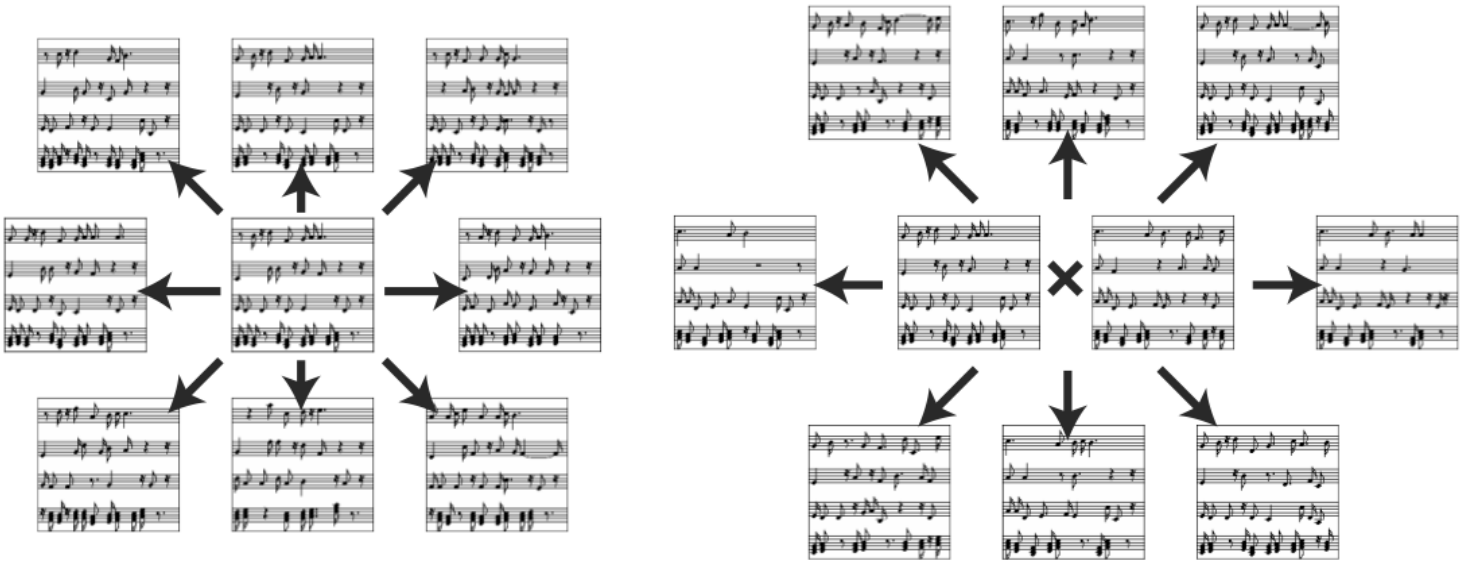
UNIVERSITY OF GOTHENBURG



Generative Music Composition Software Systems Using Biologically Inspired Algorithms: A Systematic Literature Review

Master of Science Thesis in the Master Degree Programme
Software Engineering and Management

KEREM PARLAKGÜMÜŞ

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
Göteborg, Sweden, January 2014

Generative Music Composition Software Systems Using Biologically Inspired Algorithms: A Systematic Literature Review

KEREM PARLAKGÜMÜŞ

Examiner: LARS PARETO, MIROSLAW STARON
Supervisor: PALLE DAHLSTEDT

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden, January 2014

Abstract

My original contribution to knowledge is to examine existing work for methods and approaches used, main functionalities, benefits and limitations of 30 Generative Music Composition Software Systems (GMCSS) by performing a systematic literature review. GMCSS are created by using biologically inspired algorithms. While analyzing methods and approaches of 30 software systems, biologically inspired methods, types of GMCSS, and programming languages and environments are considered. Moreover, main functionalities, benefits and limitations of 30 software systems are explained in detail. The systematic literature review revealed that using biologically inspired algorithms for GMCSS is still immature. This systematic review is based on 30 articles, journals, and conference proceedings specified after a multistage selection process.

Acknowledgements

I would like to show my appreciation and gratitude to all those that have been involved in the whole process of this thesis. Therefore, I would like to thank Associate Professor Sven-Arne Andreasson at the Department of Computer Science and Engineering, in the Division of Software Engineering at Chalmers University of Technology for sharing all his valuable, crucial information about Computer Science and Software Engineering and conduction of research and his life experiences and for helping me to focus on my studies, Lars Pareto, senior lecturer and researcher at the Department of Computer Science and Engineering, in the Division of Software Engineering at Chalmers University of Technology, for motivating me to concentrate on my master thesis studies, Associate Professor Palle Dahlstedt, in Computer-Aided Creativity in Department of Applied Information Technology and Lecturer in Composition and Artistic Director of the Lindblad Studios Academy of Music and Drama, for offering me the opportunity to work on this topic and for the providing information and input, Assistant Professor Çiğdem Turhan, at Faculty of Engineering, Department of Software Engineering at Atilim University, for motivating me to generate interest in computer science, endearing software engineering to me, affecting my life in a very positive way, understanding me and my inner world, being kind-hearted, and debonair to all her students, Joakim Jahlmar, language support tutor from the Department of Languages and Literatures at the University of Gothenburg, for his valuable help regarding linguistic aspects of my work, Oliver Bown, James McDermott, Daniel Brown and Amy Hoover, very successful and experienced researchers who share their highly valuable information about their studies, Murat Açar, Can Peşkersoy, Waseem Soomro and Ulf Eliasson, the greatest friends and brilliant software engineers who make my life meaningful for always supporting me in all respects without hesitating, my faithful and warm hearted parents Kamuran & Ekrem Parlakgümüş and my helpful and generous brothers Eren and Alper Parlakgümüş for believing in the accuracy of everything I do and supporting me with all their spirits.

Gothenburg, November 2013

Kerem Parlakgümüş

Contents

1 Introduction

The subject of this master thesis is a systematic literature review (SLR) of Generative Music Composition Software Systems (GMCSS) using biologically inspired algorithms. The goal of an SLR is to asses, analyze and combine evidence from primary research studies using a rigorous and explicit method. It has been widely performed in sociology and medicine and is an important methodology of Evidence Based Software Engineering (EBSE) [ZBT11]. Because the study subject is related to computer generated music, generative music and evolutionary algorithms (EAs), background information about these phenomena is provided below.

1.1 Background

The first computer generated music composition, the Illiac Suite for String Quartet, was created by Leonard Isaacson and Lejaren Hiller in 1956 [Hil81]. This was the first successful attempt recorded. Following this success, in 1956 Geoff Hill generated digital computer music on the CSIRAC computer. In 1962, Iannis Xenakis implemented software which generated numerical data in FORTRAN. Another successful attempt was David Cope's implementation of computer programs which analyze works of other composers, like Mozart, to produce new works in a similar style. In the 1970s, Gottfried Michael Koenig implemented a program which generated sounds of the composition as well as the score [Doo04]. Digital computer music led to the emergence of generative music. The term generative music refers to music which is changing and ever-different and is created by a system [IO12]. For a better understanding of the generative music phenomenon, another definition made by Schulz, Geiger, and Reckter is presented: It is music generated by an external system which modifies musical parameters [SGR09]. Generative music is specified by an algorithm, a set of rules, a set of processes and a mapping from randomness [SMOB10]. One way of creating generative music is EAs. They are created by utilization of modern genetics and natural evolution, where the main idea is that random variations in inheritable genetic traits within a population cause some individuals to survive while others do not. After analyzing and working within the field, Dahlstedt describes the relationship between EAs and music [Dah09b]:

> *EAs provide a way to search and explore spaces of possible solutions to a problem, especially when an exact form of a solution is not known, or when the goal is simply unknown.*

As technological advances increase, generative music composition software systems develop in different formats. In this thesis, the field of GMCSS which uses biologically inspired algorithms is systematically reviewed. The research study has been performed and the literature on GMCSS has been reviewed in order to show the current status, problems, and solutions with regards to implementation. Moreover, it is worth mentioning that the scope of the systematic review is limited by the boundaries of the

software engineering discipline. No SLR of GMCSS has been performed so far, which proves the uniqueness of the study. The aim of the systematic review is to capture the current methods and approaches used in implementation of generative music composition software systems, main functionalities, benefits and limitations of generative music composition software systems, limitations of EAs which are used in generative music composition software systems, and to clarify opportunities and needs for future research. There have been several proposals and suggested solutions for ways of improving generative musical composition software systems such as the temporary storage created for Patch Mutator by Dahlstedt to solve the problem of fitness bottleneck, that is, manual evaluation of the user takes too much time for many individuals. Users need to hear each individual and set a score for those individuals such as good or bad [Bil07]. Proposals for solutions in the field confuse those people who intend to implement GMCSS as well as people whose research field is GMCSS. This results in **problems concerning accessing main functionalities, benefits and limitations of GMCSS** as well as **the methods and approaches used for GMCSS in the field**. There is a need for a map instead of a survey. An SLR answers this need and it was what motivated me to systematically evaluate the status of the field of GMCSS from a software engineering perspective, and provide guidance for future progress.

Some of the generative music composition software systems can be found in repositories where research papers, articles and books exist or on App Store, Google Play, Internet, and the websites of researchers who implement generative music composition software systems.

This thesis does not address the GMCSS which are implemented with the technique of Markov chains, generative musical tension modeling, or the systems which were implemented before 2000. It focuses on the music systems that are produced by biologically inspired algorithms.

The remainder of the thesis is structured as follows. Section 2 provides a detailed definition of the research methodology for the systematic literature review, the research questions, and the processes of the methodology; i.e. data retrieval, study selection, data extraction, and data synthesis used in the systematic literature review. In Section 3, the results of the systematic review are introduced. A discussion is presented in Section 4. The conclusions based on the findings are presented in Section 5.

1.2 Related Work

In order to get a better insight in the field of systematic reviews, it is worth stating an important and interesting publication in advance, which is SLR of SLRs in software engineering [KPB+09] by Kitchenham et al. It is a tertiary study whereas my study is a secondary study. The literature was reviewed to evaluate the impact of SLRs which are the suggested EBSE method for aggregating evidence. The results of the SLR are based on 20 selected included studies which were all SLRs. Eight of the included studies address research trends rather than technique evaluation, 7 of the included studies address cost estimation. The quality of the included studies was fair with only

3 scoring less than 2 out of 4. The results mean that the topic areas are limited. European researchers are the leading exponents of SLRs. The SLRs who made cost estimation show the value of EBSE for synthesizing evidence.

Since there have been no systematic review studies on generative music composition software systems, I have searched for the literature including review or survey-like studies in this field. Fernandez and Vico [FV13] performed a comprehensive survey about algorithmic composition which is the total or partial automation of the process of music composition by using computers. Artificial Intelligence (AI) Methods in Algorithmic Composition: A Comprehensive Survey presents a thorough view of the area for researchers in AI which is used for algorithmic composition, consisting of probabilistic methods, grammatical representations, symbolic rule-based systems, neural networks, evolutionary algorithms and constraint programming. Roads [Roa85] reviewed the AI methods for music composition. He surveyed the literature about AI techniques for music making. This survey mentions the need for AI techniques in 4 areas of musical research: performance, digital sound, composition and music theory. Second section surveys state of the art AI and music. The discussion focuses on generative music composition software systems in the four areas of research just mentioned. The final section analyses how AI methods of learning and planning can be used to develop the knowledge base. The other algorithmic composition review discusses the subject from a point of view related to music theory [Col09], or from subjective assessment of an author [Pro89], [In 95]. Some of the reviews gives an comprehensive view of a particular method for algorithmic composition as Miranda and Anders [AM11] do for constraint programming or as Santos et. al [SAD+00] do for methods of evolutionary algorithms.

Burraston and Edmonds et al. [BE05] reviewed the literature about sonic art applications and electronic music of Cellular Automata (CA) in a technical and historical context. Computational and algorithmic processes have been interesting for artists to create culture of generative electronic art for many years. Evolving sequences and patterns are very important for the creative artist working temporally and spatially within a chosen medium. The results of the study are shown in Figure 1. Thirteen CA music systems were reviewed based on the main features such as **dimensions**, **cells**, **states** and **rules** which are related to architectures of the CA system used and **CA**, **seeding** which identifies the number of CA within each system, and their seeding mechanisms, and **MIDI** and **audio** which indicate particular domains of application.

	Dimensions	Cells	States	Rules	CA	Seeding	MIDI	Audio
Beyls CA Explorer 2.0	1	12	2 - 8	3,5 or 7 neighbour	9	Random, user	X	
Ca (CCM)	1	32	2	K2r1	1	Random		X
CAM (Millen)	1,2 and 3	1 and 2D up to 100	2	K2r2, Life, 3D Life	1	Random, Life forms, user	X	
CAM OSX (Millen)	1	35 to 700 (in 5 fixed steps)	2 - 3	K2r1, k2r2, totalistic k3r1	1	Random, user, single seed	X	
CAMUS / CAMUS 3D	2 and 3	40x40 12x12x12	2 (Life) 2 - 16 (DCS)	Life,Demon Cyclic Space, 3D Life	2	Random, user	X	
CAW (York)	1	User specified	2	K2r1, user	1	Random, user	X	X
Chaosynth	2	Up to 999x999	>=3	Chemical Oscillator	1	Random		X
FractMus 2000	1	128 to 512	2	K2r1	Up to 16	User	X	
Harmony Seeker	1	50 (practical limit)	4 (practical limit)	User	Multiple as batch	Random	X	
LASy	1	512 per wave	4096	Function / Time based	Multiple	Waveform		X
Martin	1 and 2	24x24	NA	R-D	Multiple networked	Equilibrium	X	X
SoftStep	1 and 2	Up to 32	2	Life, HiLife and k2r1	Multiple free assigned	Random, user	X	
Virtual Waves	1	32	2	4 cell addition	1	Random, user		X

Figure 1: A Comparison of CA Music Systems.

[***BE05***]

Due to the lack of related works, I partitioned my topic into smaller facets. Since this thesis has two major points of view, namely **generative music composition software systems** and **biologically inspired algorithms**, I have found some related literature on these topics separately. First, surveys/reviews of generative music/music generation/music software are as follows:

Papadopoulos and Wiggins [PW99] surveyed the use of different AI methods for algorithmic composition, showed their advantages and disadvantages, examined some important general issues and proposed desirable future prospects. The AI methods are categorized as mathematical models, knowledge based systems, grammars, evolutionary methods, systems learning behavior of the user and hybrid systems. Jin [Jin05] pre-

sented a comprehensive survey of the research on fitness approximation in evolutionary computation. Approximation levels, approximate model management schemes, model construction techniques are analyzed. In conclusion, open questions and interesting issues in the field are presented. Srinivas, Patnaik and Lalit [SP94] presented the science and art of GAs and survey current issues in GA practice and theory. The analogy between GAs and the search processes in nature is drawn. Then the genetic algorithm that Holland introduced in 1975 and the workings of GAs are described. After a survey of techniques suggested as improvements to Holland's GA, the advances in GA theory related to modeling, dynamics, and deception is surveyed.

