Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 120 (2017) 317–324

Procedia Computer Science

www.elsevier.com/locate/procedia

9th International Conference on Theory and Application of Soft Computing, Computing with Words and Perception, ICSCCW 2017, 24-25 August 2017, Budapest, Hungary

# Genetic algorithms for music variation on genom platform

Vsevolod Arutyunov[a]*, Alexey Averkin[b]

[a]*Dubna International University of Nature, Society, and Man, Institute of system analysis and management, Universitetskaya str., 19, Dubna, 141980, Russian Federation*

[b]*FederalResearch Centre of Computer Science and Control of Russian Academy of Sciences, Vavilova str., 40, Moscow, 119333, Russian Federation*

**Abstract**

This paper considers the problems of algorithmic music variations and modeling of musical creativity in general with a help of genetic algorithms. The proposed model of algorithmic musical variations with genetic algorithms is based on analysis and adaptation of ideas about algorithmic musical variations provided by R. Kh. Zaripov. Implementation of created model is based on Genom software, the platform for experiments in modeling of musical creativity with genetic algorithms. Experiments show that the proposed model can be used to create simple musical variations. The practical application of this system lies in interactive help for composer in creationof various musical ideas for further development. Several experimental results are shown in a form of musical scores.


Peer-review under responsibility of the scientific committee of the 9th International Conference on Theory and application of Soft Computing, Computing with Words and Perception.

*Keywords:*genetic algorithms; evolutionary music systems;search space; algorithmic music; modeling of music creativity.

## 1. Introduction

Genetic algorithms (GA) is a set of techniques, mainly used for problems of optimization. The idea behind GA is inspired by natural selection. In GA each possible solution from search space corresponds to some individual in a

* Corresponding author. Tel.: +7-985-386-0643;
*E-mail address:* sevaru@inbox.ru

1877-0509 
Peer-review under responsibility of the scientific committee of the 9th International Conference on Theory and application of Soft Computing, Computing with Words and Perception.
10.1016/j.procs.2017.11.245

population. Population evolves until it reaches the desired solution or wastes all available resources (time, material or maximum number of generations). This concept is suitable for applications with large and ill-behaved search space. Minimal criteria for use of GA is the possibility of representing solutions in a symbol form, and the presence of methods to evaluate the solutions. The specifics of modeling of music compositions and problems inherent in this area correspond the concept of GA. This set of techniques finds its application in not only algorithmic music composition like in Biles (1992) and Jacob (1995) works but also in rhythm generation Dostal (2005), FM synthesis Horner, Beauchamp and Haken (1993) and so on. One of the first works in this area was done in early 90s. It shows the results of usage of GA for computer-assisted music composition Horner and Goldberg (1991).

This paper describes the possibility of algorithmic music variation (MV) with GA. Implementation of the proposed model is based on Genom software platform. First part describes theoretical background and methods of algorithmic musical variation. Second part describes implementation details and challenges of applying GA to MV on Genom and in general.

## 2. Methods

### *2.1. Musical variations as algorithmic process*

In general, music variation is one of the composing techniques where a chosen main theme or a melody is repeated with some melodic, harmonic or rhythmic modification. There are many different forms of variations but this work considers only simple monophonic melody variations. This simplification allows building more comprehensive model of algorithmic MV preserving the main essence.

First work in modeling of musical creativity on the computer was carried out by Russian researcher R. Kh. Zaripov (1960). In his book R. Kh. Zaripov explores the possibility of modeling of musical variations on a computer. The system he created is able to generate musical variations for short monophonic melodies. The choice of monophonic melodies as a research material is not accidental; this limitation significantly reduces the space of search and provides more vivid and definite results for analysis. Melody is one of the most important parts of musical composition as the author says. Melody is a monophonic expression of a musical idea and it is its main building block Zaripov (1983).

The key idea behind his work is the transfer of invariant structure. Invariant structure (invariants) always relates to some specific form or shape and it is a set of immutable characteristics. There is no one universal rule to choose invariants from a music artifact. One melody may have different sets of invariants. Other important part of the transfer is transformants — procedures, which are applied to a melody to transform it without changing its invariants. Different transformation can break invariants of some sets and do not change them in others.

Before the creation of those key components the author analyses the melody and decomposes its structure. Firstly, melody is a sequence of tones in major or minor scale that is divided into smaller structures (phrase, sentence, chain).

