as initial population**

In those kinds of experiments, using different styles of melodies, it is important to note that the algorithm does not have the capability of distinguishing which measures belong to a different style and producing a new melody with some measures in each style. Instead, the characteristics of the different measures are likely to mix-up and form a specific intermediary "style".

The style of a song can change considerably if restrictions are imposed to individuals by any biased fitness function. That can change original characteristics of a melody by giving preferences to some certain individuals. Having demonstrative examples which are also not very musical are useful to compare the influence of the initial population of the algorithm and its operators. Figure 10 shows an experiment in which the results can exemplify this assumption.

The initial population is a simple execution of each notes 1 half tone far from each other, using always the same duration to the notes. At least for most musicians, this sequence of notes would only be used as an exercise or a warm-up before a performance. The results achieved using a fitness-free genetic algorithm can be considered as musical (or as not musical) as the chromatic scale used in the initial population. They do not respect any specific scale, similarly to the initial population, and the rhythm pattern is also the same.

On the other hand, the result using a minimum fitness approach that penalizes individuals with notes out of the scale of C made the final melody respect this aspect. Even though the same repetitive rhythm pattern is inherited from the initial population, the restriction applied to the notes made the whole melody sound far more musical.

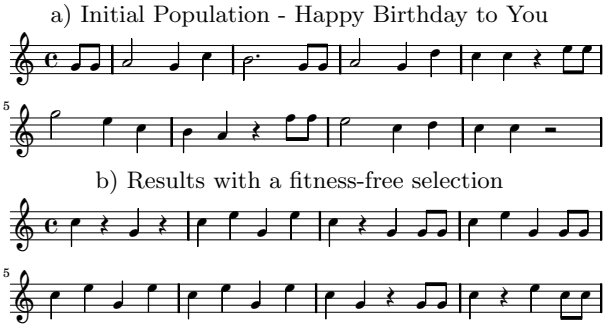


**Figure 11: Using *Happy Birthday to You* as the initial population.**

The drawback of this fitness based approach is that a strong limitation was imposed to the melodies even though the measures had the same fitness value in most generations. That led to a result that lost part of its link to the initial melody, making the algorithm loose one of its features. In this specific case, the results using a fitness-based approach may be better because there was not a good initial melody, which is the most effective way to control the results of a fitness-free solution. Thus, the fitness-based approach may have limited the link between the results and a bad melody. Even though the initial melody might not be considered good by many people, the user may have this explicit intention to create another bad melody having the first one as a reference to control the algorithm.

In cases in which the initial solutions are very musical and creative, those constraints could probably restrain this creativity of the initial solution, making it a simpler melody. In those approaches, melodies such as the one in Figure 8 would never be generated.

Also, as the algorithm has results that are related to the initial population, a fitness-free approach can be used with a good initial population that already respects the desired rules. That would result in new melodies that have the possibility of being creative but they will also be related to the initial population. In these cases, a fitness control may even remove some control over the algorithm, if the user expects to have something similar to the initial population.

For example, a simple initial melody that respects the scale of C is likely to generate another simple melody that still respects that scale in some degree. The experiment represented in Figure 11 can show evidence of that.

Figure 11 shows a simple initial solution which respects the scale of C. Applying a fitness-free selection scheme, the final solution is also a simple solution that neither has notes that last small fractions of time nor disrespect the original scale of C. All this control of the algorithm was made by controlling the initial population instead of using fitness functions that could change the style of the results.

## 5. CONCLUSIONS

Perhaps the main fact to be noticed in the development of evolutionary systems for algorithmic composition is that it is a task in which discussion in the relevant domain is strictly necessary. Any attempt to create a compositional system without discussion about musical theory cannot be

useful. The algorithm proposed in this paper shows that it is possible to create new compositions and mix known elements to create related new songs. Despite the influence from the initial population over the final result, the way the algorithm is controlled can lead to completely new results since no strict rules (such as an ordinary fitness function) are applied to the evolutionary process.

The rhythmical influence partly explains the functioning of this melody generator and how it generates music in similar genres, since it maintains a great portion of the rhythmic structure of the initial melody. That shall not be a creativity problem when there is a melody database large enough to cover many kinds of melodic structures where the generator can learn from. In fact, new creative melodies could be created from the initial ones. In the end, it may be very difficult to guess which melody the initial population represents.

