State of the Art in Procedural Music Generation using Genetic Algorithms and Rule Based GA Implementation with Attractor Waves

Rui Luz, 17984[1] and Rafael Silva 17983[2]

[1,2] Instituto Politécnico do Cávado e do Ave, Barcelos, Braga, Portugal

Abstract. In this report, a compact state of the art related to the use of genetic algorithms in music composition is presented. Additionally, the goals for implementation are laid and carried out.

Keywords: Genetic Algorithms, Automatic Music Generation.

1 Genetic Algorithms for Music Generation – Review

1.1 Introduction

The act of composing a piece of music in regarded as a creative process which encompasses objective and subjective rules. Objective rules can be pointed as the minimum necessary ones so that a given composition is considered structurally valid – music theory. Subjective rules depend on the composer ability to make a creation that is pleasant to the majority of the human listeners – or at least an intended specific audience.

1.2 Fitness Function – Different Perspectives

Given that the creation process uses the human composer's experience and sensibility – which in some form is an unknown rule set – it is assumed by some authors that, by analyzing the characteristics of the produced pieces of music spanning from several composers, underlying rules can be inferred [1], [2], [3].

Other authors start the generation process using an already available musical part, generating the main voice, bass or chords as an accompaniment of melody line [4]. Other strategy consists in using valid music sections, previously composed by humans, as starting sets [5] of the algorithm.

In he opposed spectrum, some authors advocate to allow a human to actually score some iterations as the piece evolves. Either as the central focus [6], or as an optional fitness parameter [7]. This strategy depends on the specific user, adding issues of speed and fatigue.

Other authors rely solely on music theory and harmony, using tables to rate which intervals are pleasant in a given context, such as [8].

As the fitness function has the main role in evolving the composition and given that several strategies have been attempted in its construction, there is also research effort in trying to classify how different fitness function performs [9].

1.3 Information Representation

Genetic algorithms will make the evolution of data encoded in a structure analogous to the chromosome. Traditionally, this piece of information is represented as a binary base, but it is not restricted to 2 symbols, as long as the needed processes for the evolution can be applied.

There are many dimensions linked to music composition that could be encoded, such as: number of voices, pitch, rhythm, dynamics, and tempo, among other.

Regarding the representation of the information, the more information is intended to be represented by this dataset, the denser or complex its structure may become. Some authors cover just rhythm and pitch using a single Byte [8], some use text musical notation, such as ABC[1] , as in [1] where the chromosome is encoded by a 4 X N matrix, where each element has a gene for accidents, steps (notes), octaves and durations. Other authors use a single vector [N], limited to a given scale (no accidents) and spanning 2 octaves as in the case of [7], where each element has the smallest available time representation, 0 is a pause, numbers 1 to 14 the notes starting in that scale tonic and 15 represents a prolongation of the previous element.

1.4 Evolution Process

Generally, the full length of the composition is predefined. The original sequence to be iterated is populated either with a Random [1], [7], [8], Random Bounded [2], or from melody made music segments [5]. The chosen sequence for the Genetic Algorithm is generally Selection – Crossover – Mutation, with some exceptions as in [7], where only Mutation and Selection is used. The Fitness function is used once per cycle to evaluate the results.

Each of these steps has specificities that each author uses in different ways. In [7], the selection process groups the best 1/3 of all individuals, creating new ones from mutation. All elements are regrouped and new elements rated by the fitness function, discarding the 1/3 lowest rated ones and keeping the population count. In [2], the selection process is made through tournament and in [1] elitism is used in the selection. Mutation process can have different strategies. In [7] 3 type of mutations are used, either changing the note octave, changing one tone or swapping 2 consecutive notes. The authors of [8] also have mutations for octave, but have differences for note mutation (can be any note within the octave) and a multipoint mutation, where the new symbol can be any possible set value.

---

[1] http://abcnotation.com/

2 Work Proposal

2.1 Information representation

As previously stated, the representation of information is a critical point that can generate future difficulties in the process. There is a compromise between how much detail we wish a chromosome be able to represent, and how difficult will be to deal with the amount of data that this information will generate when codified.

For this work, we chose to codify each gene as a Byte in a chromosome array, similar to what has been the strategy of [7] but instead of using a nibble, using the full Byte. As the spanning of the scale, we set it for 6 octaves, centered in A4 (440Hz), with 3 chromatic scales up and down. The Byte will not be fully used, but in case the strategy needs any update during the development process, changes could be made.