Second, surveys/reviews of biologically inspired algorithms are as follows: Tang and Wu [TW09] reviewed the development of biologically inspired algorithms, regarding EAs and swarm intelligence, and the newly emerged bacterial foraging algorithms. The review is classified to these three areas of biologically inspired algorithms, and their common features and functionalities are discussed. To be able to determine the roots and analyze the variants of various biologically inspired algorithms, the discussion is extended to pinpoint the difference between each algorithm, particularly by arguing their biological background. The review focuses on the background of the original studies of biologically inspired algorithms and their principles and applications. Nakano [Nak11] reviewed the emerging interdisciplinary area of biologically inspired network systems. These systems grouped into 2 classes: 1) in silico, 2) vitro/vivo network systems. For each class, background knowledge is given regarding the biological mechanisms, systems, or materials used. Conclusions are given with the goal of specifying future challanges for each class of biologically inspired network systems.

2 Research Methodology

In this section, the description, the main purpose, the advantages and disadvantages, the design, and the execution of an SLR are presented. An SLR is a tool for analyzing, assessing and interpreting all available research concerning a specific research question. The main reasons for undertaking an SLR are summarizing existing evidence related to a treatment or technology, analyzing gaps in current research for suggesting areas for further studies and giving a background for positioning new research activities. Most of the research starts with a literature review of some sort, but unless a literature review is accurate and objective, it has got little scientific value. An SLR synthesizes existing work which is **objective** and **fair**. The main purpose of conducting an SLR is to provide an objective assessment of a research study by using a reliable, accurate method. Advantages of an SLR are that results of research are not biased, provide information about effects of some phenomena across empirical methods, and have the possibility of combining data using meta-analytic techniques. However, it has one disadvantage. In fact, it requires more effort than traditional literature reviews [KC07]. For fine details of an SLR, [BKB+07] is referred.

In the light of this information considering the two research questions of this thesis

study, SLR was selected as the most appropriate research method. In this chapter, the way of conducting this SLR is demonstrated. In this thesis, the systematic review is divided into planning, realization, and reporting activities. The systematic review took nearly one year to complete. An outline of the review which presents the activities in the SLR is shown in Appendix E.

A systematic review protocol was developed. The protocol included research questions, search strategy, evaluation strategy and two exclusion criteria, data extraction and synthesis methods. The protocol was reassessed and filtered in iterations. The review was conducted in four steps; i.e. data collection, study selection, data extraction and data synthesis.

2.1 Research Questions

The results of this thesis will be beneficial for software developers who want to implement GMCSS (RQ1) and for the researchers who want to have an access to main functionalities, benefits and limitations of 30 GMCSS (RQ2).

Table 1 - Research Questions for the SLR

ID	Question	Aim
RQ1	What methods and approaches have been used in implementation of generative music composition software systems so far?	There are a lot of proposals and solutions in the field. From a software engineering perspective, the best and the most appropriate method to implement GMCSS must be found out and highlighted.
RQ2	What is possible to know from the literature about main functionalities, benefits, and limitations of generative music composition software systems?	Making musical composition software systems by using biologically inspired algorithms is a new field. Because the field is very young, it has only been practiced for a short time; however, it has been evolving. The main goals for conducting the systematic review for this research question are to identify the limitations of generative music composition software systems, and to present and summarize the existing practices. This research question is specified for presenting the current state-of-the-art GMCSS and hence the systematic review should create a connection for research in the future and the software developers who want to implement generative music composition software systems.

2.2 Data Retrieval

While performing data retrieval, a comprehensive search for research papers was performed. Boundaries of the systematic review were determined and keywords were fixed for the search. The query for the search is mentioned below:
(sound OR music) AND (biologically OR evolutionary OR evolution OR generative OR genetic OR creativity OR creative) AND (algorithms OR composition OR software OR system)

The query mentioned above provides the most relevant results considering the subject of this thesis. The query providing specific music systems using **biologically inspired algorithms** in the repositories is considered as providing relevant results. Nearly 40 variations of this query were tried out to find the relevant results for this research. The decisions for the search strategy are mentioned in the table below:

Table 2 - Search Strategy

Searched items	Books, conference papers, workshop papers, journal articles, thesis
Searched databases	ACM Digital Library, IEEE Xplore, ScienceDirect, Springer Link, Web of Knowledge, Wiley Interscience Journal Finder
Search applied on	Full text
Language	English

Every search result was documented in a careful way. The table which indicates total number of included studies is shown below.

Table 3 - Total Number of Included Studies

The Database	Total Found After Search Query	Exclusion Criterion 1	Exclusion Criterion 2	Number of Included Studies
ACM	652	254	69	4
IEEEXplore	633	228	45	4
ScienceDirect	73	25	7	5
Springer Link	589	217	46	4
Web of Knowledge	48	19	8	4
Wiley	5	3	2	2
Other	43	15	10	7
TOTAL	2043	761	187	30

Table 3 shows the names of the repositories, the number of the studies found in each repository after the search query, the sum of the studies found in all repositories

after the search query, the number of studies found in each repository after exclusion criterion 1, the sum of the studies found in all repositories after exclusion criterion 1, the number of the studies determined after exclusion criterion 2, the sum of the studies after exclusion criterion 2, the number of the included studies found in each repository, and the sum of the included studies. Under the database column, there is a box named other, which refers to the papers that could not be found through search engines. These papers mentioned in the other section were found in the 1st International Workshop On Musical Metacreation (MUME2012) [Pas13], EvoMUSART, which has become an EvoStar [evo13] conference with independent proceedings, and Generative Art International Conferences, Exhibitions and Live-Performances [art13]. There are varying levels of documentation about the works. There are many generative music systems out there, but they are hard to find and not well documented.

2.3 Study Selection

To be able to determine the relevant studies for this systematic review, study selection was performed. The search strings were wide. For this reason, all the results that were found were not relevant for the intended study. Some of the studies found had to be eliminated by determining exclusion criteria to be able to find relevant studies.

Table 4 - Exclusion Criteria

Exclusion Criterion No	**Details**
1	The studies which are not **Computer Science Related** were excluded.
2	The studies which do not **offer a method or methods** such as biologically inspired computational methods were excluded.

Table 4 presents the two exclusion criteria of this SLR. Exclusion criterion 1 requires that the studies which are not Computer Science Related are excluded. Exclusion criterion 2 requires that the studies which do not offer a method or methods such as biologically inspired computational methods are excluded.

In addition, the studies which are in English were included.

Thirty studies selected in accordance with the above mentioned criteria and providing information of reference number of the included study, software name, and reference are presented in Appendix D.

2.4 Data Extraction

To perform data extraction, Data Extraction Form, which is shown in Appendix C, was created and for each included study data extraction form was filled out. To answer two research questions, contents of papers were categorized into six; i.e. general information, motivation, problem statement, methods and approaches, features of the software system, and results and conclusion. This efficient activity helped the author to analyze, understand and review the papers once more.

2.5 Data Synthesis

When results were presented, papers were aligned to each other. The results of 30 included study is gathered from the included studies. However, reporting software engineering (SE) papers is rather different; thus data synthesis became harder. The papers were divided into groups and the results were presented accordingly. Thirty studies were considered as relevant for the goals of the systematic literature review; however, while presenting the results according to the research questions, some of these papers turned out to be inapplicable for the specific inquiry.

3 Results

Each included study describes one generative music composition software system. Consequently, 30 software systems were reviewed. The results of the systematic review of the 30 studies finally selected are presented below the respective research questions. The results are provided based on the research questions which are mentioned in Section 2.1.

It is important to mention that all of the information that is given in main functionalities, benefits and limitations and biologically inspired computational methods were extracted from the included studies by systematically reviewing them. In some of the included studies music types, programming languages, programming environments of the systems were not mentioned. The authors of the included studies were interviewed to gather this information. The authors were contacted via email. For example in Spieldose programming language and programming environment was not mentioned in the included study. An email was sent to the authors and they were asked which programming language and environment were used to implement Spieldose. The author replied back via email, and the information needed was provided in this research. Some of the authors of the included studies did not reply to the email, so the information needed could not be provided.

3.1 Methods and Approaches Proposed for GMCSS (RQ.1)

3.1.1 Biologically Inspired Computational Methods Used for Music Generation

Since every included study describes one generative music composition software system, biologically inspired computational methods related to names of **software systems** are presented in this subsection.

3.1.1.1 Definition of Terms

Neural Networks (NNs) and Genetic Algorithms (GAs) mimic biological processes. NNs substantiate the evolutionary device of learning from experience, as animals and humans do. GAs are based upon rules that simulate the laws of natural selection [BV99]. GAs are one specific type of Evolutionary Algorithms (EAs). Furthermore, in some studies the term GAs is used in its general meaning which creates confusion between GAs and EAs. In this study the term EAs is used. EAs form a subset of Evolutionary Computation (EC) in which they only include techniques implementing mechanisms such as reproduction, mutation, recombination, natural selection and survival of the fittest [Wik13d]. EAs are very effective techniques for searching very big, unstructured solution spaces. EAs start with randomly generated solutions to a problem. To be able to find better solutions, they use the equivalent of biological recombination. They find an optimal set of solutions which are represented by chromosomes. Strings of alleles are represented by strings of numbers and the recombination of chromosomes is a matter of creating new strings with alleles taken from the parent chromosomes. Because solutions are generated by experimenting answers and mixing the answers which work best, the method suits to solve fuzzy problems where the solution domain poorly behaves or where there is not a clear path to evaluate the solutions objectively [app95].

Reproduction which is the first operation of the EAs includes the selection of individuals from the mating pool, each with a probability in proportion to its fitness. **Crossover**, which is the next operation, is applied to these individuals in the mating pool. Pairs of individuals are randomly selected and a point along their length is also randomly selected. Two new individuals are produced by swapping the bits between individuals after the crossover site. This procedure is iterated for all the individuals in the mating pool. **Mutation**, which is the third phase of the algorithm, is a low chance event where a single bit is changed in a string in the population. The aim of mutation of the algorithm is to interrupt important genetic material from being lost irrecoverably. Moreover, mutation helps preventing premature convergence of the algorithm on a sub optimal solution.

Interactive Evolutionary Computation (IEC) is a term for methods of evolutionary computation which use human evaluation. Generally human evaluation is necessary when the form of fitness function is not known or the result of optimization must meet the requirement of a user [Wik12]. Interactive evolutionary algorithms are a form of evolutionary algorithms which gives the opportunity to perform the selection of the fitness function to the users [Tak01], [Hor94].

IEAs and Fitness Functions.
The goal of IEAs is to learn criteria of the user for generating music. IEAs allow the user to execute fitness functions. In fitness functions, the user chooses the rhythms or features of rhythms or music pieces that he likes. The user does not need to understand the details or parameters of these functions. As the system learns what users like, the quality of the music and rhythms improves according to taste of the user [Hor94].

EA and Fitness Functions.
It is important to mention that Interactive evolutionary algorithms are a sub type of evolutionary algorithms. If the software is not interactive, then *fitness function* decides which individuals to be breed and killed. If the software is interactive then the user selects which individuals to breed for the next generation. To analyze fitness functions in terms of EAs and IEAs, the following information is needed: in IEAs the user is the only fitness function, so in every generation the user has to judge the music subjectively. However, because of the fitness bottleneck problem, it takes a lot of time for the user to judge the music subjectively [Hor94]; thus there is no real evaluation and it takes too much time to carry out some real evaluation. Because of that reason, a large battery of research has been performed on how to optimize automate fitness functions. One solution is to do a few generations automatically and then evolve the individual to do a few generations and back and forth. Another solution is to create a neural network in parallel which learns from the users' choices and can take over during several extra generations. A third option is that the system can learn over longer time and it actually becomes autonomous.

Cellular Automata are computer modeling techniques which are used to model systems in which time and space is represented discretely and quantities take on a finite set of discrete values [BE05].

Ant colony optimization algorithm is an algorithm to find optimal paths which is based on the behavior of ants seeking for food. At the beginning, the ants wander randomly. When an ant finds a source of food, it walks back to the colony leaving markers which show the path has food. When other ants discover the markers, they are likely to follow the path. If they follow the path, then they populate the path with their own markers as they bring the food back. As more ants discover the path, it gets stronger until there are a couple of streams of ants traveling to various food sources near the colony [Mac13]. For example, ANTRACKS uses a very simple ant colony optimization algorithm (no pheromone, only one ant per hive); however, the behavior could be characterized by exploration and seeking food.

The combination of neural networks and evolutionary algorithms provides a different method for solving complex problems where neither the size of the solution nor the detailed structure is known in advance [SG99]. Evolutionary algorithms and neural networks are used together in two major ways. The first major type of collaboration is to use EAs to help the training of neural networks. In particular, EAs are used to search for the weights of the network, to search for proper learning parameters, or to reduce the size of the training set by selecting the most relevant features. The second major type of collaboration is to use EAs to design the structure of the network. The

structure mostly decides the efficiency of the network and the problems that it can solve. To solve non-linearly separable problems, the network must have at least one layer between the inputs and outputs; however determining the number and the size of the hidden layers is mostly a matter of error and trial [CPKK93].

N/A represents the information which can not be reached via reviewing the included study and interviewing with the author or authors of the included study. N/A is used in the Table 5, 6, 7, and 8.

Table 5, which is shown below, represents the distribution of software systems according to biologically inspired computational methods used with reference numbers of the included studies.

Table 5 - Distribution of Software Systems According to Biologically Inspired Computational Methods with Reference Numbers

Biologically Inspired Computational Methods Used	**Ref**	**Total**
Evolutionary Algorithms	[3],[5],[6],[8],[9],[11],[12],[17],[18],[24],[25],[27],[29]	13
Interactive Evolutionary Algorithms	[10],[13],[16],[19],[20],[21],[26],[28],[30]	9
Cellular Automata	[23]	1
Neural Networks	[1],[22]	2
Ant Colony Optimization Algorithm	[4]	1
Combination of NNs and EAs	[14],[15]	2
Combination of IEAs and NNs	[2],[7]	2
Total Number of Systems		30
N/A		0

In the included study named Experiments in Modular Design for the Creative Composition of Live Algorithms whose reference number is [22] there are two systems. One uses Continuous-Time Recurrent Neural Networks. The other uses an idiosyncratic type of Decision Tree with feedback. Both are complex systems with lots of recurrences and both have their structure modified by a NON-interactive fitness-function-based evolutionary process.

