A design of genetic encoding for breeding short musical pieces

Tatsuo Unemi
Department of Information Systems Science
Soka University
Tangi-machi 1-236, Hachiōji, Tokyo, 192-8577 Japan
unemi@iss.soka.ac.jp
http://www.intlab.soka.ac.jp/˜unemi/

Abstract

This paper describes a design of genetic encoding of short score of music for Interactive Evolutionary Computation. A genome includes three types of chromosomes, rhythm, pitch and velocity. Each chromosome is a two dimensional array of sixteen beats by 23 parts. Each element for rhythm is interpreted as indicating play, rest and continuation. Elements of pitches and velocity chromosomes are used as parameters for a recursive algorithm designed to guarantee resemblance between a parent and its children. To keep an acceptable harmony, pitches of solo parts are generated a variation of the basic melody produced through the algorithm. Elements for pitches for the parts of drums and percussion are interpreted to indicates timbre but not pitch. Combining with some optional control set up by the user, it is possible to generate a rich variety of acceptable phrases and to provide a powerful guide for breeding.

Introduction

One of the useful technologies brought through Artificial Life researches is Interactive Evolutionary Computing (IEC) (Takagi 2001), that is a promising technique to find better solutions in the domains for optimization by user's subjective criteria. Its root can be found in *Blind Watchmaker* by R. Dawkins (Dawkins 1986). Differently from ordinary methods of evolutionary computing, fitness values are not calculated automatically by the predefined evaluation function but are given by the user for each individual in some manner.

Our project named SBEAT is to try to build a support tool for composing music by means of breeding. We started from a prototype of small system SBEAT1 (Unemi & Nakada 2001), of which individual includes sixteen beats of three parts, guitar, bass and drums. We extended it to eight parts by adding four more solos and percussion part (Unemi & Senda 2001). The current newest version SBEAT3 (Unemi 2002b) can create a score including at most 23 parts using full of sixteen channels of General MIDI (Mid 1995).

One of the important issues for building an application of any type of Evolutionary Computing (EC) is to design an appropriate encoding for solution candidates on the gene as a *type* in computer program. Similarly to any kinds of object oriented programming frameworks, *type* must be defined not only by data structure but also by procedures to manipulate it. For EC applications, it is necessary to define procedures to create, transform, and evaluate genes. Creation is for setting up the initial population. Transformation corresponds to mutation, crossover and other types of genetic operations. Evaluation is for selection, which includes developmental process from genotype to phenotype.

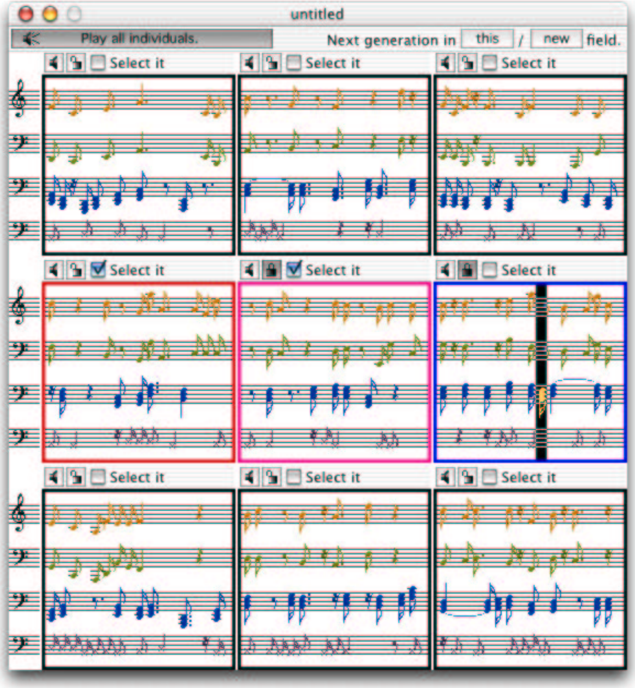


Figure 1: A field window of SBEAT3.

The content of this paper is partially overlapped with our previous papers, but it focuses on the method to encode a short musical piece including multiple part actually used in SBEAT3.

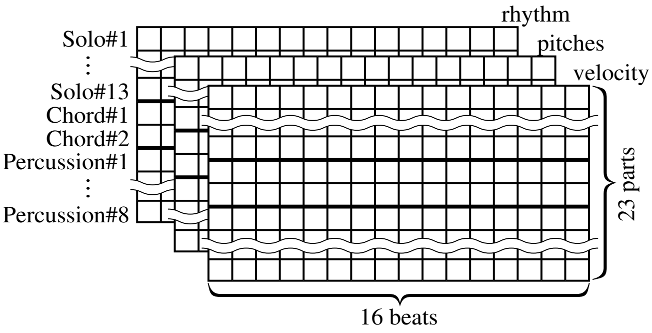


Figure 2: Structure of genotype.