His system implements following top-level algorithm:

- Random generator suggests new note
- The predefined rules determine whether the suggested note is appropriate or not
- Previous steps are repeated until the entire composition is created

This process, as explained by R. Kh. Zaripov, imitates a composer at work. As a rule a composer does not know the next note and the composing process seems random, however it actually obeys certain rules. The researcher tries to encode these possible rules manually. The rules are provided via matrix of transitions.

Most systems encode domain knowledge via parameters. Some of them are constants and others variables. For the sake of simplicity and due to the limit knowledge of the object of study some variable parameters became constants. For example, the system in the current paper uses fixed timeline encoding and only diatonic scale. These assumptions free more resources for more meaningful aspects of the domain area. R. Kh. Zaripov uses the following parameters in his system:

- Pitch range
- Measure structure in bars
- Time signature
- Count of anacrusis notes
- Scale
- Distribution of intervals

The experimenter identifies two types of parameters, which are vital and not vital. Vital parameters do not vary in one set of composition, which was chosen by some criteria like genre. Basic example of a vital parameter for dance music can be time signature of the value 4/4.

System does not have clear distinction between analysis and synthesis. Instead, knowledge of domain area are obtained by the author and encoded directly into algorithm itself. In addition, the lack of any measure of variation similarity makes the analysis of results difficult. Such information may be useful to determine the most appropriate parameters to achieve the desired similarity. Moreover, there is not a lot of available documentation about the model and experimental results. Original implementation of model, Variation, was written in algol-60 programming language back in 1980 and the source code is not available at present time. All these facts make it impossible to experiment with the system R. Kh. Zaripov created and to analyze the results he obtained. In addition, it is quite difficult to extend this model without changing a lot of its internal logic.

Therefore, further development of ideas about algorithmic MV requires new model, which can cover the weaknesses described previously. To achieve this goal it was decided to create a more general model for algorithmic MV using GA. The desired model should have a set of basic tools for experiments in MV and the potential for further extension for more advanced tasks. Genom, a platform for GA experiments in musical creativity, created by the author of the current paper, covers some of these requirements, but it is not specifically designed for MV. The next part of the paper is devoted to this exact problem.

### *2.2. Genetic algorithms for music variations*

As it was discussed in previous works on GA and algorithmic music, GA can show worthy results in the area of musical creativity. MV as a subgroup of musical creativity also can be represented as a problem of optimization and consequently can be transformed into a task for GA. Search space here is a set of all available variations of a given melody. However, that “ideal” version of solution, which is an original melody, already exists so the algorithm should define overly similar ones as not optimal. There are several ways to solve this problem, which will be discussed later.

The creation method of MV with GA is based on two concepts in this research. The first one is deriving of invariants from original piece of musical composition. The second — synthesis of a new musical artifact with the derived characteristics. These two steps are directly related to GA operators: fitness, mutation, and crossover. When invariants are derived from the source material, selected parameters and the choice of specific fitness executors represent concrete attract points in a search space. Space dimensions depend on the choice of fitness executors as well. An experimenter can change the attractors and create various sets of dimension in search space by applying different parameters and choosing executors. This idea is borrowed from the VoxPopuli system by Moroni, Manzolli, Von Zuben and Gudwin (2000), which creates similar attractors and uses them for fitness function calculation in real time. Another way of changing the course of experiments is to apply different parameters to transformation executors (mutation, crossover). Various transformation options can lead to different results and significantly change the experiment execution time. As mentioned above, one melody may have many invariant structures. Different variations can also relate to one invariant structure with some degree of similarity.

One of the goals of this paper is to transfer the ideas of Rh. K. Zaripov to GA.To achieve this it is necessary to implement all of the key components within the GA paradigm. In the original worktransformants perform the procedure of transformation and guaranties that this transformation does not break invariant structure. However, transformations in GA model do not provide such a guaranty. Mutations and crossovers transform genes while fitness functions judge suitability. Therefore, transformants easily fit the model of mutations and can be even more varied because they do not need to saveinvariants.

The fitness function is always one of the bottlenecks in non-generalizable fields like music creation and variations as well. Towsey, Brown, Wright and Diederich (2001) suggest three main types of fitness functions, which are:

- Rule based fitness function
- Human fitness function
- Learned fitness function

Rule based fitness functions are most various and common ones. They also can be divided into the following subgroups: the first group includes the rules based on music theory, which are extracted from existed studies of this area and encoded into systems; the second group consists of the rules based on harmonics and other physic characteristics of audio; the thirdgroup is based on the comparison of statistical data extracted from existed musical compositions and used by the following systems by Matic (2010), Papadopoulos and Wiggins (1998) and Ozcan and Ercal (2007).The last one uses different metrics to compare similarity between newly composed and reference compositions also used by Alfonceca, Cebrian and Ortega (2006) and Grachten, Arcos and Lopez de Mantaras (2004).