When the included studies were reviewed for biologically inspired computational methods used for music generation, 30 out of 30 software systems were extracted for this specific information. Thirteen systems use evolutionary algorithms, 9 systems use interactive evolutionary algorithms, 1 system uses cellular automata, 2 systems use neural networks, 1 system uses ant colony optimization algorithm, 2 systems use a combination of neural networks and evolutionary algorithms, and 2 systems use combination of interactive evolutionary algorithms and neural networks.

The statistical information about biologically inspired algorithms used is provided in Figure 2.

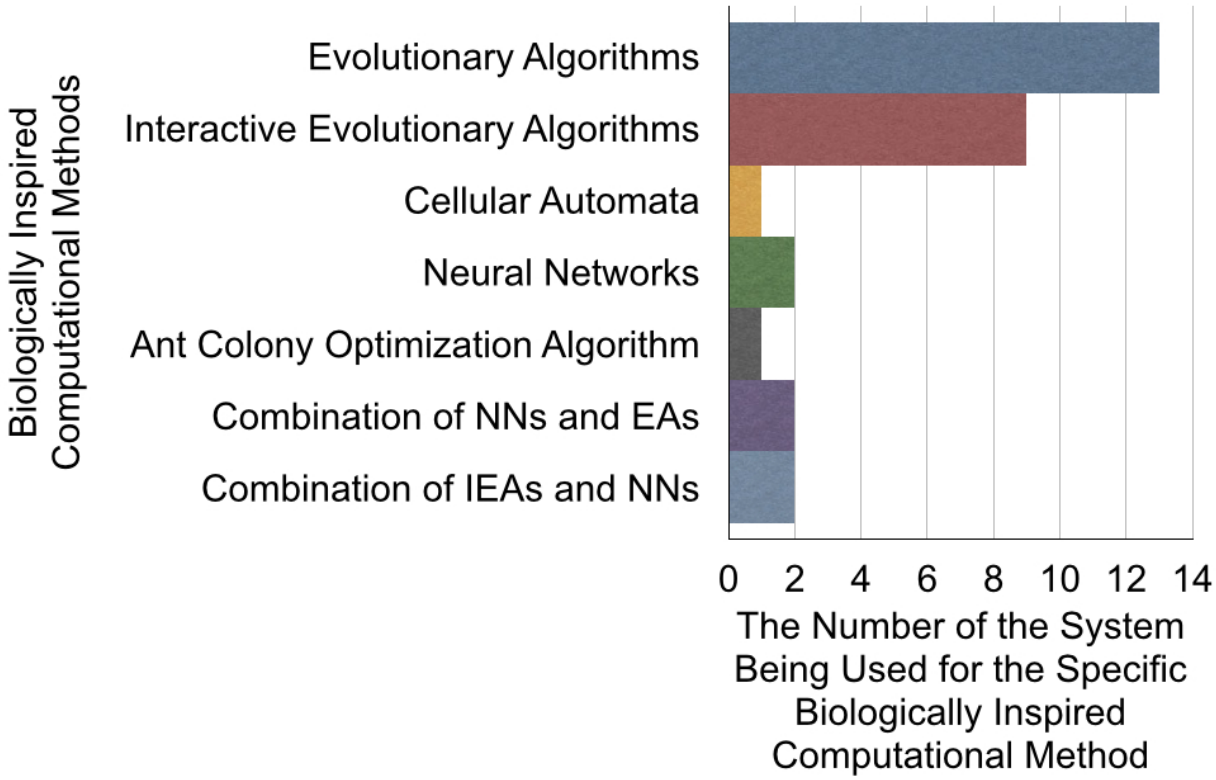


Figure 2: The Statistical Information About Biologically Inspired Computational Methods Used.

3.1.2 Music Types of GMCSS

3.1.2.1 Definition of Terms

One music type of GMCSS is **harmony generation**. Harmony is the use of simultaneous pitches (notes, tones), or chords [Mal77]. The harmony includes chords, their arrangement, chord progressions and the theory of connection that carry them out [GJ79]. Harmony is the vertical aspect of music while melodic line is known as a horizontal aspect [Bur67]. Harmony generation is the theory behind the progression of chords; i.e. harmonies. The order of harmonies is suitable to order different chords and these chords can follow upon others. Harmony is the skeleton of a tonal piece of music. In that case, **voice leading**, which means how to arrange the notes of the chords, can be thought to be a part of the theory of harmony. Types of GMCSS are divided into subtypes called **electronic music generation** and **complete composi-**

tion. Electronic music generation can easily be confused with complete composition. Nevertheless, the systems which belong to electronic music generation deal primarily with sounds not notes while in complete composition the focus is notes.

In the scope of this research complete composition refers to musical melody that has more than one musical instrument playing at the same time. When the user listens to the generated music (which is a complete composition), he hears the music of an orchestra playing.

Soundscape generation including the sound of birdsongs, waterfalls, rain and wind is not exactly music. They are soundscapes.

Table 6, which is shown below, represents the distribution of software systems according to music types with reference numbers of the included studies.

Table 6- Distribution of Software Systems According to Music Types with Reference Numbers

Music Types	**Ref**	**Total**
Rhythm Generation	[7],[20]	2
Melody Generation	[2],[4],[8],[9],[11],[25],[27],[28]	8
Harmony Generation	[12],[18]	2
Electronic Music Generation	[6]	1
Soundscape Generation	[14]	1
Sound Synthesizer	[23],[24]	2
Complete Composition	[1],[3],[5],[10],[13],[15],[16],[17],[19],[21],[22],[26],[29],[30]	14
Total Number of Systems		30
N/A		0

In the included study named Experiments in Modular Design for the Creative Composition of Live Algorithms whose reference number is [22] there is an interactive behavior. A performer plays into the system and it outputs music. In addition to the decision layer described above there is a hand-coded generative music layer.

When the included studies were reviewed for the types of generative music composition software systems 30 out of 30 software systems were extracted for this specific information. Fourteen systems generate complete composition, 8 systems generate music melody, 2 systems generate rhythm, 2 systems generate music harmony, 1 system generates electronic music, 1 system generates soundscape, and 2 systems are sound synthesizers.

Statistical information about types of GMCSS is provided in Figure 3.

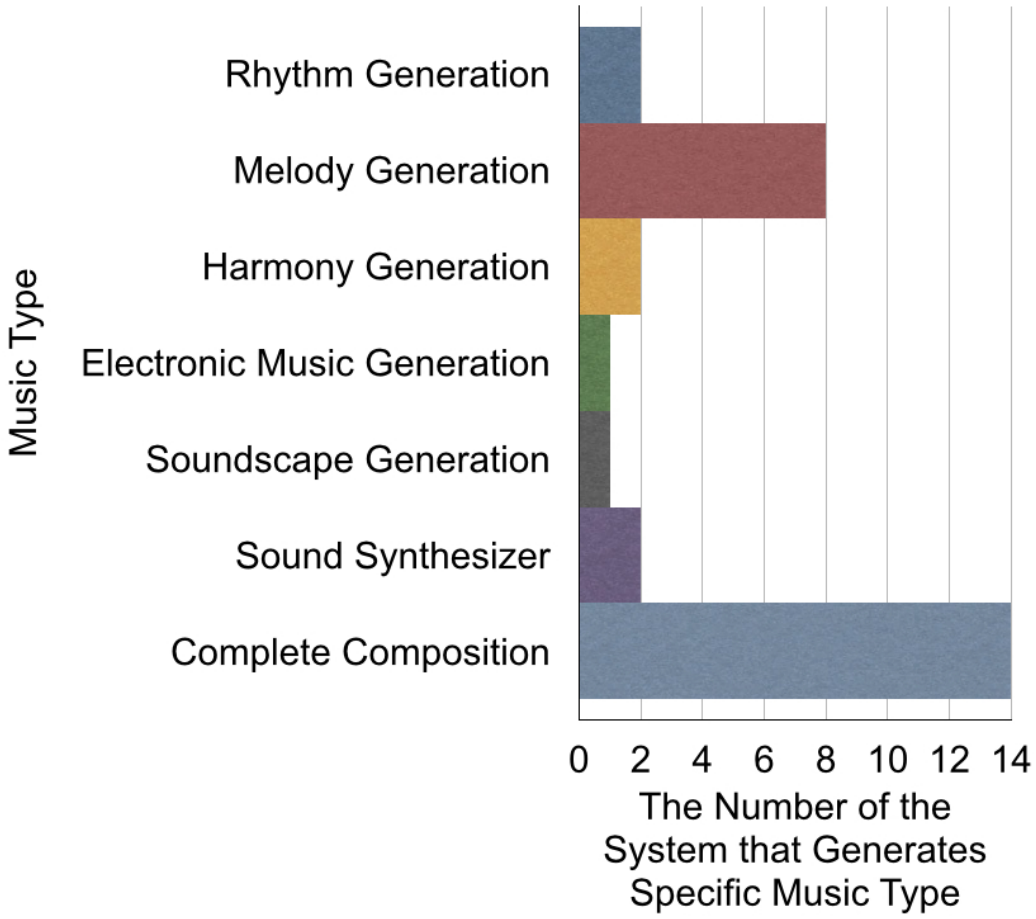


Figure 3: The Statistical Information About Music System Types.

3.1.3 Programming Languages and Programming Environment

This subsection is directed towards explaining with which programming languages and programming environment 30 generative music composition software systems were created.

Table 7, which is shown on the next page, represents the distribution of software systems according to programming languages with reference numbers of the included studies.

Table 7- Distribution of Software Systems According to Programming Languages with Reference Numbers

Programming Languages	**Ref**	**Total**
C	[18],[23]	2
C++	[15],[16],[20],[23],[24],[28]	6
C#	[7]	1
Objective C	[4],[29]	2
Python	[17]	1
Java	[11],[13],[22],[19],[25]	5
THINK C version 5	[5]	1
Common Lisp	[27]	1
Total Number of Systems		19
N/A		11

Vox Populi [19] was developed in 99, in Visual Basic. It was further translated to Java in Eclipse environment, and became the kernel of AURAL.

GenJam [5] is implemented in Think C, version 5, on top of the Carnegie Mellon MIDI Toolkit by Roger Dannenberg, running under Mac OS 7.1. THINK C was an extension of ANSI C for Mac OS developed by THINK Technologies. The following versions of THINK C were mainly a subset of C++ and supported basic object oriented programming concepts such as single inheritance as well as extensions to the C standard which was adapted more closely to the requirements of Mac OS programming [Wik13i].

Moreover, 16 out of 30 of software systems are implemented in **C based languages** which are C, C++, Objective C, Java, C# and 16 out of 30 of the systems are implemented with **object oriented programming languages** which are C++, C# , Objective C, Python, Java and THINK C.

When the included studies were reviewed for programming languages of generative music composition software systems, 19 out of 30 software systems were extracted for this specific information. Six systems were implemented by using C++, 5 systems were implemented by using Java, 2 systems were implemented by using C, 1 system was implemented by using C#, 2 systems were implemented by using Objective C, 1 system was implemented by using Python, 1 system was implemented by using THINK C version 5, and 1 system was implemented by using Common Lisp.

Statistical information about programming languages is provided in Figure 4.

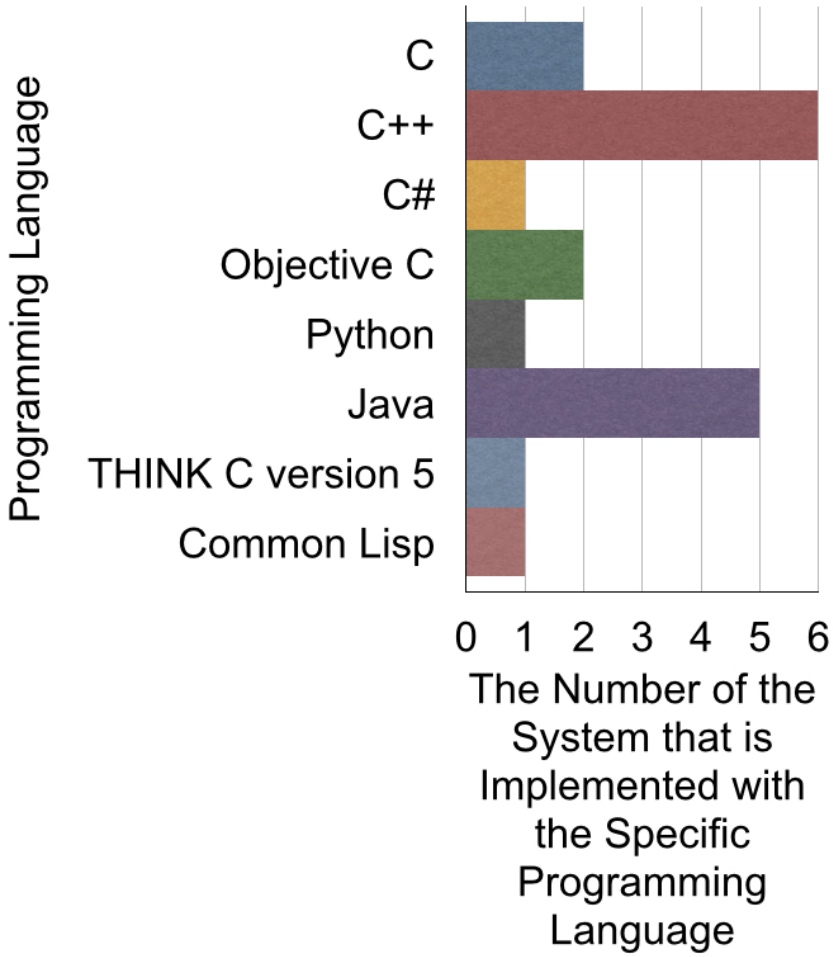


Figure 4: The Statistical Information About Programming Languages.

3.1.3.1 Definition of Terms

Pure Data (Pd) is a real-time graphical programming environment for audio, video, and graphical processing [fPD13]. It is an open source of visual programming language and helps musicians, visual artists, performers, researchers, and developers to create software graphically, without using lines of code. Max is an object-oriented language to be able to program interactive musical processes. It has a graphic user interface in which functional objects can be connected in larger systems. It receives input from the keyboard, the mouse and instruments with MIDI capability. Besides MIDI output, Max can control compact and video disk players, show still and animated graphics, and communicate with digital signal processing chips. It also allows the installation of user written objects in C to be able to extend functionality beyond the already large set of standard features. The reason why Pd and Max/MSP are in the same category is that Pd is a major branch of the family of patcher programming languages such as Max/MSP. Pd was implemented to further the Max paradigm by extending data processing to applications other than audio and MIDI, such as real time video and

web interaction [fPD13]. Pd is written in plain C. When it parses a patch, it does not actually generate code; it simply builds up internal data structures that represent the signal processing graph.