The range will span from 0 (pause), 1 to 73 (notes ascending in half tones - chromatic scale) and 255 is used to the prolongation of the previous value, as in the reference paper. 37 will be A4 (La4).

Although there is no deterrent to create a polyphonic score using a matrix with several chromosomes harmonized, we are not considering at the moment polyphony.

2.2 Fitness function strategy

For the fitness function, we will use music theory as done in many previous papers, penalizing notes that are outside the chosen scale and giving bonus to those that match. Other scenarios we must account for are:

- Bound the pitch to the range of the instrument we wish to simulate in the composition;
- Account for specific music style (modern, baroque, specific composer, etc.):
  - Penalization for lake of variation;
  - Penalization for consecutive intervals with high amplitudes;
  - Penalization for the no-occurrence of large intervals over large periods of time;
  - Rewards previous pattern repetitions;
  - Etc.
- The current representation of the chromosome defines just one bit to pause and one to extend the previous figure. In order to address the short percentage of these important figures, it will be needed – both in the starting populating stage as well as during mutation, to have them probabilistically being more often addressed. The unused range in the byte is either to be ignored, or to be used as a non-occurring mutation;

2.3 Additional Features to the Fitness Function

One of the characteristics that seem to be occurring in several music scripts is an almost oscillatory component, not always with the same characteristics. In [9] one of

the parameters the authors analyze is the occurrence of frequency components in the score – not fully detailed – through the use of furrier transforms. The understanding if in fact there is a significant occurrence of underlying frequencies and respective influence in musical pieces throughout the history, is outside the scope of this work. Even so, the effects of its presence will be evaluated in this work in the following way:

- One or more sine functions, with n periods matched to the duration of the score will be added to the fitness function;
- For each sine wave, the maximum and minimum of the functions will correspond to pitches inside the range of the simulated instrument;
- Each sine wave will be either centered of shifted upwards or downwards (keeping the pitch limits) to simulate different musical voices;
- To prevent the score notes to exactly follow the sine wave, each one will have an attraction range of a given number of spanning notes. For clarification **Fig 1** with two sine waves with 3 and 5 periods over the hypothetical score was created. The eventual attraction basins are denoted;

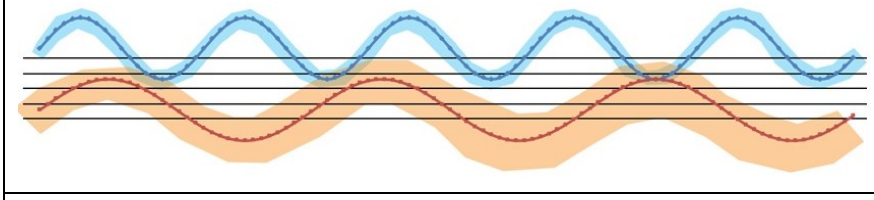

Fig. 1. Representation of basin of attraction for 2 sine waves

- The sine waves are not combined due to the fact that the intention is not to create a combined one but to, instead, have both capable of attracting the initial population to within its limits as one of the fitness factors. This can be depicted in the example created with the start of the *Bach Suite 1 BWV 1007* in **Fig 2**;

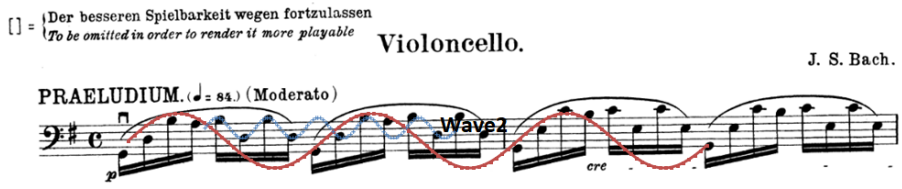

Fig. 2. Frequency components observable in Bach compositions

2.4 Selection, Mutation and Crossover Strategy

The population will be set to a pre-defined number and, to avoid the creation of a wide range of chromosomes, the selection process will ensure that the number remains constant after each iteration cycle. Mutation and Crossover are yet to be defined.