In this melody generator, changes across the generations have more importance than the search for a particular result that best satisfy a specific set of rules. Thus, it is more significant to have robust musically conscious operators than fitness functions based on rules that might otherwise remove some of the creativity from the results. The takeover matrix is a good new approach devised to find a set of good solutions related to each other.

Melodies, as a result of a creative process, cannot be created based only on simple musical rules, the system must have some implicit experience about what human composers actually do. Moreover, a process to create melodies with less rules can be much easier to be implemented and used when compared to algorithms which use the artifice of a human mentor to evaluate the whole population. This algorithm can be not only useful to develop new ideas but also to extend previously created ideas which are competitive with human created ideas.

## Acknowledgments

This work was supported by the following Brazilian agencies: the National Council for Scientific and Technological Development (CNPq) and the Coordination for the Improvement of Higher Level Personnel (CAPES).

## 6. REFERENCES

[1] T. Back, F. Hoffmeister, and H.-P. Schwefel. A survey of evolution strategies. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 2–9, 1991.

[2] J. A. Biles. Autonomous GenJam: eliminating the fitness bottleneck by eliminating fitness. In *GECCO-2001 Workshop on Non-routine Design with Evolutionary Systems*, 2001.

[3] J. A. Biles. Improvizing with genetic algorithms: GenJam. In *Evolutionary Computer Music*, pages 137–169. Springer, 2007.

[4] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Natural Computing Series. Springer, 2003.

[5] D. E. Goldberg and K. Deb. A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of Genetic Algorithms*, pages 69–93. Morgan Kaufmann, 1991.

[6] J. Graf and W. Banzhaf. Interactive evolution of images. In D. B. Fogel, editor, *Proceedings of the Fourth Annual Conference on Evolutionary Programming*, pages 53–65, 1995.

[7] G. R. Greenfield. Simulated aesthetics and evolving artworks: a coevolutionary approach. *Leonardo*, 35(3):283Ű–289, 2002.

[8] B. L. Jacob. Algorithmic composition as a model of creativity. *Organised Sound*, 1(3):157–165, 1996.

[9] A. C. Jobim and N. Mendonca. Samba de uma nota só, 1959.

[10] M. Kennedy and J. Bourne. *The concise Oxford dictionary of music*. Oxford University Press, USA, 2004.

[11] V. Kumar. Algorithms for constraint-satisfaction problems: A survey. *AI Magazine*, 13(1):32–44, 1992.

[12] M. Lewis. Aesthetic evolutionary design with data flow networks. In *Proceedings of Generative Art*, Milan, Italy, 2000.

[13] M. Lewis. Evolutionary visual art and design. In P. Machado and J. Romero, editors, *The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music*, pages 3–37. Springer, 2008.

[14] P. Machado, J. Romero, M. L. Santos, A. Cardoso, and B. Manaris. Adaptive critics for evolutionary artists. In *Applications of Evolutionary Computing, EvoWorkshops 2004: EvoBIO, EvoCOMNET, EvoHOT, EvoISAP, EvoMUSART, and EvoSTOC*, volume 3005 of LNCS, pages 437–446, Coimbra, Portugal, 2004. Springer.

[15] Z. Michalewicz and D. Fogel. *How to solve it: modern heuristics*. Springer-Verlag New York Inc, 2004.

[16] E. R. Miranda and J. A. Biles, editors. *Evolutionary Computer Music*. Springer, 2007.

[17] A. Moroni, J. Manzolli, F. V. Zuben, and R. Gudwin. Vox populi: An interactive evolutionary system for algorithmic music composition. *Leonardo Music Journal*, 10:49–54, 2000.

[18] B. J. Ross, W. Ralph, and H. Zong. Evolutionary image synthesis using a model of aesthetics. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2006*, pages 1087–1094, Vancouver, BC, 2006.

[19] H. Takagi. Interactive evolutionary computation: fusion of the capabilities of EC optimization and human evaluation. *Proceedings of the IEEE*, 89(9):1275–1296, 2001.

[20] G. Wiggins, G. Papadopoulos, S. P. Amnuaisuk, and A. Tuson. Evolutionary methods for musical composition. In *Proceedings of the CASYS98 Workshop on Anticipation, Music and Cognition*, 1998.