Moxc is a real-time programming environment that is very ideal for writing interactive music programs. GenJam [5] is written in the programming environment of Moxc. It is based on Douglas Collinges Moxie language and is an extension of the C programming language [Dan96].

MATLAB is an interactive environment and a high-level language for numerical visualization, computation, and programming. [PPPPTPSHTM13].

Table 8, which is shown below, represents the distribution of software systems according to programming environments with reference numbers of the included studies.

Table 8- Distribution of Software Systems According to Programming Environments with Reference Numbers

Programming Environment	**Ref**	**Total**
Eclipse	[22],[11],[13],[19]	4
NetBeans	[23],[24]	2
CodeWarrior IDE	[20]	1
Microsoft Visual Studio 6.0 on Windows	[19]	1
Moxc	[5]	1
Matlab	[10]	1
Pure Data and Max/MSP	[14],[17],[18],[29]	4
Arduino Environment	[25]	1
.NET with DirectX	[28]	1
Macintosh Common Lisp	[27]	1
Total Number of Systems		17
N/A		13

When the included studies were reviewed for programming environment of generative music composition software systems, 17 out of 30 software systems were extracted for this specific information. Four systems are implemented by using Pure Data, which is a user friendly IDE, and Max/MSP, 4 systems are implemented by using Eclipse, 2 systems are implemented by using NetBeans, 1 system is implemented by using CodeWarrior IDE, 1 system is implemented by using Microsoft Visual Studio 6.0 on Windows, 1 system is implemented by using Moxc, 1 system is implemented by using MATLAB, 1 system is implemented by using Arduino Environment, 1 system is implemented by using .NET with DirectX, and 1 system is implemented by using Macintosh Common Lisp.

DOT which has included study number [25] the author used the Arduino Enviroment which is based on Java programming language.

Statistical information about programming environments is provided in Figure 5.

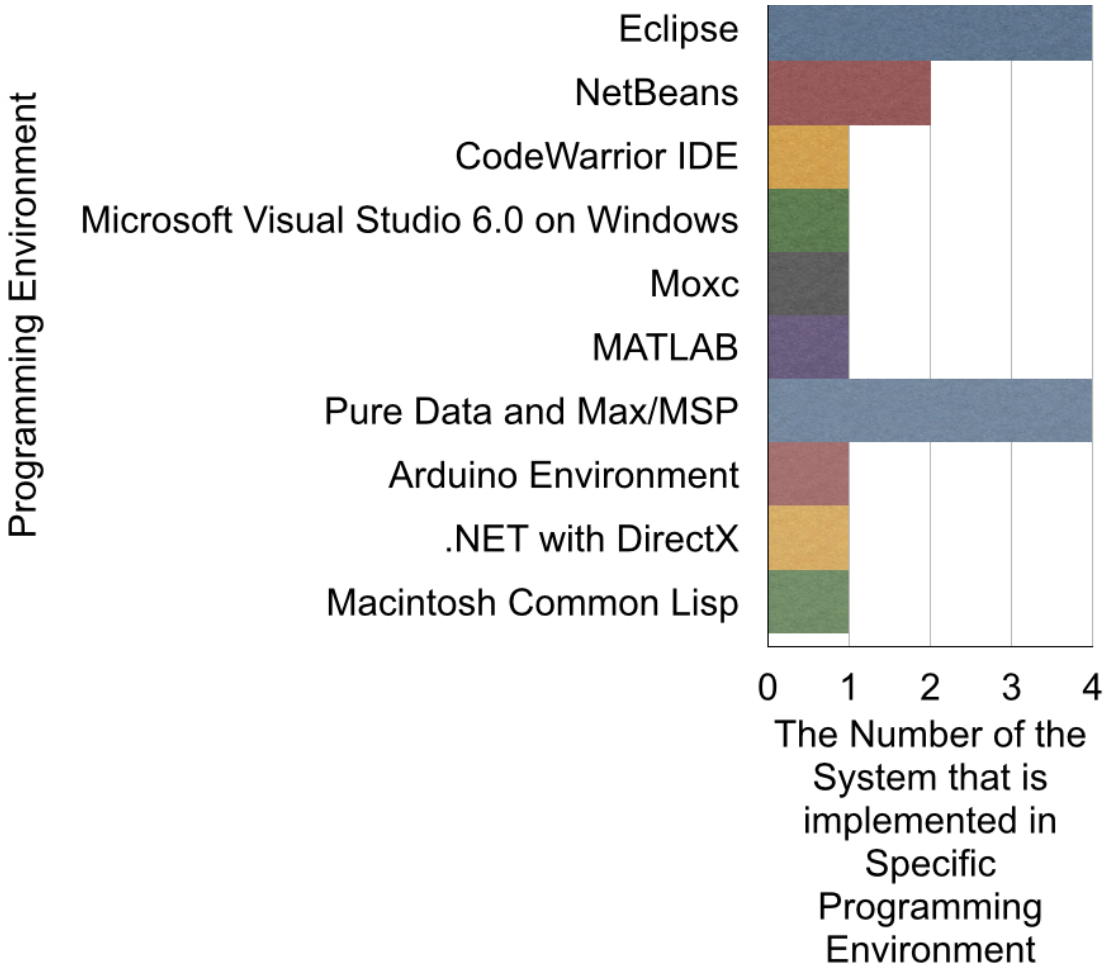


Figure 5: The Statistical Information About Programming Environments.

A reader might be confused when he observes the software named Mezzo whose included study reference number is [17] because programming environment is MAX/MSP and the programming language is Python. While implementing a software, more than one programming languages and programming environments can be used. For instance, if the software developer wants to implement a generative software system for mobile devices, he can use Pd to implement the sound engine of the system, OpenFrameworks for handling the graphs and the multitouch gestures, and XCode (which implementer writes his code in Objective C). Mezzo is written in Python, and uses Max as an interface for sending game signals to the Python code and handling playback. Communication between Python and Max is implemented by using Open Sound Control.

3.2 Main Functionalities, Benefits and Limitations of GMCSS (RQ.2)

3.2.1 Nodal

Nodal [MM11] was created by McCormack, McIlwain, Lane, and Dorin. It is important to note that Nodal is different from all the remaining systems that were included in the study. Philosophically it is related with NNs.

Main Functionalities.

- Users create a graph by using nodes and edges. Nodes are connected via edges. A player which is a musical agent traverses the graph in real-time. While the player traverses the nodes, music is created.
- The system allows the composer to control and structure processes in a compositional sense.
- The system allows the users to design dynamic graphic notation systems.
- The users can interpolate controller information as they travel along edges.
- The users can change the musical composition while the program is running.

Benefits.

- The composer can achieve difficult tasks by using conventional notation software.
- The system provides many possibilities for generating complex, emergent structures
- Nodal has an extremely simple interface.
- There are live performance tools in the system.

Limitations.

- The system specifies the graphs in two dimensions. This becomes a limitation when trying to design complex topologies. To eliminate this limitation, software developers created wormhole edge. It transfers any agents traveling on it between the nodes to which the edge is connected.

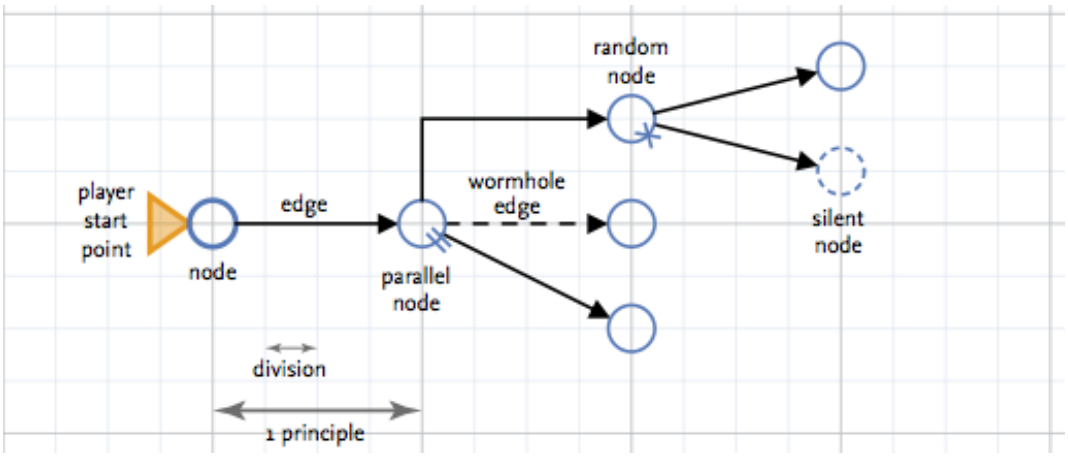


Figure 6: Elements of a network: the player start point, nodes, edges and the timing grid. The darker grid lines represent principles and lighter grid lines are divisions. In this example there are four divisions per principle.

[***MM11***]

- In the system there is a rigid structure of the metrical grid. It can make sequenced musical notes. Because of the perfect timing, sequenced musical notes sound mechanical. Software developers are working on musically useful ways to circumvent this problem, without losing the benefits that the grid system provides.

3.2.2 GP-Music

GP-Music [Bra98] was created by Johanson and Poli.

Main Functionalities.

- The system allows users to derive short musical sequences.
- Users can interactively evolve music.
- GP-Music system uses IGP. The main feature of IGP is that it achieves good results without the need of explicitly clarifying a lot of domain knowledge for a problem. As a result, this important feature belongs to GP-Music system.
- GP-Music User Interface, which is shown in Figure 7 below, gives an opportunity to users to rate individual sequences using a list. The users rate each musical sequence by giving numbers between 1 and 100. Moreover, they can change their mind about rating of sequences after they heard what the competing sequences sound like.

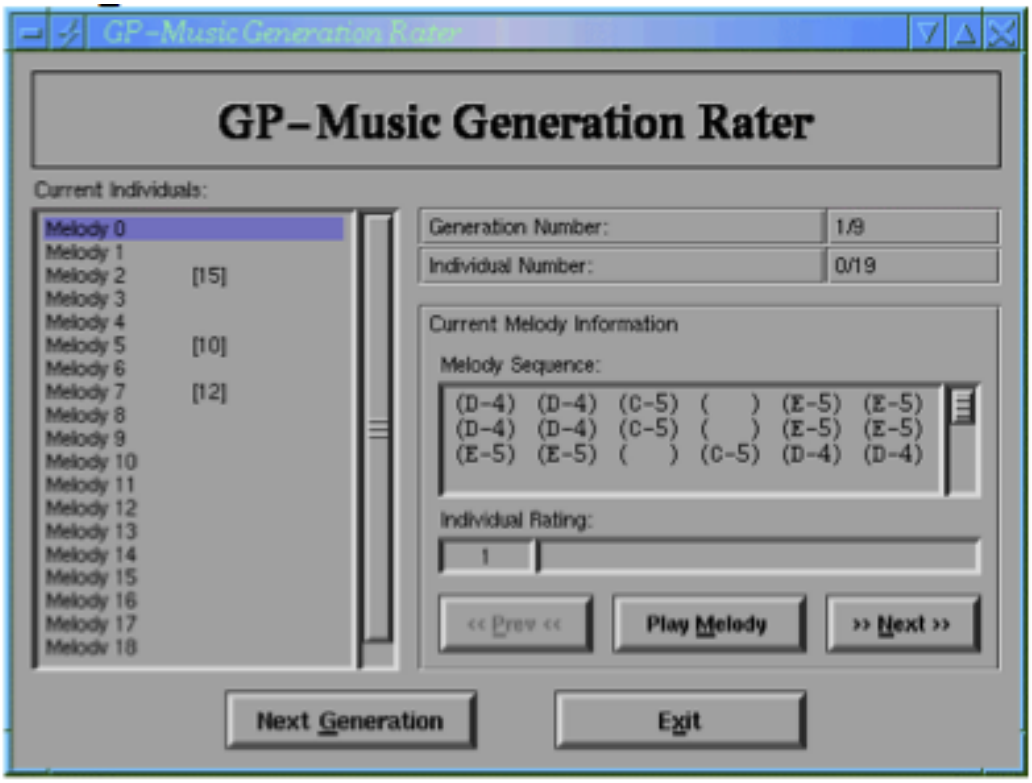


Figure 7: The GP-Music User Interface.

[***Bra98***]

Benefits.

- There is an extension to GP-Music system called automated fitness raters. During the evolutionary process, automated fitness raters learn to rate in a similar way to the user. This functionality allows users to have longer runs to make. Moreover, it allows the system to operate both with and without user interaction. Lastly, it allows the system to work in a fully automated mode.
- IGP is better than GA for the tasks such as the runs using only simple concatenation and using complex structuring functions. Even though either GA or IGP is used, there is still the user bottleneck problem. The user is able to rate a small number of sequences. This problem is alleviated with auto raters. They learn to rate sequences in a similar fashion to the user, allowing longer runs to be made.

Limitations.

- Before automated fitness raters were implemented and integrated into GP-Music system, one of the main problems with the system was that the user had to listen to and rate each musical sequence in every generation during a run.
- Ratings being subjective are a difficulty with IGP applied to music. In the reproduction an individual is copied from the previous generation into the new generation. If the copied sequence is presented to the user beforehand, possibly the user could rate

the copied sequence differently from the previous generation. To manage this problem, ratings of the individual are locked from generation to generation. The goal is to lead the user to rate consistently.
- The problem which occurred in the previous versions of the system was that genesis of melodies was either too long or too short. These melodies always receive low ratings from the users. Having these melodies in the population decreased the efficiency and diversity of the evolution process. The problem is solved in the next version of the system by making the users choose a certain note sequence on minimum and on maximum for a run. Individuals which do not meet the criteria are killed by the system. A new individual is created when a satisfactory individual is found.

3.2.3 SBEAT

SBEAT [US01] was implemented by Unemi and Senda.

Main Functionalities.

- The system uses simulated breeding which is a method for finding solutions in some domains and for optimization based on subjective criteria of users. The users select favorite individuals from a population as parents for the next generation.
- GUI (Graphical User Interface) of SBEAT is shown in Figure 8. The window includes nine sub-windows divided into three by three grids. The users evaluate each bar by hearing the music and viewing the scores on the screen. Each sub-window shows the score of 4 parts chosen from: percussion, drums, piano, bass, baritone, tenor, alto, and soprano.
- As it is shown in Figure 8, there is a button labelled **play all individuals** at the top left corner which plays all the individuals in the population sequentially. This feature is useful for finding the best candidate in the population.
- *See, play and listen.* As it is shown in Figure 8, for each individual the scores are shown so that the user can imagine what the sound will be like without listening to it.