Human fitness function requires an interaction step for judgment and it is used in interactive GA by Biles (1998). Despite the fact it successfully functions in GenJam, this extra step consumes a valuable amount of time.

There are several possible solutions to this problem and learned fitness function is one of them like in works by Johanson and Poli (1998) or Anthony and Burton (1998). This approach mostly uses artificial neural networks to judge the quality of musical composition. Nevertheless, it requires a full process of teaching the network and does not always produce reasonably well results which showen by Biles (1996).

Another interesting approach occurs in one of the GenJam versions, where Biles (2001) refuses to use any kind of fitness function and relies only on domain specific crossovers and mutations as well as on preselected proper source material.

As told above, R. Kh. Zaripov encodes invariant safety in his transformations, so most of the borrowed ones already satisfy these criteria. Moreover, his analysis indirectly suggests some methods to check this safety, which means that at least we can evaluate the variations.

Most of new fitness functions and mutations introduced by MV have the same structure. Firstly, all of them extract masks from the artifacts. Secondly, they compare these masks via function-comparators or perform mutation. So decision was made to create special configurable factories for these functions. These factories receive mask extractor function and executor function as input (compare for fitness, modification for mutations). For example, all fitness function factories compare two masks and return value between zero and one. Further, it is possible to combine multiple fitness functions to achieve the desired results.This structure leads to the idea of multiple representation for different genetic operators (GO). Simple fitness functions and mutations can operate on a single layerof raw gene material. However, more complex GO, requirethe layer of abstraction for more convenient interaction.

### *2.3. Genom platform*

Genom (acronym for *Gene of Music*) is a software platform designed for experiments in algorithmic music compositions with GA.The system by Arutyunov and Averkin (2016) is a web-based application and it provides the following range of features:

- The ability to perform and monitor experiments, their current state and progress in real time, see Fig. 1
- The opportunity to add custom GO such as crossover, fitness, mutations and user interfaces to setup options in a main application view
- The interface to setup the options for GA
- The interface to setup the options for GO
- The ability to create, update and remove reference music compositions
- The ability to view musical compositions as musical scores
- The ability to listen to a musical composition

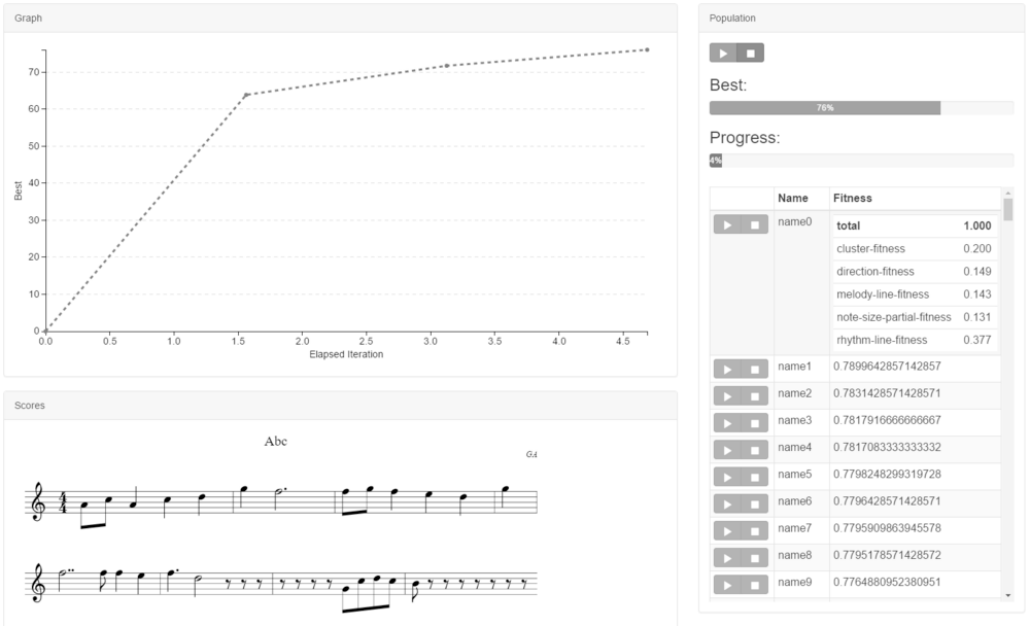


Fig.1. Experiment view. Population table, scores, graph of progress.