Phenotype

In SBEAT3, an individual, unit of breeding target, is a bar of sixteen beats including 23 parts, thirteen solos, two chords and eight percussions. Figure 1 shows an example of field window that corresponds to a population including nine individuals. We named it *field* inspiring from a field of experimental farm for breeding agricultural products.

The score of each solo part is a sequence of single notes, that is, there is no beat in which more than one notes are allocated. The score of a chord part is a sequence of combination of three notes. The score of a percussion part is a sequence of timbre selected from one of pre-defined sets of instruments as described later. The timbre for each solo and chord part is chosen by the user from more than 128 types of instruments programmed in MIDI.

Genotype

Figure 2 shows the structure of genotype, that consists of three types of chromosomes for rhythm, pitches and velocity. Each chromosome is a two dimensional array of sixteen by 23 elements, for beats and parts. Each element involves four or five bits.

We employ mutation by bit flip in a constant possibility 5% for each bit in genotype. Relatively high mutation rate is useful when the population size is small. Crossover operation is done in a style of one point crossover, where it cuts a chromosome at randomly selected position of element boundary into two parts and exchanges them. Mutation is applied when the user selects only one individual as a parent to move to the next generation. Crossover is applied when more than one parents are selected.

Morphology

Morphology, the process to develop phenotype from genotype, plays important roles in two stages. The first is to generate feasible candidates in the initial population usually generated from random gene. It would be difficult for users to start breeding from the initial population without any individual to be a hopeful seed toward final acceptable product. The second is to guide the search process. It provides an advantage of EC comparing with other types of search algorithms, by affecting the shape of fitness landscape. In any type of EC, search points represented by individuals move on the space of genotype, but evaluation is made on the space of phenotype transformed from genotype through morphology. This characteristics enables EC to make a flexible search by keeping various aspects of building blocks.

Morphology of each type of chromosome is as follows.

```
**fill_notes**($g$, $w$) begin
    $x := (w + 1) / 2$;
    if $x \leq 1$ then $k[0] := (g[0]$ & $0111_2) + 4$;
    else **fill_notes**($g$, $x$);
    $i := x$;
    while $i < w$ do begin
        $s := k[i - x]$ + **delta**($g[i]$);
        $k[i] := \min(\max(s, 0), 15)$;
        $i := i + 1$
    end
end
```

Figure 3: Recursive algorithm to generate a *basic melody* from genotype. $k[i]$ is the $i+1$st integer for the sequence of basic melody. The function **delta**(x) returns an integer in $[-2, 2]$ from the value of argument x.

Rhythm

The value of each element in rhythm chromosome is interpreted in similar manner of GenJam (Biles 1994) as follows:

1. it continues the previous note or rest, if the most significant bit is one;

2. it rests, if the left most three bits are 011_2; and

3. it plays a new note, otherwise.

This implies the probability assignment in which continuation is 50%, rest is 12.5%, and play is 37.5%. We added restriction to prohibit continuation at the first beat position in every eight beats to produce rhythmic patterns of a relatively stable feeling. This means that rest is 12.5% and play is 87.5% at these positions.

Pitches

We employ a type of recursive algorithm shown in Figure 3 from SBEAT1 and 2 to guarantee some degree of similarity between parents and children in terms of human's feeling. Data in the chromosome for pitches in the first chord part are used as parameter values for the algorithm. It fills out all of the sixteen beats with integers from 0 to 15. The integer k_i for ith beat is calculated by adding ith gene g_i and integer $k_{i-w/2}$, where w is the greatest value of 2^j less than i, when total number of beats is 2^n. Here, j and n are integers. Concretely saying, g_1 gives the basis of the whole of the sequence, g_2 is added to every second beat $(k_2, k_4, k_6, \ldots)$, g_3 is added to every fourth beat beginning with the third beat $(k_3, k_7, k_{11}$ and $k_{15})$, g_4 is added to every fourth beat $(k_4, k_8, k_{12}$ and $k_{16})$, g_5 is added to every eighth beat beginning with the fifth beat (k_5 and k_{13}), and so on. This algorithm works even when the total number of beats is not 2^n by dividing the sequence into the appropriate small number of sub-sequences .

Each integer of the result could be interpreted as a note on the twelve pitches of the equally tempered scale, but it is mapped to a pitch in two octaves plus one step of the natural minor scale starting from A when the key is C/Am by default, to generate melody acceptable by ordinary listeners. Key note and scale are also changeable by user's settings. We call this sequence of sixteen notes as *basic melody* here.