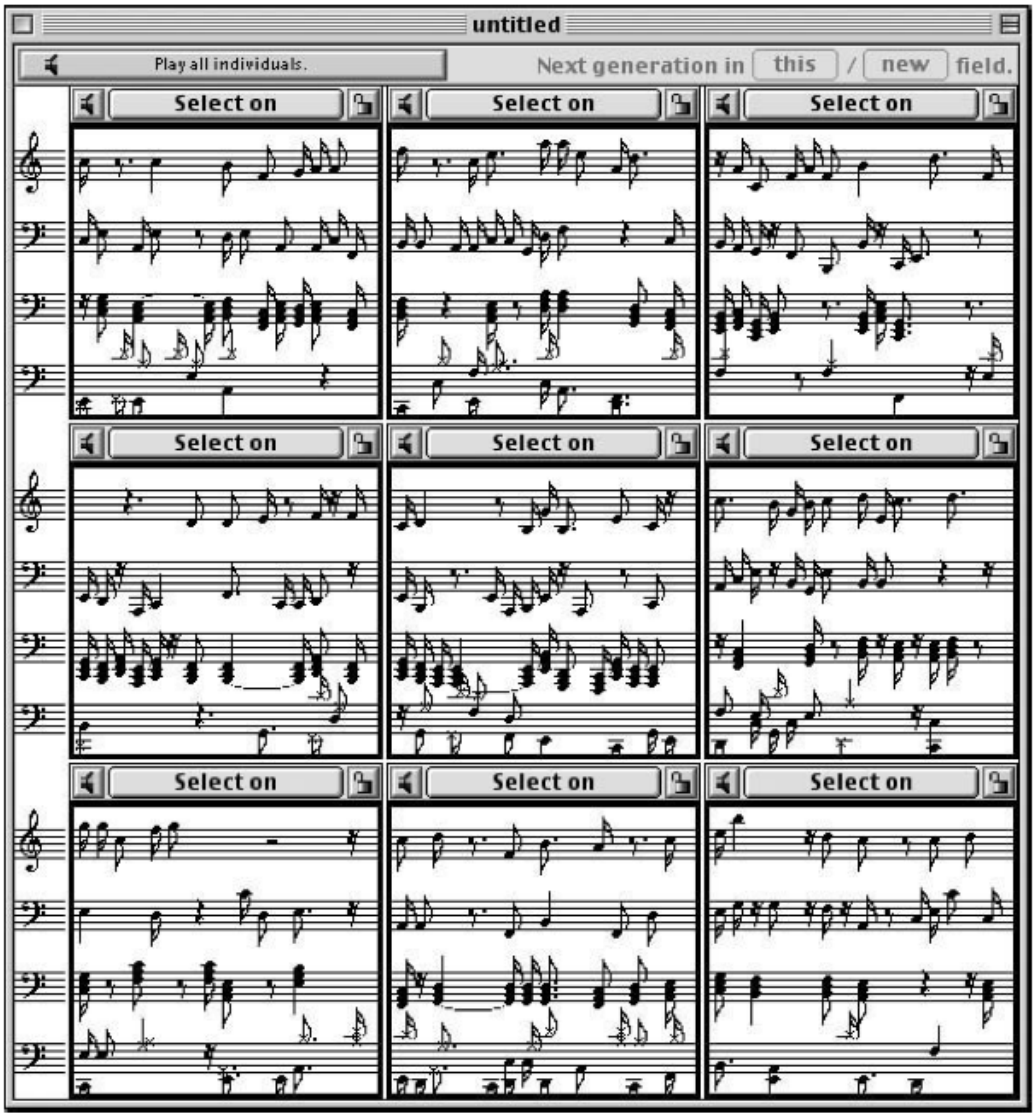


Figure 8: A typical field window of SBEAT containing nine initial individuals.

[***US01***]

GUI of part options is shown in Figure 9. SBEAT can play eight instruments; however, it can only show a maximum of the notes of four instruments, which are selected by the user, because of screen space restrictions.

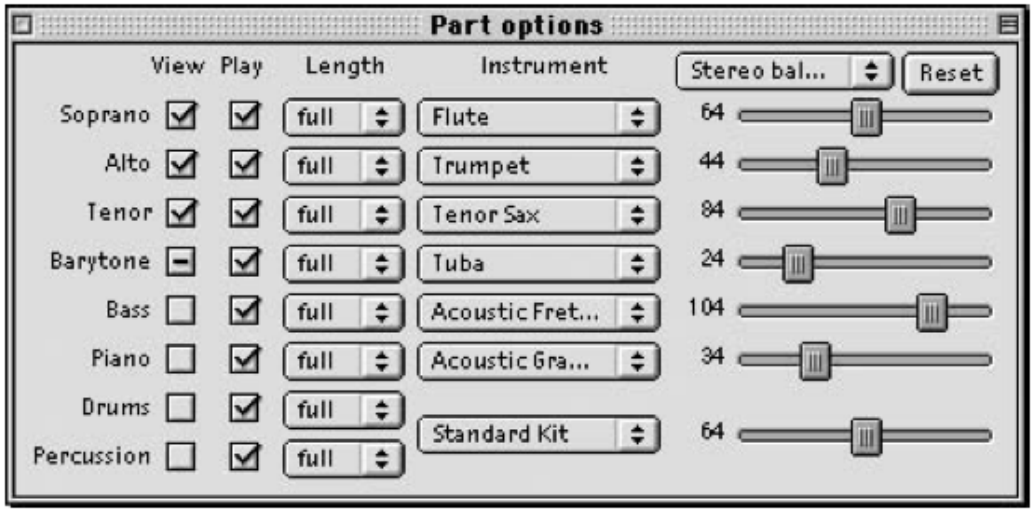


Figure 9: Part option dialog.

[***US01***]

In Figure 10 below, an example of a mutation produced from a single parent and the crossover from two parents are presented.

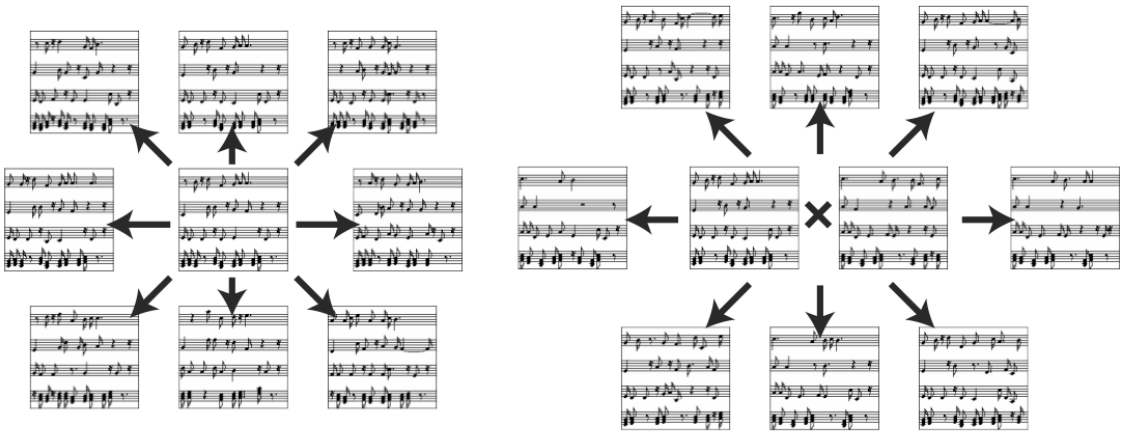


Figure 10: Typical examples of a mutation produced from a single parent are on the left, and the crossover from two parents is on the right.

[***US01***]

- *Migration and integration.* The user can create different populations using several fields independently and make some individuals migrate to another field by the drag and drop operation.

- *Genome editor*. The user can edit chromosomes directly with the help of the genome editor to achieve better results. Figure 11 shows two windows, a score of an individual on the left and an editing panel on the right. The user selects a part to be edited by operating buttons allocated to chromosomes and beats.

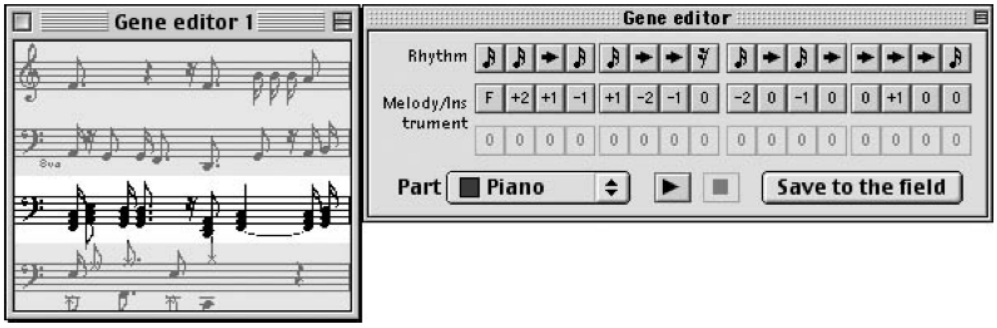


Figure 11: Genome editor.

[***US01***]

Benefits.

- See, play and listen to the feature explained above; this feature helps beginners to learn to read scores. Moreover, this feature helps the musicians who can read notes to gain time because they can select the individuals that they want without listening to them.

Limitations.

The included study provides no explicit limitations.

3.2.4 ANTracks

ANTracks [SGR09] was implemented by Schulz, Geiger, and Reckter. In the included study ANTracks IOS application is introduced. The software developers of ANTracks were inspired by colonies of ants which seek food and bring it back home.

Main Functionalities.

- The user configures a system by controlling the number of virtual ants to generate harmonic musical expression.
- The user creates ants. They start moving on the grid. The user defines directions of ants by moving his finger with the wipe gesture in the way that he wants.
- GUI of ANTracks (b) and the corresponding musical notes for each grid (a) are shown in the figure below:

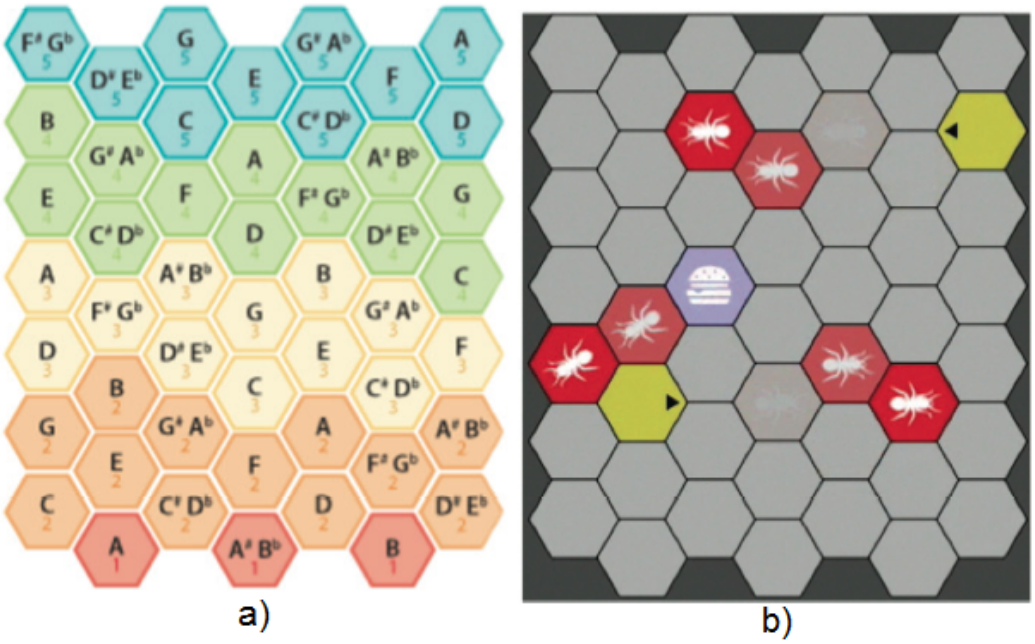


Figure 12: Harmonic Scale and ANTracks Grid with Three Ants. a) Assigned musical notes for each grid b) A harmonic table map populated by three ants.

[***SGR09***]

- In the system, the notes are sequenced with harmonic restrictions. The team, who implemented ANTracks, designed an interactive grid which has 7x7 hexagonal elements. If a grid element is filled with an ant, the corresponding musical note which is shown in a) is played. For example, if we look at b), there is an ant in the leftmost and bottom third. When we look at the same place in a), we realize that D musical note is specified. When the ant is in the specified place in b), D note is played. In b) the previous position of an ant is specified by a grayed image.
- The user is able to start/stop the music generation, place and change the objects on the grid and modify musical parameters such as pattern length and speed.
- As it is seen in the figure, the user can change parameters of tempo (BPM) and pattern length, send notes on, note scaling and more by using preference pages.

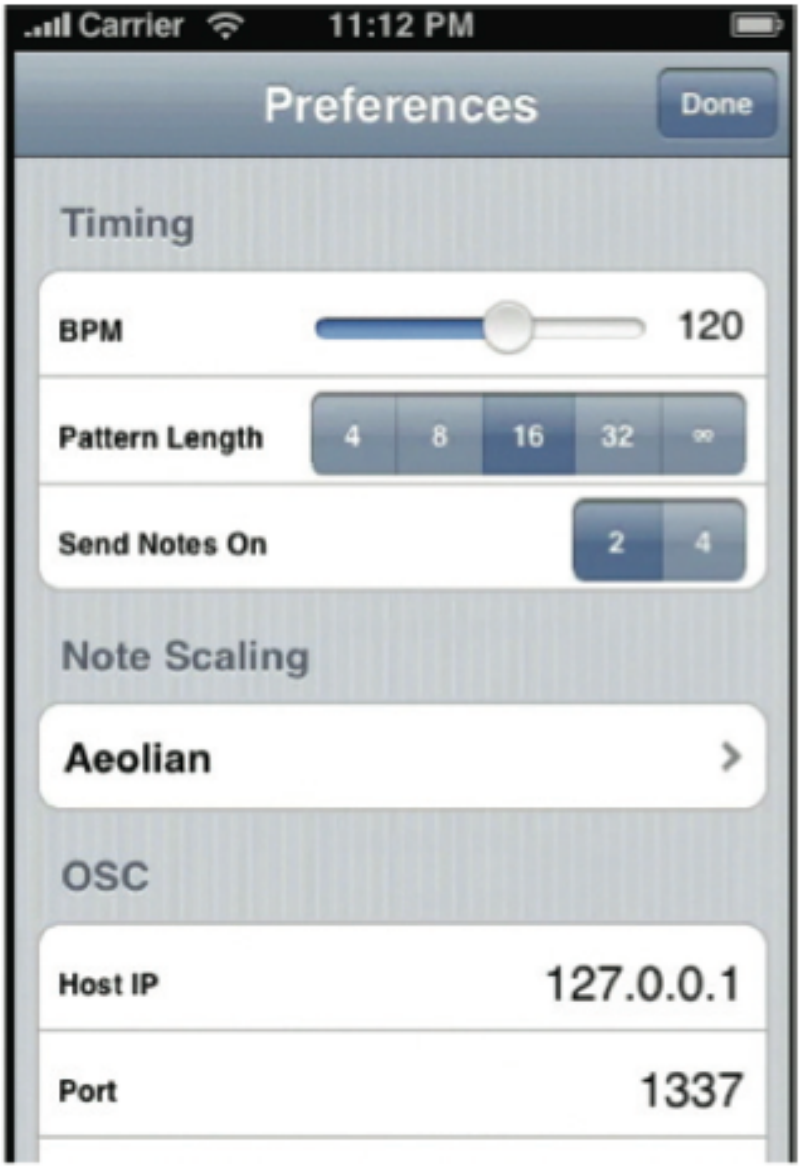


Figure 13: ANTracks preferences.