2.5 Full composition strategy

Another different approach we wish to test if in case there is still enough time, is to run the generation algorithm more than once for smaller music durations, creating a set of finalized short scores. These scores will then be assembled with the same repetition patterns we see in classical music such as [A, B, A] , [A, B, Av1, C, Av2, Bv1, A] or [A,B,A,C,A,B,A], where Av1, Av2 and Bv1 are variations of A and B respectively. To create a sense of homogeneity, the new full sequence will be iterated once more.

2.6 Initial strategy

Initially, we will start by limiting the timing component so that we can improve the melodic space. Afterwards, we will start to address the rhythmic dimension.

3 Work Evaluation Proposal

After the main parameters of the system successfully adjusted, and the system is stately converging either after a given number of iterations of after reaching a given fitness number, the work evaluation will take place.

For the final subjective evaluation proposal, all the selected compositions will be converted to MIDI and subsequently to MP3, preserving the MIDI instruments timber. The following pieces will be created:

- One without the waveform convergence;
- One With the waveform convergence;
- One with Pattern Repetition (ex: ABA);

In parallel, one MIDI Score with dynamics removed, from a real composer (but a less known composition) will be added to the set. It will be requested to classify these 4 pieces in a range from 0 to 100 on how likable they are to the listener.

4 Implementation

4.1 Notes regarding the implementation vs the intended plan

- Prolongation was set as any note above 74, as opposed to 255, but even so, the defined initial population limits was bounded up to 74;
- The musical patterns were restricted to rhythmic repetitions in the form of leitmotiv. No larger patterns as ABA or others were implemented;
- Instead of having 2 evolutions moments centered on harmony and rhythm, restricting alternately each other, the evolution time was divided in 2 moments which allow all the ratings to be customized;
- The stop criteria used was set to use a customized timer;
- For the subjective evaluation with an inquiry, just one piece of music was added to the evaluation set, with polyphony through self-harmonization, comprising a set of just 3 scores (one from AI and 2 from real authors);
- In the algorithm implementation, as much as possible the variables were implemented as byte, but due to issues with chromosome types, the *FloatingPointChromosome* function was used without floating point components.
- The selected crossover was *UniformCrossover*.

4.2 Applied Rules

As previously stated, the fitness algorithm is mainly based on music rules. The major contribution or innovations from the best of our knowledge, is the implementation of a convergence rule based on periodic oscillations.

Next the used rules will be enumerated: Note Attraction Rating; Interesting Consecutive Intervals Rating; Scale Range Rating; Pitch Range Rating; Pause and Prolongation Rating; Excessive Repetitions Rating; Interesting Repetitions Rating; Interesting Rhythmic Patterns; Interesting Harmonic Intervals Rating; Attractor Wave Rating; Alternated Measures Repetitions Rating; Leitmotif Rating; Score termination Rating; Self Harmonization Rating;

Non-Editable Rule – Hardcoded, for score Balance – For the full chromosome the None Note Percent (CNNP) is obtained. Also the minimum and maximum non not percent are defined as MinDef and MaxDef, respectively. Used range [7, 40]. The rule will penalize when outside the percentage interval. Note: ‘k’ is a tunable constant.

$$Result = (CNNP - MinDef) * (CNNP - MaxDef) * \frac{k}{CNNP},$$
$$if\ Result > 1, Result = \sqrt{MaxDef - MinDef}$$

Equation 1: Rule to bound percent of none-note figures

4.3 Music Output Result Example

For the following excerpt **Fig. 3**, convergence of the attraction waves and leitmotiv was enforced to a higher degree, with this strategy its effects are expected to be more pronounced. Having more pronounced values, it allows to more easily understand if the outcome is occurring as expected.

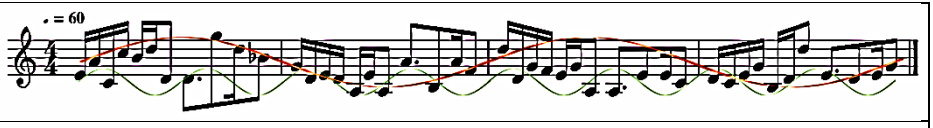

Fig. 3. Attractor Waves effect and leitmotiv

4.4 Inquire Results

To understand how likely the piece of music was to be identified as not human made, an inquiry was set up **Fig. 4**. For this purpose, apart from an music generated by with this algorithm, an excerpt of a generated music by real composers from the baroque and contemporary era (representing extremes of the music spectrum ends), were added to the set. Bach, Johann Sebastian, BWV 1087 - "14 Canons, Canon 13". Also: Stravinsky, Igor, and "Piano-Rag-Music".