The scores are generated by combining with the rhythm information. If the rhythm part indicates continuation or rest then the pitch information at the corresponding beat position is ignored.

Three notes are played at the same beat for the chord parts by adding two notes, three steps above and below the basic note, to the note of basic melody.

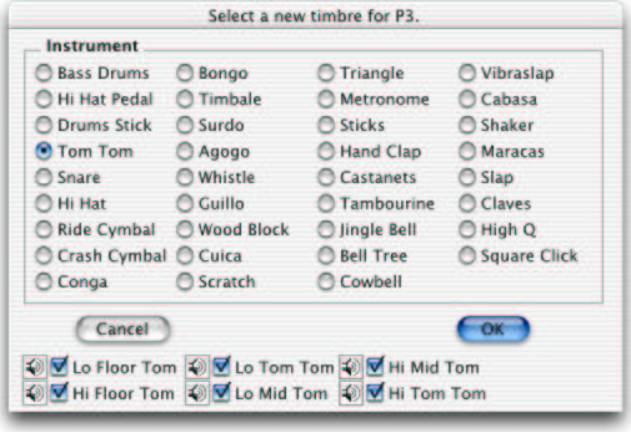
The actual pitches for solo parts are calculated by adding a value from data in pitches chromosome. The value can be -3, 0 or 3 steps to keep an acceptable harmony of multiple parts rather than discord.

Timbre for drums and percussion

For the drums and percussion parts, at most five bits are assigned for each beat to select one timbre. In older version SBEAT2, it has a drums part and a percussion part of which sets of timbre are fixed. In SBEAT3, we can use at most eight parts for drums and percussion, and can change the correspondence between parts and timbre sets.

Four sets of timbre can be for a drummer's legs and arms. Kick drums and hi-hat pedal are for legs, and a set of other drums and cymbals are for arms. For the user's convenience, sets of tom tom, snare, hi-hat, ride cymbals and crash cymbals are separately prepared. Conga, bongo, triangle, whistle and so on have more than one timbre of high and low, open and muted, or short and long. But tambourine, castanet, cowbell and so on have only one timbre in the GM sound resource. The information of pitches are ignored for these sets because there is only one candidate.

Figure 4 shows a dialog window for selecting a set of timbre being allocated for one part of drums and percussion. There are totally 35 sets listed in the dialog window. The user can listen to the actual sound of each timbre by clicking the button with speaker icon listed at the lower part of the window.



Select a new timbre for P3.

Instrument

○ Bass Drums
○ Hi Hat Pedal
○ Drums Stick
● Tom Tom
○ Snare
○ Hi Hat
○ Ride Cymbal
○ Crash Cymbal
○ Conga
○ Bongo
○ Timbale
○ Surdo
○ Agogo
○ Whistle
○ Guillo
○ Wood Block
○ Cuica
○ Scratch
○ Triangle
○ Metronome
○ Sticks
○ Hand Clap
○ Castanets
○ Tambourine
○ Jingle Bell
○ Bell Tree
○ Cowbell
○ Vibraslap
○ Cabasa
○ Shaker
○ Maracas
○ Slap
○ Claves
○ High Q
○ Square Click

☑ Lo Floor Tom ☑ Lo Tom Tom ☑ Hi Mid Tom
☑ Hi Floor Tom ☑ Lo Mid Tom ☑ Hi Tom Tom

Figure 4: Dialog window for selecting timbre set of drums and percussion part.

Velocity

The chromosomes for velocity are in the same form as for pitches, that is, each integer is represented by four bits. They are also used to give parameter values to the recursive algorithm to calculate the sequence of velocity values for each part. This process produces a type of fluctuation sequence of velocity, by which the melody can sound relatively natural as if a human player would operate the instrument.

Unit and iteration in a bar

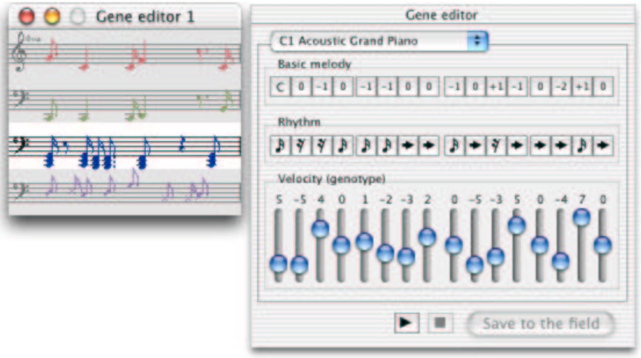
There are three options to control the developmental process that makes it possible to produce richer variety of melody.

The first is to set up the length of the unit beat, the shortest note filled in a bar. The minimal unit is sixteenth note as described before, but the user can change it to eighth or quarter independently for each part.