The system consists of two main components: graphical user interface to setup, execute and analyze results of experiments and calculation module, which controls actual GA computations. Calculation module uses GO providers, which in turns use executor component to calculate results.

Work with the system consists of the following steps:

- The user chooses the reference individual (musical composition)
- The user sets GA/GO options and starts an experiment
- GUI module sends request to calculation module
- Calculation module asks providers for executors according to passed options
- Calculation module starts computation and sends back the results of its progress
- During the computation or at the end of it, the user can analyze current results in a form of musical scores as well as listen to them

The initial goals behind the creation of the system were to provide an extensible platform for various ranges of experiments in modeling of musical creativity with GA. However, the ability to add plugins for various GO opens the possibilities to achieve creation of MV in Genom.

### *2.4. Musical variations with genetic algorithms*

Genom platform implements modification part of *transformants* of R. Kh. Zaripov by adding a list of new mutations. The following list represents the most used variation mutations:

- Simple repetition with parameters: group size, count of groups
- Change of rhythm
- Ornament variation
- Transposing with parameters: group size, transposition size
- Combination of previous

The last one isavailable in system without any modifications. Genom allows users to set weight for different mutations.The challenging part is to create component to judge variation. During conversion to GA, *transformants* should not break *invariant*s, but mutations/crossovers in GA in most cases operate on the low level and do not have these domain knowledge. To accomplish this algorithm it should be determined how much invariant structure changed. As shown in the current works — there is no definite way to determine the quality of musical compositions also shown by Wiggins, Papadopoulos, Phon-Amnuaisuk and Tuson (1998), so there is no "right" way of determining the quality of variation as well.

The idea of fitness function in Genom is to extract domain logic from algorithm itself and put it in subprograms and parameters. Therefore, during the work on implementation of features for MV in Genomthe necessity of separation of concerns became more prominentand for this reason different levels of representation of musical artifacts were created. As shown in previous works the most low-level representation is a collection of symbols, genes. These collections in Genom use a form that quite similar to GenJam'sGJNF, where numbers from 1 to 14 are two octaves of diatonic scale, 0 is a pause, and -1 is a hold of previous note or pause. The next one, the more domain-aware level looks like a collection of note objects, each of which contains the information of its step in a diatonic scale as well as a note duration. Third level is more abstract then the previous and it is represented by results of analyzer work. Analyzers can derive different information from artifacts, for example, analyzer can derive rhythmic patterns or frequency distribution for bars or a whole melody. Then GO such as fitness or mutation can use these representations for their needs.

Generally, the proposed algorithm for musical variation in Genom has some similarities with other systems such as Alice or EMI by Cope(2000) and ACSSM by Chan, Potter and Schubert (2006). These systems use four main steps in their process of creation musical artifact in given style. The steps are:

- Preparation
- Analysis
- Deconstruction
- Reconstruction

These steps are not that complex in Genom, but can still be distinguished. For now, first step requires manual encoding of the melody in special notation similar to GJNF in GenJam.Analysis step is not performed by Genom itself but rather by researchers who implements plugins for fitness/mutation/crossover calculation. Nonetheless, deconstruction and reconstruction are done by system and performed on each iteration of algorithm.

Several improvements have been made to Genom to provide it with the ability to carry MV experiments. Many of them are made without changing the internal system structure but some of them inevitably affected it. The previous fitness provider that could choose only one executor have been replaced by the new ones. There are two main types of them:

- Probability execution provider $R_p$:

$$R_p(p) = \{r_i, \sum_{k=1}^{i-1} w_k < p \leq \sum_{k=1}^{i+1} w_k \quad (1)$$

- Composite execution provider $R_c$:

$$R_c = \sum_{k=1}^{n} r_k w_k \quad (2)$$

Where $r_k$ — result of executor in index *k* and $w_k$ — is a weight on this executor.

Probability execution provider ($R_p$) is used in crossover and mutation to provide more variety of genes.Each request for execution results in new, pseudo random choice of executor. Composite execution provider is used in fitness, due to its precise control over calculation. In addition, these partial results of pairs weight $w_k$ and result $r_k$ are used in reports of experiments for more accurate analysis.