[***SGR09***]

Benefits.

- The users who evaluated the system mention that the product is innovative and attractive and stimulates them in a positive way.

Limitations.

- To generate music with computer based systems is a limitation in general because users are restricted with WIMP (windows, icons, menus, pointer) based human computer interaction. Since mobile devices provide smaller interaction spaces, the limitation about WIMP is even more pronounced.

- There are hardware restrictions about iPhone and iPod touch. For this reason, the system cannot be extended.
- Because usability of ANTracks has not improved enough, the users who evaluate the system cannot easily learn how to accomplish a specific task.

3.2.5 GenJam

GenJam, which represents Genetic Jammer, is a real-time jazz model system. This system is an interactive performance system which uses evolutionary computation to generate a jazz improviser. GenJam [Bil07] was implemented by Biles. The abbreviation of GenJam comes from GEN for Genetic Algorithms and JAM for jazz jam session. GenJam uses GA to improvise jazz music. Figure 14 below shows how GenJam works. GenJam focuses on applying EC in improvisation.

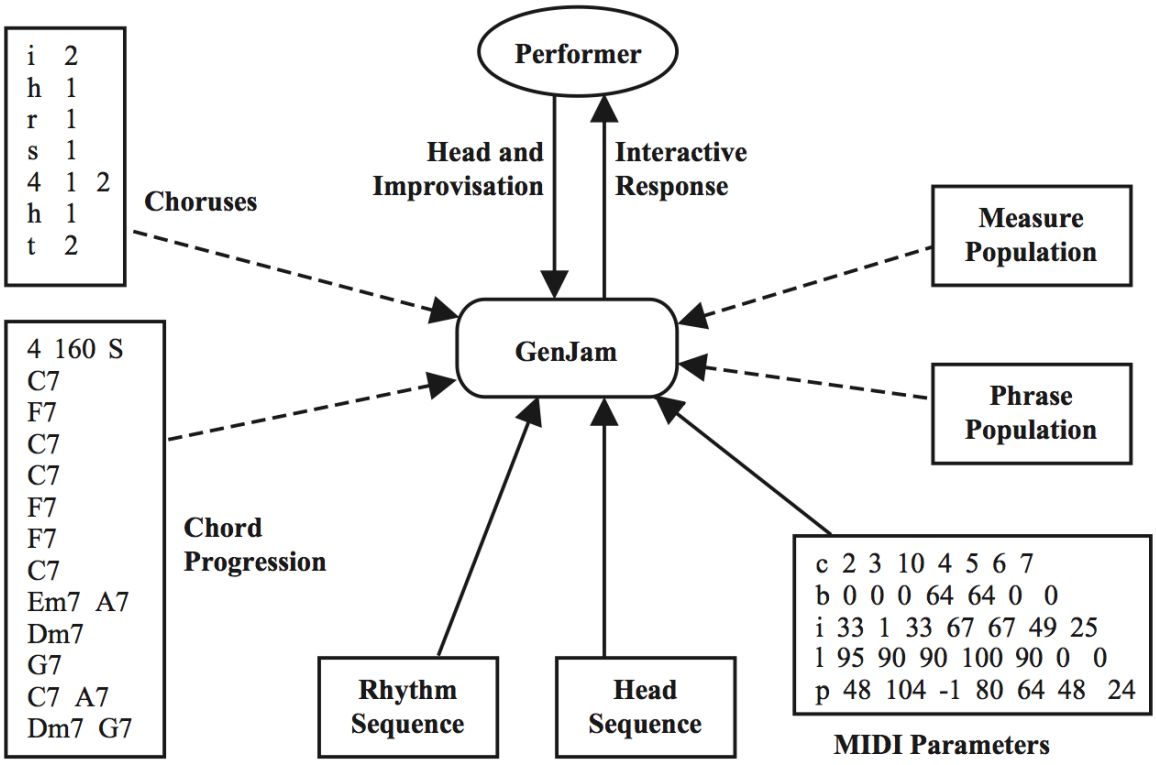


Figure 14: GenJam architecture in performance.

[*Bil07*]

There is a musical conversation between the user and GenJam. The user improvises and GenJam listens to the user, responds to and produces an interactive response. GenJam needs to know some important information about the tune to be able to play on the music piece of the user. The user should know the following lines to use GenJam properly:

1- Chord progression
2- Playing tempo
3- GenJam knows what to do for each of the choruses
4- The user needs to setup the tone generator so that GenJam plays the tenor sax such as the piano, drums, and the bass.
5- There is a type of file, called MIDI, and generated by GenJam. MIDI files have head sequence harmony. Band in a Box produces the rhythm sequence, piano, bass and drums by the help of these head sequence harmony [Inc13]. By means of these MIDI files, background music like piano, bass, drums etc. is generated by GenJam.

GenJam uses EAs for generating music. An interesting part of evolutionary computation is two populations of melodic ideas; i.e. a measure population and phrase population. The chromosomes live in these populations. A simple figuration of musical chromosomes is given below:

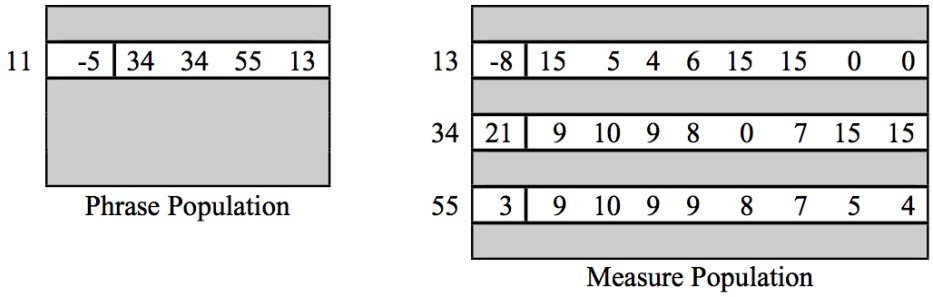


Figure 15: An example of phrase individual and its constituent measure individuals.

[*Bil07*]

The genotype measures into measure population.

Main Functionalities.

- The user improvises while the background music is playing. When the user plays, GenJam listens and maps the notes it hears to chromosomes. GenJam uses those chromosomes to generate its reply. The mutations are clever and they develop an idea for the user.
- There are five channels that have been evaluated. These are bass, piano, drums, strings, and guitar.
- After GenJam produces a piece of music, the user evaluates the music by entering G for good or B for bad so that the system produces a good piece of music (according to input from the subjective user) for the next generation.

Benefits.

- GenJam interacts effectively in real time with a human performer. In live performance this functionality is a very crucial benefit.
- There is a loudness threshold. It filters out sound from the speakers playing the rest of the band and ambient noise in the room, so that GenJam pays attention to the close-miked trumpet.

- If the user plays a good musical phrase, the mutations are guaranteed to be a good phrase in response.
- Theoretically GenJam can not play a wrong musical note unlike the human beings.

Limitations.

- As in all generative music software, fitness bottleneck is the biggest limitation for GenJam. The fitness bottleneck problem is solved by eliminating fitness itself.
- GenJam does not fulfill all the criteria required by the annual human competitive awards in genetic and evolutionary computation [BGGP04], directed towards science and engineering applications; however, the author thinks that GenJam at least holds its own with competent amateur improvisers.

3.2.6 Patch Mutator

Patch Mutator [Dah07] was created by Dahlstedt. It is a tool inside the NMG2 synthesis environment. It makes interactive evolution available in a sound processing and professional synthesis environment. The Patch Mutator is not really about evolution of sounds, but rather acts as an explorative tool based on evolutionary algorithms.

3.2.6.1 Definition of Terms

A sound synthesizer is an electronic instrument which is able to create a wide range of sounds. A synthesizer patch is setting of a sound. Modular synthesizers used patch cords for connecting the variant sound modules together. Because these machines had no memory to save settings, musicians wrote down the locations of knob positions and the patch cables on a patch sheet. Since that time, an overall sound setting for any type of synthesizer has been known as a patch [Wik13g].

Main Functionalities.

- The Patch Mutator has 5 different sections. These 5 different sections and their functionalities are shown in Figure 16.

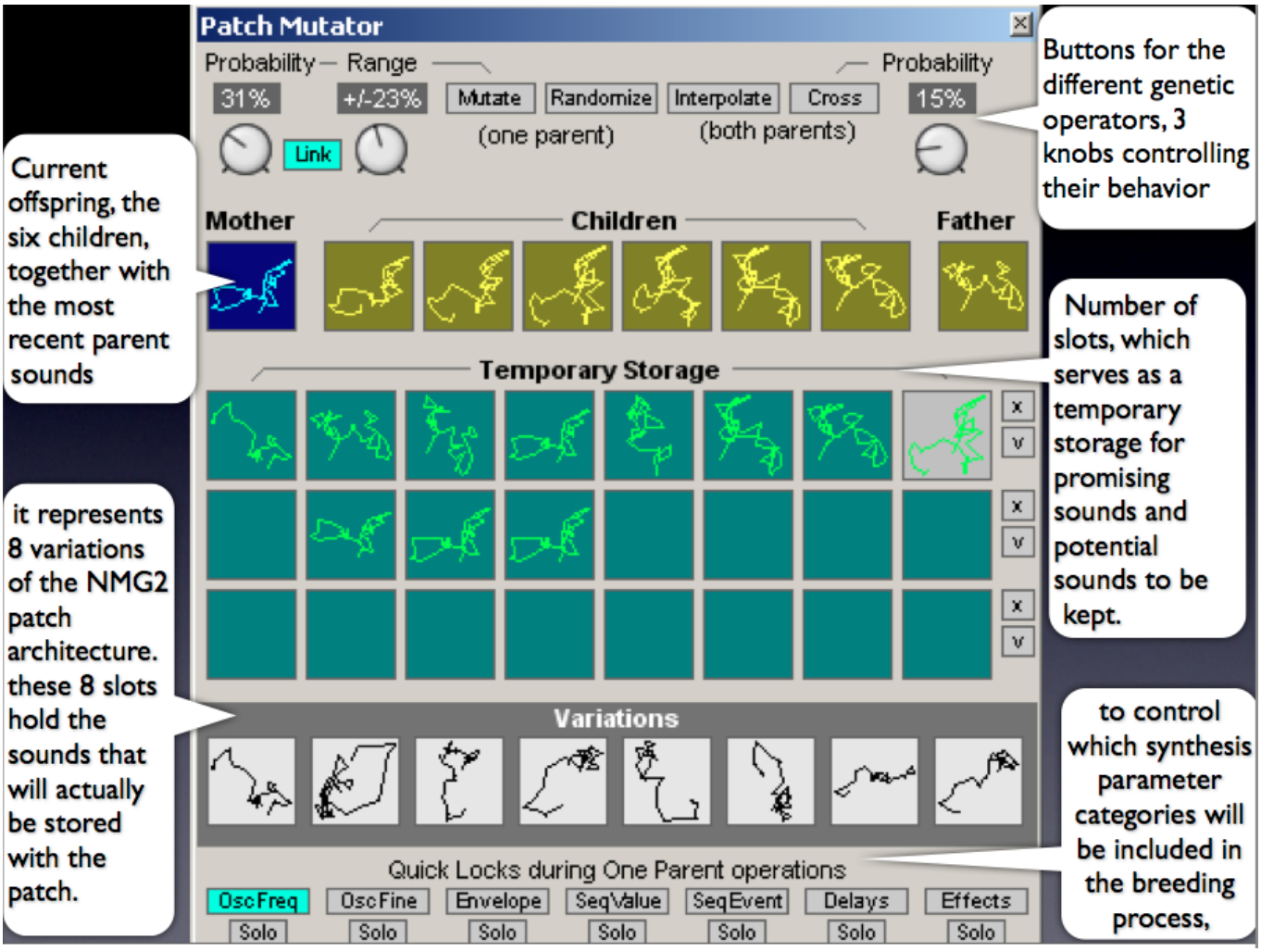


Figure 16: The Patch Mutator Window.

[*Dah07*]

- Each sound has a visual representation for giving a quick impression of the similarity of sounds and aiding memory.

- For fast and efficient evolution, all operations can be controlled with either through keyboard short-cuts or the mouse.

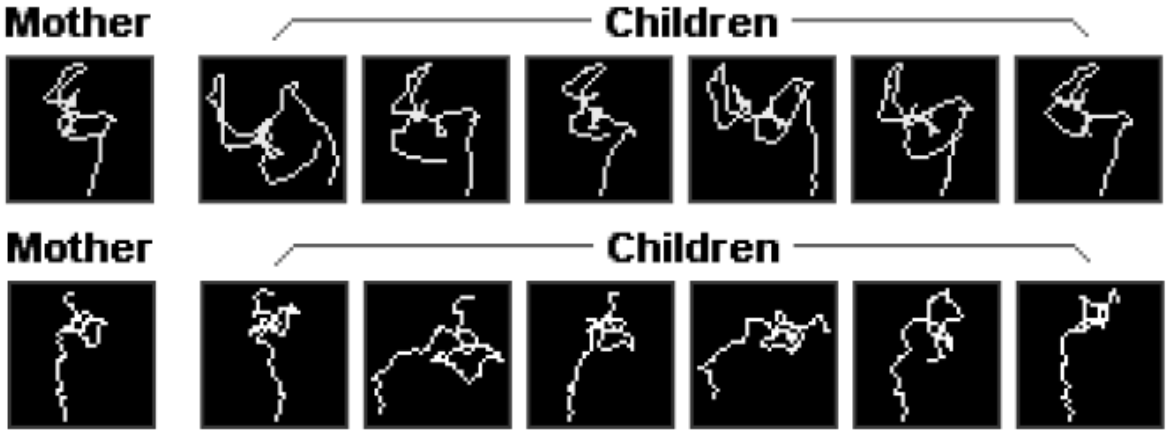


Figure 17: A wiggly line derived from the parameter values to help the user keep track of different sounds. It also indicates the degree of differences between two sounds, or rather between their parameter values up to a certain degree. For example, as seen above, the children are clearly similar, but slightly different from their mother.