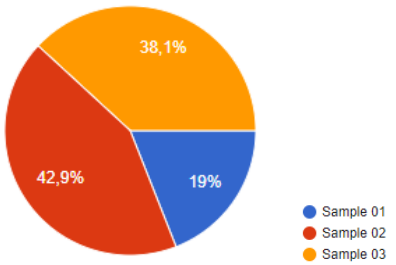

Fig. 4. Result of survey – Identify the music not composed by a human

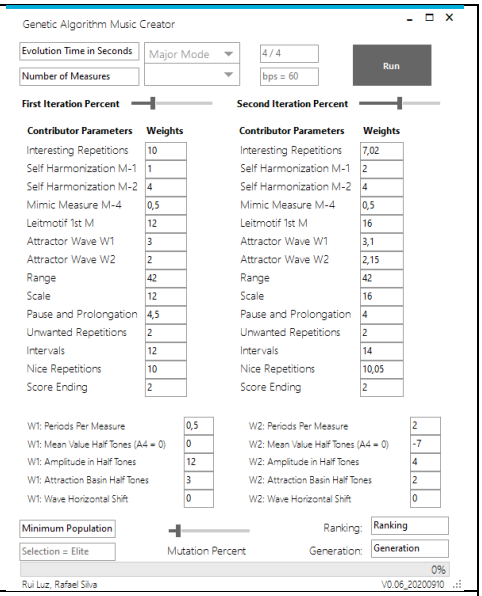

Fig. 5. Application GUI

To the question "Identify the non-human made music", there were a total of 21 answers.19% identified Bach's composition as the non-human one, 42,9% correctly identified the GA composition and 38.1% linked it to Stravinsky composition.

4.5 Application Graphical User Interface

To be able to present to the user a more practical way to interact with the program, a graphical user interface (GUI) was constructed with windows forms. It allows customizing almost all the parameters used in this GA, as can be depicted in **Fig. 5**.

5 Discussion of Results

5.1 Fitness Algorithm Results

To understand in a first approach, what rating values were most fitting, a first try-out was made by translating the start of some classical pieces into the chromosome output structure. This ideal output was then rated by the fitness algorithm. By obtaining each individual rating component for each tested sample, it was possible to predefine a start set of values to be used. The exception was the attractor wave, which was set subjectively.

Using two attraction waves, it is possible to visualize intended effect. In proximity with the wave, the notes fall into its attraction basin. When two waves are near, the effect either oscillates between the two or compromises during their proximity.

Overall, many of the generated compositions have a subjective pleasant tone to them.

5.2 Performance Tests

To evaluate the most advantageous set of configurations to obtain higher ratings from the fitness algorithm, a batch of tests were performed. Results are displayed in **Fig. 6**.

For this test, variations of three parameters with different values were performed. The selected parameters are: Mutation Probability (0.1, 0.3, and 0.7), Selection Type (Elite or Tournament) and Minimum Population (5, 20, 100, 200, and 900), generating a batch of 30 non-repeated tests.

From the results, it can be depicted that Elite Selection is performing better than Tournament Selection. Also, increasing mutation probability is deteriorating maximum attained performance. Population size is not having a detrimental contribution, and it seems that populations between 20 and 100 are having a better performance than populations with higher values.