In addition, as shown in the work of Merz(2014)ad hocsolutions play important role in most automatic music creation systems. This is another big topic to discuss which will be covered briefly. Genom is not algorithmically pure system as well as work of David Cope EMI and John Biles' GenJam. It has valuable amount of knowledge encoded directly in algorithm. Some of GO in Genom are domain aware which make them tight coupled to domain itself. This is a well-known fact and Genom does not make claim to be a system to solve general problems,but it is a highly specific system that covers only a little section of area, but in a more detailed way.Moreover, it is designed not for a total automatization, but for being interactive helper for composers. The degree of creativity depends mostly on the custom GO, so better results can be achieved by composers with programming skills.

## 3. Results

Some results obtained by means of Genomare given below. For the demonstration purposes, an unknown simple melody created by the author of Genom was used. This choice is supposed to prevent listeners from any bias. The melody has several variations and each of them is made by Genom.As it can be seen from the results, each melody variation has its own unique characteristics, which are provided by the choice of the input algorithm parameters. The latest version of the software is accessible by the link: https://github.com/sevaru/ga-app5.

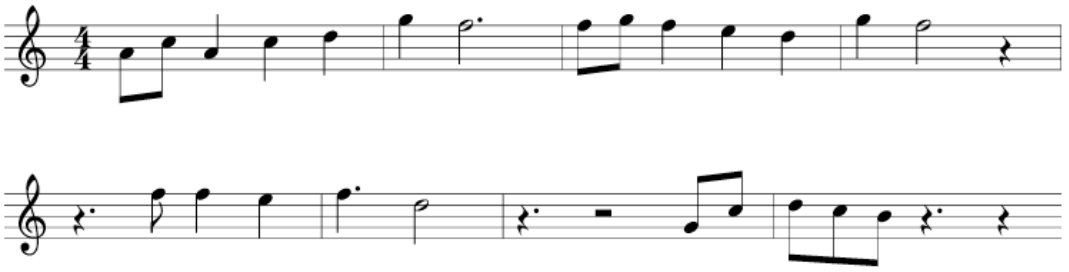

Fig. 2. Original melody.

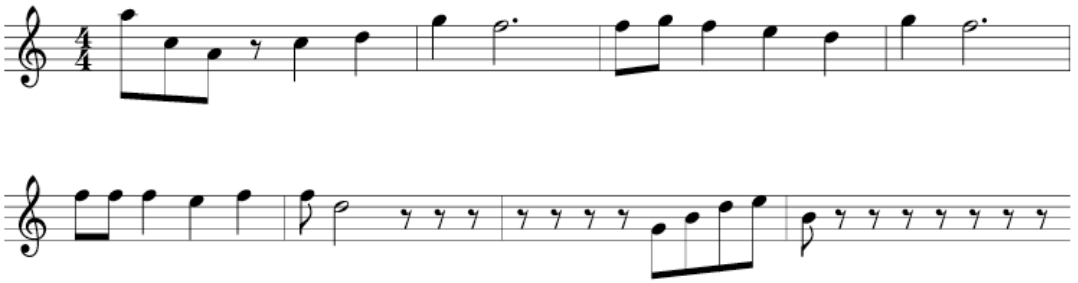

Fig.3. Melodyvariation.

Genomdoesnotproduce "ideal" solutionsandsometimes listeners can hear that variations appear a little bit uncoordinated.This problem mostly stems from the fact that mutations work with low-level elements.To be more precise, system works with discrete, time sliced pieces of notes. As one of the solutions to this problem, the usage of more hierarchical structures on each layer of an algorithm can be proposed. Melodies can have more dimensions. The encoding can be more complex — like multichromosome. Therefore, mutations and other GO can be applicable to several layers. Another idea is to bring more context to this process, add current chord context to each note or group of notes to provide more ways for calculation of quality of the solutions.

## 4. Conclusions

Basing on the obtained results it can be seen, that the current version of Genom already can be used for simple variations. Furthermore, Genom as a platform shows its extension points and receives new features that provide support for MV experiments with GA. One of the important things that this work shows — R. Kh. Zaripov's ideas of algorithmic variations still relevant today even after more than quarter of century.Despite of all these benefits there are still a huge room for improvement. For now, Genom encodes in itself a noticeable part of ad hoc solutions as well as it works only with monophonic melodies.

## References

Alfonceca, M., Cebrian, M., Ortega, A. 2006. A Fitness Function for Computer-Generated Music using Genetic Algorithms. WSEAS Transactions on Information Science and Applications. 3(3):518-525.

Anthony, R., Burton, A. 1998. Hybrid Neuro Genetic Pattern Evolution System Applied to Musical Composition. PhD Thesis. University of Surrey, School of Electronic Engineering, Information Technology and Mathematics. Guildford, Surrey.