[***Dah07***]

- To manage the fitness bottleneck problem, Dahlstedt created a temporary storage. Temporarily stored results are included in the evaluation, and can be reintroduced later in the breeding. Temporary storage can be seen as a flexible extension of the population.

- In addition to these main functionalities of this system, different parts of the sound space are most probably found by the help of a navigational strategy of an evolutionary tool. Chance and unpredictability are well-known devices for reaching uncharted field in art. The breeding process can be performed under the control of outputs. There are two approaches about the breeding process.

One user calls it a means to find inspiration for sounds that one would not think of naturally. Another user says the breeding process allows discovering some sounds from a patch that one normally would not have thought of making by customizing all the parameters.

Benefits.

- The detailed level of actual synthesis parameter values is hidden from the user so that the user does not lose his attention because of technical details and is able to focus on the sonic result, facilitating the sound design process. A musician concentrates on the high-level perceptive characteristic of the sound, such as brightness, intensity or rhythmic feel instead of forcing himself what to achieve and how to achieve it. When the user creates a number of sounds or musical textures, he can focus on the aural relationship between the sounds instead of how to formally create such relationships.

- The new way of adjusting synthesis parameters by ear, so to speak, also facilitates the actual design of sound engines for the musicians who previously were using existing sound.

- Patch Mutator makes modular architecture more accessible to beginner musicians who do not have detailed information about sound synthesis.

- Patch Mutator allows the users to focus on creating music for the project rather than designing patches.

- Patch Mutator makes creation of sound engines more accessible to less initiated users because they do not have to focus on exact adjustments of parameters.

- Patch Mutator helps the creative processes of the users.

- If a system had hidden sound engines, this system would be more difficult.

Users report that applications are a quick tweak tool and proper for adjusting existing patches. In a studio situation, if time is an issue, this tool is very powerful. In a short span of time (e.g. a few seconds) twenty or more variations on a sound can be ready to play.

- When a library or interesting sounds are created, the result is building the basis of a new piece or recording. In this context, there are several advantages compared to manual editing. Different sounds can quickly be tested. The composer can focus on aural relationships between fragments of material, and leave the formal relationship to the breeding engine.

- For sound design and sound synthesis, interactive evolution is evidently a feasible, new paradigm. It fulfills two main purposes. First, it makes sound design and synthesis much more available to and efficient for inexperienced users, who can control advanced synthesis techniques and complex sound engines by ear so that they can focus on the musical relevance of the sounds. Second, it provides a very powerful tool for both the composer and the advanced sound designer. Very complex sound engines can be reclaimed, comprising hundreds of parameters, and novel musical material can be explored and refined in a spontaneous way.

- It allows completely new ways of working with sounds, and represents a new approach to creativity and the creation of novel musical material, being based on the exploration of unknown spaces.

- The user can work with the sound in an innovative way.

- Until founded sounds are discovered, they do not exist. All undiscovered sounds (mathematical truths) should wait until they are proven.

Limitations.

- It can be frustrating when the user knows what exactly he wants or when the user exactly knows what is missing from a sound, and evaluation does not lead you there.

- It is not a perfect tool for all sound design situations, but it is a strong complement to manual sound design, often transcending it by far.

3.2.7 NEAT Drummer

NEAT Drummer [HS07] was implemented by Hoover and Stanley.

Main Functionalities.

- NEAT Drummer is programmed without any expert musical knowledge. You can give it to any MIDI and it will produce a drum pattern that follows the contours of a song.

- NEAT is the most certainly a generative software system for music composition. It specifically composes drum patterns for existing songs.

- NEAT Drummer uses Interactive Genetic Algorithms (IGA) and Neural Network; it interactively evolves neural networks.

Benefits.

- The benefit is that it can theoretically compose a drum pattern for any MIDI that you provide.

Limitations.

- The drawbacks are that it does not work in real time and requires MIDI.

3.2.8 Ossia II

Ossia II [Dah12] was created by Dahlstedt. It is a generative musical system which uses recursive binary trees as a genetic representation for score material. It is created for the interactive breeding of score fragments. The last version of the software focuses on the evolution of score material which is combined with automated fitness evaluation, making it possible for the system to autonomously compose and perform more or less complete miniatures for the piano.

Main Functionalities.

- The special feature of Ossia II is the **representation of the musical notes**.
- The **recursive feature** is unique in that it has not been used in other places and allows the users to allow to evolve complex musical pieces.
- It is a heuristic rule of thumb based analysis of how the choice is made, how the manual choice is made, how the selection of an individual is made, how the negative selection is performed and intensity changing. Ossia II has both a manual and automatic evaluation.
- Ossia II uses evolutionary algorithms to produce sound and music. Random variation is used during reproduction. Mutation and crossover random variations are used. Mutation means random changes in the data while crossover means combining two genomes by merging parts of two or more parent genomes.
- After the evolution, the greatest score from the last generation is achieved on an acoustic grand piano. This greatest score is saved on the disk both as a genome file in a custom format and as a MIDI file.
- When a user or a visitor plays a melody on the piano keyboard, this melody is translated into a genome data structure, used as an initial population for a new piece. This new piece is immediately performed.
- For each note, information about onset time (measured from the beginning of the list), pitch, amplitude (loudness of a note), note duration (time elapsing from the note that follows) and articulated duration (the actual duration of the played note) is stored.
- Three types of mutations are used as genetic operators. The first type is a random modification of an existing leaf node. The second and third types are insertion and removal of branching nodes in the tree.
- Ossia supplies three mechanisms for computing initial populations; random genera-

tion, human input and recombination from a collection of good examples. They can be used in combination or separately.

Benefits.

- Ossia II uses formal methods and the benefit of formal methods is helping composer to prevent artistic stagnation.
- EAs which are used in Ossia II can efficiently search a large number of possible solutions with the help of random selection and variation and can be used to solve specific design problems.
- The system can generate and perform pieces of music with the piano that could be considered human-composed contemporary pieces which vary in expression and style.
- The recursive mechanism in the system provides clear thematic structures, where each section of the generated material is rhythmically and melodically homogeneous and consistent.
- Different sets of generative parameters and fitness target ranges create interesting results.
- The musical output of Ossia II is modified with the help of flexible representation and different generative parameter sets and target ranges.
- The musical output that the system generates is structurally complex and the performances sound authentic, vivid, and musically convincing.
- The system generates music that sounds interesting and novel.

Limitations.

- Ossia does not incorporate much, if any, musical theory, of harmony or musical forms.
- There is a lack of correlation between superimposed structures. The tree operators let unconnected material to be superimposed regardless of their contrapuntal, harmonic, or rhythmic relationships.
- Arbitrariness of the large-scale forms is one of the limitations. The tree structure lets many different forms emerge, such as step wise processes, imitation with variation or repetition, and juxtaposition of different material. In short compositions it is not very important; however, it is a significant problem for long compositions.
- The statistical measures that form the basis for the fitness evaluation is not developed enough.

3.2.9 Application of Genetic Algorithms

Application of Genetic Algorithms [app95] was created by Jacob. This system accomplishes both the determinism of a rule-based system such as EMI and easiness of a stochastic process seen in M and Jam Factory.

Main Functionalities.

- The rules from the software system variations are shown in Figure 18.

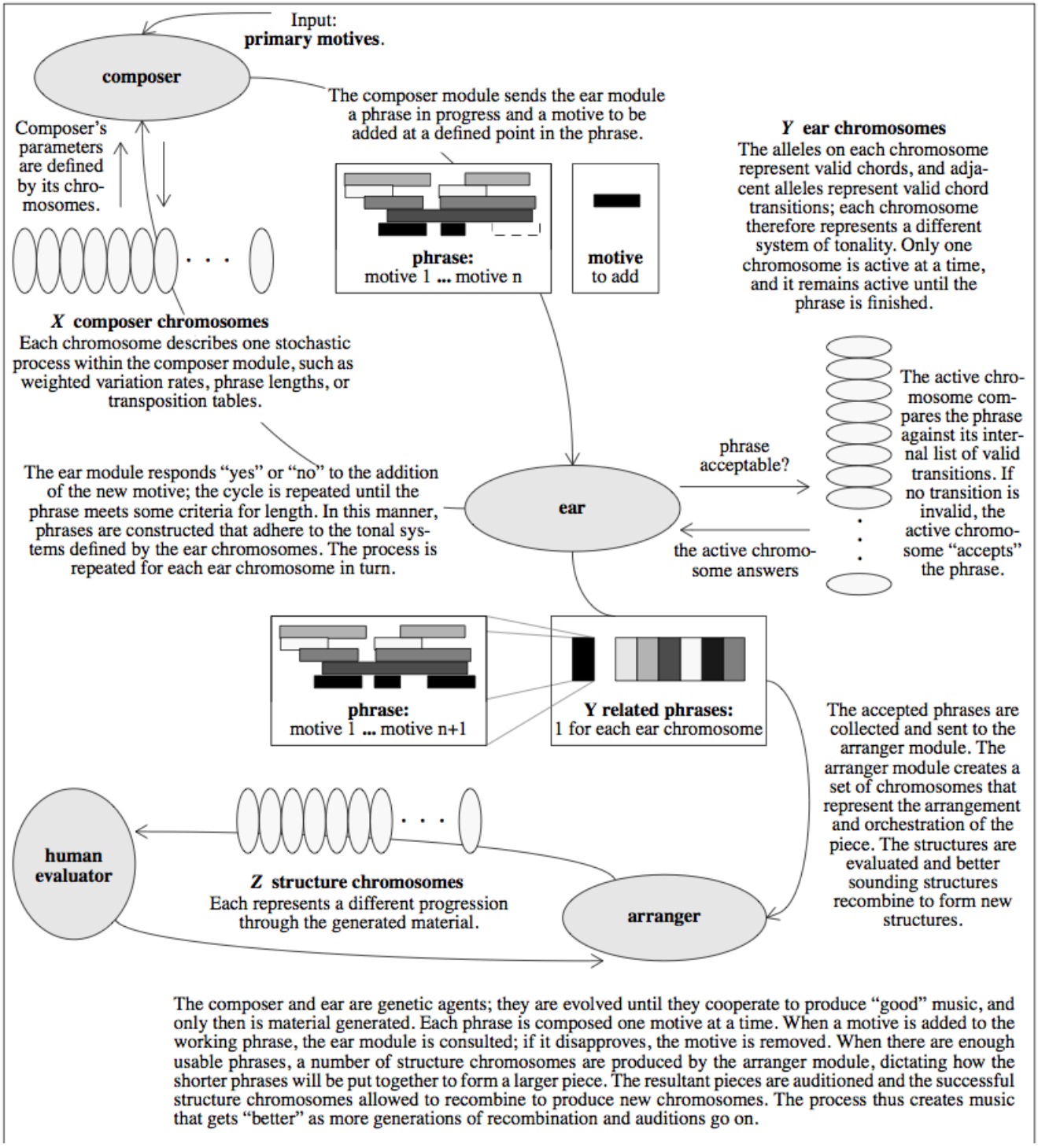


Figure 18: The algorithmic composition system variations.

[*Jac95*]

As is shown in the Figure 18, the process starts with primary motives. A motive is

a short, salient feature of a musical idea [Nat90]. It is a sub-part of a theme which is the material upon which part or all of a composition is based [Wik13h]. A **melody**, which is a linear succession of musical tones which the listener observes as a single entity, consists of one or more musical phrases or motives [Wik13e]. In other words, a melody is anything which is a sequence of tones, sequences of pitches and a rhythm. It is understood from the figure above that the system uses three different EAs. The first EAs composes phrases (melodic material), the second EAs controls the harmony, valid chords and chord transitions (chord progression), and the third EAs generates a form out of that material (adjust this material).
- This system is a clear example of making home compositions.
- In this system EAs are used to generate a set of data filters which describe adequate material from the output of a stochastic music generator. The goal of using EAs is to analyze the entire potential solutions to find one which fulfills the criteria. An important point is to arrange the set of all potential solutions. In this way one does not have to verify every solution, allowing the search to accomplish in a finite amount of time.
- Algorithmic composition system variations are used.
- This system applies to microtonal music, which uses microtones. The definition of a microtone is any interval of less than an equally spaced semitone which is the smallest musical interval commonly used in Western tonal music [MIL05] [Wik13f].

Benefits.
- The system is very flexible. It should be noted that the general representation of valid combinations does not depend on the choice of a twelve-tone octave. Microtonal vertical pitch combinations can be represented by using a different number of bits. Microtone refers to any interval smaller than a half tone. The system is accepted as flexible because of micro tonal vertical pitch combinations being used, which means the general representation of valid combinations depend on the choice of more than twelve-tone octave.

Limitations.
- The author describes the general problem of using EAs as well. GAs is one subtype of EAs. The biggest problem of using GAs is the size of the search space. Outstanding GAs music applications have restricted goals. The reason is that the problem domain becomes large instantly. Therefore, convergence to an adequate solution may take a lot of time. Horner solves this problem by changing one melody into another. Horowitz handles the rhythms which span only one measure, and Biles produces single melodies on top of given chord progressions. This study handles the problem in a very different way; instead of decreasing the amount of the problem domain, the genetic algorithms that are used in this application deal with larger building blocks.

3.2.10 Spieldose

Spieldose [SPVP07] was implemented by Sanchez, Pantrigo, Virseda, and Perez.

Main Functionalities.
- To generate and evolve a population of musical pieces, adapted GA is used as an

optimization method.
- During the genetic evolution, the user selects several good melodies according to subjective musical criteria. This process is repeated in each iteration. By this way, until the termination criteria are fulfilled, an initial population of automatically generated compositions is evolved.
- Spieldose tries to add criteria of musical composition into the Interactive GA.

Main steps of Interactive GA:
- Initialization: The initial GA population is created in this stage. This operation is specified to produce an initial set of musical works or melodies that are created using some specific knowledge from theories of music. In this application they have only produced 8-bar length musical melodies.
- Selection: At each iteration of the GA a human expert selects the best melodies in the population. The user chooses a variable-size subset of melodies according to his or her musical preferences or guided by the characteristics of a predetermined musical style.
- Crossover: Three possible crossover mechanisms are used to combine selected individuals (according to crossover probabilities) to produce a new generation of child melodies (offspring).
- Mutation: By this operation some chromosomes in the new generated melodies can be altered according to a mutation probability.
- Improvement: In this stage, some musical errors are corrected automatically. This stage is automatically performed and it strongly takes into account the considered criteria of given musical theory.
- Invasion: New randomly generated individuals are included in the population of musical works in this stage. It is needed to avoid the loss of diversity in the collection of melodies after a number of evolution iterations in the Interactive GA.