TabName	Minimum Population	Selection Type	Mutation Probability	Max Population	Slope m, last 15% points	Max Ranking	Las 15% values
1	5	EliteSelection	0.1	44738	0,000	4149,0	38027
7	20	EliteSelection	0.1	9885	0,017	5414,3	8402
13	100	EliteSelection	0.1	1953	0,004	5170,7	1660
19	200	EliteSelection	0.1	963	0,000	5160,8	819
25	900	EliteSelection	0.1	215	2,268	5277,1	183
4	5	TournamentSelection	0.1	47624	0,088	3336,0	40480
10	20	TournamentSelection	0.1	10039	0,050	4090,8	8533
16	100	TournamentSelection	0.1	1911	0,000	4859,9	1624
22	200	TournamentSelection	0.1	929	0,000	4793,5	790
28	900	TournamentSelection	0.1	183	0,000	4863,4	156
2	5	EliteSelection	0.3	47109	0,000	5010,7	40043
8	20	EliteSelection	0.3	9585	0,214	5095,3	8147
14	100	EliteSelection	0.3	1891	0,000	5137,7	1607
20	200	EliteSelection	0.3	947	0,000	5059,5	805
26	900	EliteSelection	0.3	206	16,289	4648,2	175
5	5	TournamentSelection	0.3	45315	-0,018	3240,4	38518
11	20	TournamentSelection	0.3	9624	-0,289	4414,8	8180
17	100	TournamentSelection	0.3	1859	0,000	4316,2	1580
23	200	TournamentSelection	0.3	903	0,000	5000,5	768
29	900	TournamentSelection	0.3	179	0,000	4444,5	152
3	5	EliteSelection	0.7	43432	0,000	4142,4	36917
9	20	EliteSelection	0.7	9469	0,049	2905,4	8049
15	100	EliteSelection	0.7	1804	0,136	4840,3	1533
21	200	EliteSelection	0.7	904	0,000	4459,1	768
27	900	EliteSelection	0.7	196	0,000	3659,3	167
6	5	TournamentSelection	0.7	44967	0,000	2862,0	38222
12	20	TournamentSelection	0.7	9374	0,219	2919,1	7968
18	100	TournamentSelection	0.7	1758	0,000	4726,0	1494
24	200	TournamentSelection	0.7	826	0,000	4610,6	702
30	900	TournamentSelection	0.7	165	0,000	4210,8	140

Fig. 6. Obtained results from the variation of algorithm parameters

6 External Resources used:

MIDI Library used to translate a sequence of notes and durations into MIDI file format:
Library by Kevin Boone, 2011, Midi Sharp in Java, http://kevinboone.me/javamidi.html?i=1, translated to C# by Alberto Simões https://gitlab.com/ambs/midi-sharp

API to process genetic Algorithms: https://github.com/giacomelli/GeneticSharp

References

1. Farzaneh, M., Toroghi, R.M.: GGA-MG: Generative Genetic Algorithm for Music Generation. arXiv Prepr. arXiv2004.04687. (2020).
2. Stoltz, B., Aravind, A.: MU_PSYC: Music Psychology Enriched Genetic Algorithm. In: 2019 IEEE Congress on Evolutionary Computation (CEC). pp. 2121–2128. IEEE (2019). https://doi.org/10.1109/CEC.2019.8790099.
3. Savage, P.E., Brown, S., Sakai, E., Currie, T.E.: Statistical universals reveal the structures and functions of human music. Proc. Natl. Acad. Sci. U. S. A. 112, 8987–8992 (2015). https://doi.org/10.1073/pnas.1414495112.
4. Chien-Hung Liu, Chuan-Kang Ting: Polyphonic accompaniment using genetic algorithm with music theory. In: 2012 IEEE Congress on Evolutionary Computation. pp. 1–7. IEEE (2012). https://doi.org/10.1109/CEC.2012.6252869.
5. Vargas, F.V., Fuster, J.A., Castañón, C.B.: Artificial musical pattern generation with genetic algorithms. 2015 Latin-America Congr. Comput. Intell. LA-CCI 2015. 3–7 (2016). https://doi.org/10.1109/LA-CCI.2015.7435956.
6. Koga, S., Inoue, T., Fukumoto, M.: A proposal for intervention by user in interactive genetic algorithm for creation of music melody. Proc. - 2013 Int. Conf. Biometrics Kansei Eng. ICBAKE 2013. 129–132 (2013). https://doi.org/10.1109/ICBAKE.2013.26.
7. Matić, D.: A genetic algorithm for composing music. Yugosl. J. Oper. Res. 20, 157–177 (2010). https://doi.org/10.2298/YJOR1001157M.
8. Marques, M., Oliveira, V., Vieira, S., Rosa, A.C.: Music composition using genetic evolutionary algorithms. Proc. 2000 Congr. Evol. Comput. CEC 2000. 1, 714–719 (2000). https://doi.org/10.1109/CEC.2000.870368.
9. Csaba, S.: Towards Automated Quality Assessment Methods in Algorithmic Music Composition. Proc. - 21st Int. Symp. Symb. Numer. Algorithms Sci. Comput. SYNASC 2019. 155–158 (2019). https://doi.org/10.1109/SYNASC49474.2019.00029.