Arutyunov, V., Averkin, A. 2016. Modelirovaniemuzykal'nogotvorchestvapripomoshchigeneticheskihalgoritmovnabazeplatformy Genom [Modeling Musical Creativity Using Genetic Algorithms Based on GENOM platform]. "Programmnyeproduktyisistemy" [Software & Systems]. 2(114):201-208. doi 10.15827/0236-235X.114.201-208

Biles, J. 1994. A Genetic Algorithm for Generating Jazz Solos. Proceedings of the International Computer Music Conference. San Francisco. 131-137.

Biles, J. 1998. Interactive GenJam: Interacting Real-time Performance with a Genetics Algorithm. Proceedings of the International Computer Music Conference, San Francisco. 232-235.

Biles, J. 2001. Autonomous GenJam: Eliminating the Fitness Bottleneck by Eliminating Fitness. Proceedings of the Genetic and Evolutionary Computation Conference Workshop Program. San Francisco.

Biles, J., Anderson, P., Loggi, L. 1996. Neural Network Fitness Functions for a Musical IGA. Proceedings of International ICSC Symposium in Intelligent Industrial Automation (IIA'96) and Soft Computation (SOCO'96). 39-44.

Chan, M., Potter J., Schubert, E. 2006. Improving Algorithmic Music Composition with Machine Learning. Ninth International Conference on Music Perception and Cognition. Alma Mater Studiorum University of Bologna. 1848-1854.

Cope, D. 2000. The Algorithmic Composer. Wisconsin: A-R Editions. 302 p.

Dostal, M. 2005. Genetic Algorithms as a Model of Musical Creativity – on Generating of a Human-Like Rhythmic Accompaniment. Computing and Informatics. 24(3):321-340.

Grachten, M., Arcos, J., López de Mántaras, R.. 2004. Melodic similarity: Looking for a Good Abstraction Level. In Proceedings of the International Conference on Music Information Retrieval (ISMIR). Barcelona.

Horner, A., Beauchamp, J., and Haken, L. 1993. Machine tongues XVI: Genetic Algorithms and Their Application to FM Matching Synthesis. Computer Music Journal. 17(4):17–29.

Horner, A., Goldberg, D. 1991. Genetic Algorithms and Computer-Assisted Music Composition. International Conference on Genetic Algorithms and Their Applications. 337-441.

Jacob, B. 1995. Composing with Genetic Algorithms. Proceedings of the International Computer Music Conference (ICMC'95). Banff Alberta. 452-455.

Johanson, A., Poli, R. 1998. GP-Music: An Interactive Genetic Programming System for Music Generation with Automated Fitness Raters. Genetic Programming 1998: Proceedings of the Third Annual Conference. 181-186.

Matic, D. 2010. A Genetic Algorithm for Composing Music. Yugoslav Journal of Operations Research. 20(1):157-177.

Merz, E.. 2014. Implications of Ad Hoc Artificial Intelligence in Music. Musical Metacreation: Papers from the AIIDE Workshop. 35-39.

Moroni, A., Manzolli, J., Von Zuben F., Gudwin, R. 200. VoxPopuli: An Interactive. Evolutionary System for Algorithmic Music Composition. Leonardo Music Journal. 10:49-54.

Ozcan, E., Ercal, R. 2007. A Genetic Algorithm for Generating Improvised Music. Eights International Conference, Evolution Artificielle, EA. Tours. 266-277. doi 10.1007/978-3-540-79305-2_23

Papadopoulos, G., Wiggins, G. 1998. A Genetic Algorithm for the Generation of Jazz Melodies. Proceedings of Finnish Artificial Intelligence Conference (STeP'98). Jyvaskyla.

Roads, C. 1996. The Computer Music Tutorial. Cambridge: The MIT Press. — 1234 p.

Towsey, M., Brown, A., Wright, S., Diederich, J. 2001. Towards Melodic Extension Using Genetic Algorithms. Educational Technology & Society. 4(2).

Wiggins, G., Papadopoulos, G., Phon-amnuaisuk, S., and Tuson, A. 1998. Evolutionary Methods for Musical Composition. Proceedings of the Second International Conference (CASYS'98). Liege.

Zaripov, R. 1960. On Algorithmic Description of Process of Music Composition. Proceedings of the USSR Academy of Sciences, 1960. 132(6):1283-1286.

Zaripov, R. 1983. MashinnyjPoiskVariantovPriModelirovaniiTvorcheskogoProcessa. Moscow: Science. 231 p.