The second is the number of iteration in a bar. It makes the score as repetition of half or quarter length of an ordinary bar. Latter half or three quarters of the chromosomes are ignored in these cases.

The third one is the correspondence between the part of chromosome and the part of musical score. The score of parts sharing a same part of rhythm chromosome are played with same rhythm, that is, synchronous scores. If the information of pitches are also shared, these parts are played as unison.

The score information is finally translated into the tune sequence to be played by the computer. Current implementation uses Tune Player Functions of the QuickTime Music Architecture on MacOS.



C1 Acoustic Grand Piano

Basic melody

C 0 -1 0 -1 -1 0 0 -1 0 +1 -1 0 -2 +1 0

Rhythm

Velocity (genotype)

5 -5 4 0 1 -2 -3 2 0 -5 -3 5 0 -4 7 0

Figure 5: Genome editor of SBEAT3.

Partial breeding

Music has complex structure, but it is well organized, that is, any piece of ordinary music can be divided into smaller pieces and can be described by combination of common features such as rhythm, pitches, expression, timbre and so on. By this reason, we sometimes want to fix some parts or some features of bred pieces and to try alternative candidates that varies the rest of parts or features. To realize it, the *part option* dialog window includes a menu to choose the type of chromosome and of toggle buttons to indicate protection of each part. The protected parts of genotype are modified neither through mutation, crossover, nor reseting (Unemi 2002a).

Genetic operation by user

Breeding is redundant when the user knows how the genotype should be modified to obtain his/her favorite phenotype. Figure 5 shows windows called *genome editor* that allows the user to directly edit genotype of an individual. It is possible to input a known melody using the genome editor. The current implementation does not allow the user to directly input a score, but this type of *reverse* morphology might be useful to help user's task to make a variation of known tune.

Concluding remarks

A design of genetic code for music composition by IEC was described above. It can produce acceptable seeds in a initial population. In the case of the author's experimental trials, a good candidate could be found from twenty or thirty individuals randomly generated. Using partial breeding by protection of a part of genome against any genetic operation, improved melody can be obtained through a few generations of breeding process. This type of tools are useful for a beginner to compose his/her favorite and original music. And also it can be helpful for the user, who already has the knowledge of music, to get inspiration for a new melody.

In the method proposed above, genetic code is only for producing a sequence of combination of notes in a bar of multiple part. As described above, this system has several parameters the user can change to produce a variety of melody, such as assignment of instrument for each part, length of unit beat, correspondence between gene and part, and so on. In addition to them, some control parameters for playing are also changeable by the user's hand in SBEAT3, such as tempo, key note, scale, pan, volume, reverb, chorus and so on. These parameters are under control by the user so far, but it would be valuable to try to embed them into the genetic information as a part of target of breeding.

The other levels of information to construct total score of a tune is also a candidate for the target. In a shorter span, effects like slur and cressiend are useful to make more expressive play. Chord progression must be important for longer span. Multiple layered population introduced in GenJam and Conga (Tokui & Iba 2000) will be useful for combination of shorter and longer features.

SBEAT2 and 3 run on MacOS 9 and X. The executable binaries and some sample tunes in SMF and MP3 format are downloadable from:

http://www.intlab.soka.ac.jp/˜unemi/sbeat/ .

We hope many persons enjoy them.

References

Biles, J. A. 1994. Genjam: A genetic algorithm for generating jazz solos. In *ICMC '94. International Computer Music Conference*, 131–137.

Dawkins, R. 1986. *Blind Watchmaker.* Essex, UK: Longman.

Midi Manufactures Association, La Habra, CA. 1995. *The Complete MIDI 1.0 Detailed Specification.*

Takagi, H. 2001. Interactive evolutionary computation: Fusion of the capacities of ec optimization and human evaluation. *Proceedings of the IEEE* 89(9):1275–1296.

Tokui, N., and Iba, H. 2000. Music composition with interactive evolutionary computation. In *GA2000. Proceedings of the thirrd International Conference on Generative Art.*

Unemi, T., and Nakada, E. 2001. A tool for composing short music pieces by means of breeding. In *SMC '2001. Proceedings of the IEEE Conference on Systems, Man and Cybernetics*, 3458–3463.

Unemi, T., and Senda, M. 2001. A new musical tool for composition and play based on simulated breeding. In Dorin, A., ed., *Proceedings of Second Iteration*, 100–109.

Unemi, T. 2002a. Partial breeding – a method of iec for well-structured large scale target domains. In *SMC '2002. Proceedings of the IEEE Conference on Systems, Man and Cybernetics*, (to appear).

Unemi, T. 2002b. Sbeat3: A tool for multi-part music composition by simulated breeding. In *Proceedings of A-Life VIII.*