Benefits.
- One important contribution of this work is the variety and its effective implementation of the operators in the Interactive GA.
- Another contribution of this work is the complete graphical user interface (GUI) of Spieldose that offers the user appropriate functionality for the musical composition task. This GUI hides the implementation details of the interactive GA and makes offers to the user. These offers are as follows:
- Generation of melody population
- It enables the user to listen to generated music interactively.
- To select the best melody subset by a roulette wheel.
- At the end of each iteration, the best melody is saved in wav or text format
- Interactively editing a musical piece in text format is such that the corresponding audio is also modified at the same time (there exists a complete updated equivalence between both formats for each melody).

Limitations.
The included study provides no explicit limitations.

3.2.11 AMUSE

AMUSE [OE08] was implemented by Ozcan and Ercal. The snapshot of AMUSE graphical user interface is shown below:

Main Functionalities.

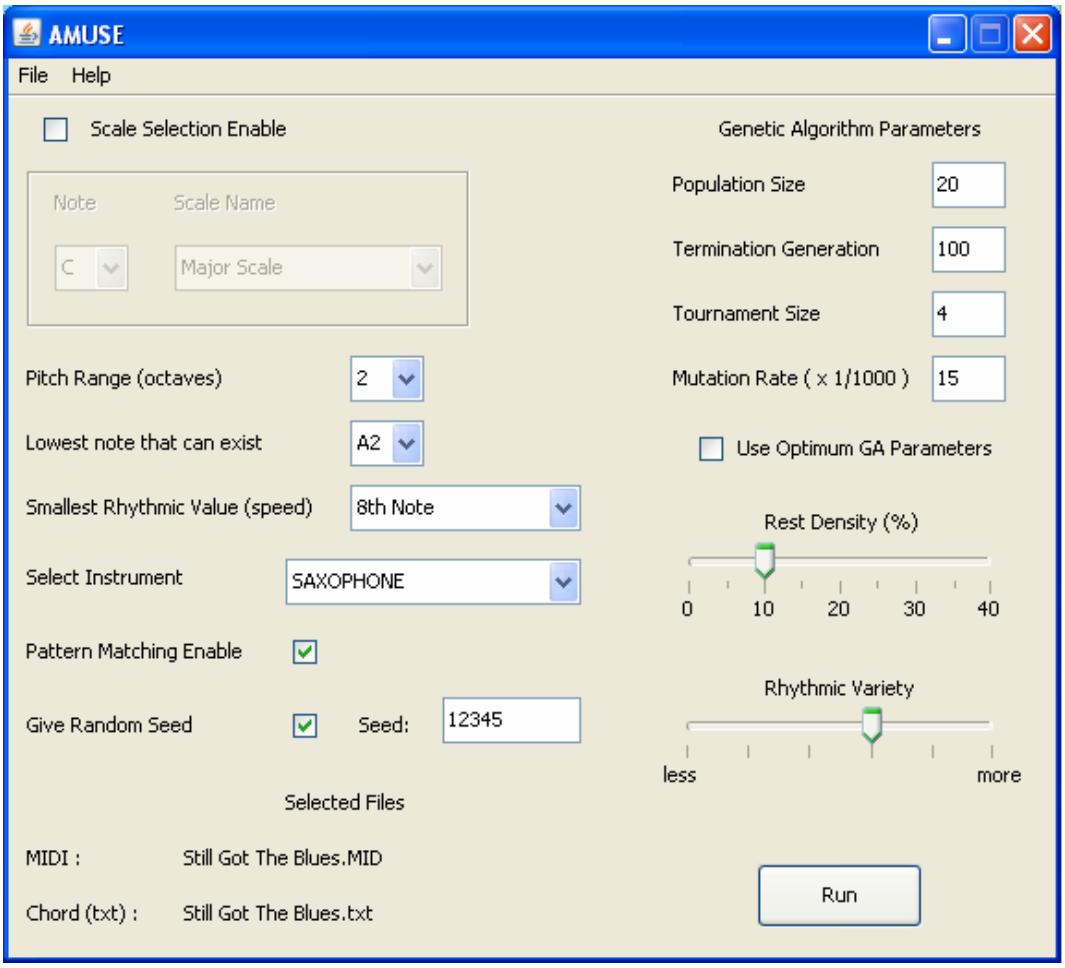


Figure 19: A snapshot of AMUSE graphical user-interface.

[***OE08***]

AMUSE is a system for generating improvised melodies over a musical piece given in a harmonic context. The software developers, who are the authors of the AMUSE, used a creative GA in this system, which is embedded into a Java user interface as a musical expert. To generate melodies automatically without a human feedback, AMUSE combines a modified representation scheme and different fitness objectives under a GA approach. With GA, they can use the traditional operators in AMUSE. They used both one point and two point crossovers with equal probability. One of them is utilized in each selected individual pairs in the population. According to experiments, this hybridized crossover performs better than applying just one type of crossover. All

individuals are chosen with a tournament selection method with a tour size of four. The population size is one third of the chromosome length. The offspring pool size is the same as the population size. The mutation rate is 1/ chromosome length. Objectives can be divided into two groups; core and adjustable objectives. The user cannot manipulate core objectives. However, the adjustable objectives are maintained according to the initial preferences of a user. This system gives a chance to the user to modify default parameters. Core objectives are; Chord note (f1), relationships between notes (f2), directions of notes (f3), beginning note (f4), ending note (f5), over fifth (f6) and drastic duration change (f7). Adjustable objectives are Rest proportion (f8), hold event proportion (f9) and pattern matching (f10). In the final stage, fitness function can be evaluated with use of all these ten functions.

Benefits.

The advantage of this representation scheme is that it is impossible to generate non-scale notes. AMUSE does not need a human for getting feedbacks.

Limitations.

The included study provides no explicit limitations.

3.2.12 Variations

Variations [Jac96] was implemented by Bruce L Jacob. The system called Variations is an algorithmic composition system. Algorithmic composition is a process which includes the generation of musical themes, phrases, or whole pieces by computational methods [BV99]. It is the method of using algorithms to create music [Wik13a]. Algorithmic composition system can be thought in two ways: First, it is considered as a cheat, a way when the composer needs musical material. Second, it is considered as a compositional tool which makes the work of a composer faster. This study explains the case for algorithmic composition as such a tool which is the second case.

The hard work type of creativity includes trying many different combinations and choosing one over others. This iterative approach can be expressed as a computer algorithm. The implementation is performed with two components: how to find out one's own creative process well enough to reproduce it as an algorithm, and how to implement software to differentiate between good and bad music. Algorithmic composition is an application of a well-defined and rigid algorithm to the process of composing music. It belongs to the designer of the algorithm, not the user of the algorithm. There is a clear goal for an algorithmic composition system: to reproduce the composer's creative methodology when the composer is in hard work mode. The result is a system which is tailored for this specific composer; if another person were to use the system to produce music, he or she would be composing this particular composer's music, not their own.

Main Functionalities.

- Variations is an algorithmic composition system which initiates to model the hard work type of creativity. The system was implemented to reproduce creative melodies which the author uses when composing music.
- The system works at the level of music motives. This simplifies the organization of

the music.
- It allows the user to pay more attention to create harmonic progression.

The compositional algorithm

1. specifies a number of primary themes (motives) to be used in the music composition.

2. composes phrases by creating motives and adding them one by one to the musical phrase. At each step, it judges the quality of the resultant phrase and removes the last motive if the combination is unfit.

3. makes motives by selecting them at random from the primary themes and motives already in the phrase, producing variations upon the selection.

4. When there are a large number of phrases, it joins them together into larger frameworks.

- As it is seen from the Figure 20, there are two primary software components which are COMPOSER and EAR modules. The COMPOSER creates musical material and the EAR filters out the bad ones. That is a producer-consumer paradigm which is popular in music composition systems. One side creates music, while the other side consumes it and critiques it, affecting the future output of the producer. Moreover, the figure represents the process. The composer takes a number of music melodies as input and composes a phrase from them motive by motive. The composer creates music by producing a variation on a previous motive and layering and sequencing it with the other motives. Each motive is either a variation upon a previous motive or a copy of a primary motive, a variation upon a primary motive or a copy of a previous motive. At each phase, the resultant phrase is tested by the EAR module and given a yes or no grade. If the EAR module does not like the piece, the COMPOSER deletes the motive and tries a different variation. If the EAR module likes the piece, the composition process continues.

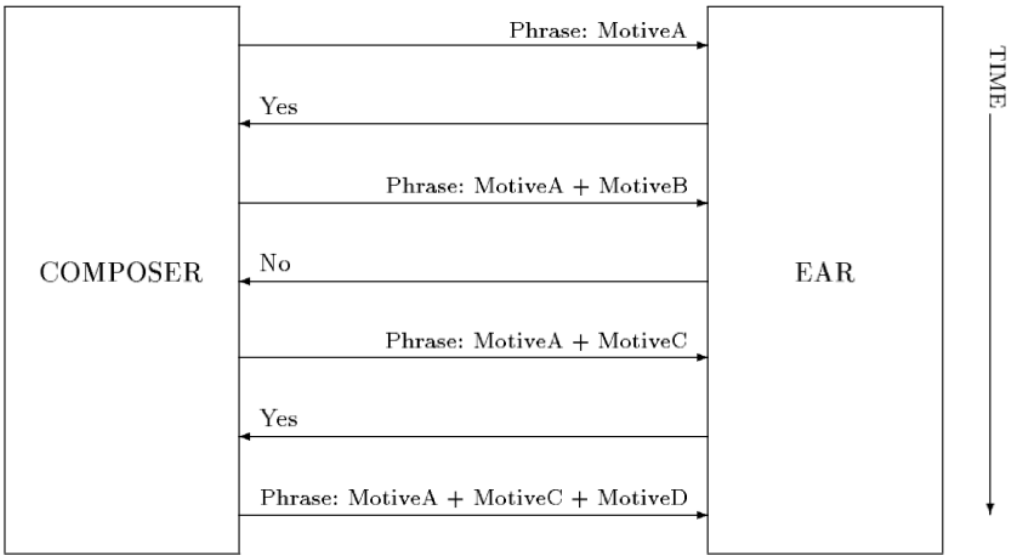


Figure 20: The producer-consumer relationship between the COMPOSER and EAR modules.

[***SMOB10***]

- The system uses an evolutionary algorithm, paired with domain-specific knowledge, and implemented as quite an interesting system.

Benefits.

- The system forms and manipulates the musical material at the level of motives, so that thematic development is inherent. A motive oriented internal representation provides and forces an implied structure on the compositional process; it gives the opportunity for easy coherent manipulation of the musical material so that the resultant thematic variations are neither unrecognizable nor trivial.
- Presented in this article, the music pieces produced with Variations system represent an illustration of the blurred distinction between a simple variation and originality. The system is allowed to create variations on the variations, and variations upon them so that the piece moves far from the original music source very quickly.
- The system uses algorithmic composition, which refers to a methodology for allowing a human composer to work more quickly. Its creative success depends on two phenomena:

1. A great match between creative methodology of the composer and the implemented algorithm.
2. An accurate mechanism for making a decision quickly about the viability of a specific musical phrase.

Limitations.

- Because each composer has a unique compositional process, one algorithmic tool does not fulfill the demands and requirements of many different composers.

3.2.13 Jive

Jive [SMOB10] was implemented by Shao, McDermott, ONeill, and Brabazon.

Main Functionalities.

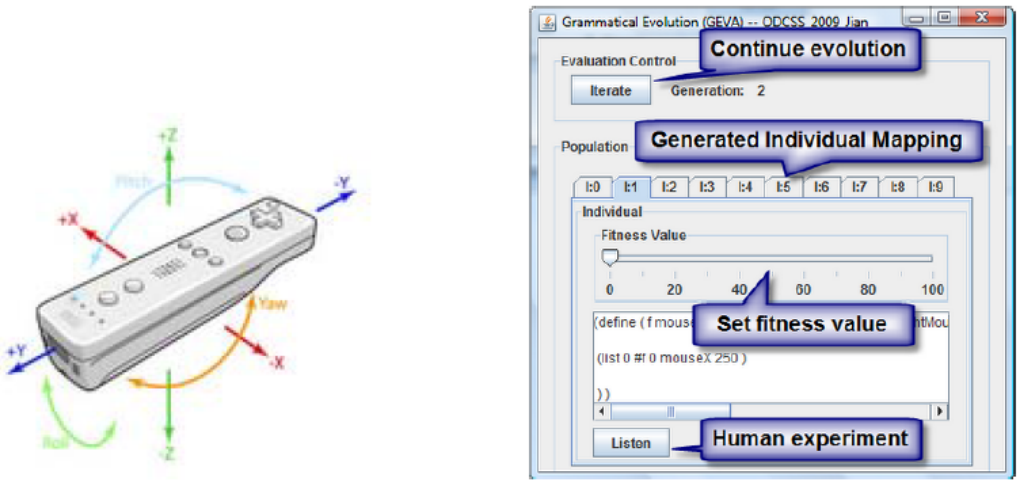
- The Jive system has four components: generative, interactive, virtual, and evolutionary.

Generative: Jive is a generative music system. In Jive, music is considered as a function of time where time is regarded as a distinct variable.

Interactive: Jive system allows the user to interact with the generative music as it plays. This allows the material to switch and develop as time passes. Accordingly, some continuous valued variables representing user input are added to the system.

Virtual: By using the word *virtual* the author means playable musical instruments which are implemented purely in software or electronics.

- The Wii controller and the graphical user interface of Jive are shown below:



(a) The Nintendo Wii Remote

(b) GUI used for auditioning and fitness evaluation

Figure 21: The Nintendo Wii Remote and GUI used for auditioning and fitness evaluation.

[***SMOB10***]

The user can use Jive in two ways: First, it is used through the mouse. It provides X and Y values. Second, it is used through Nintendo Wii remote control, which provides either an absolute position or accelerometer data in three dimensions as well as multiple buttons. The Nintendo Wii Remote, shown in Fig 21 (a), does not require a GUI.

Evolutionary: Interactive Evolutionary Computation that is used in the system allows the user to specify individual fitness values, or to perform selection directly. Because of generating multiple individuals in an iterative process, with the user required only to evaluate their quality, it prevents the need for the user to write equation directly when creating a piece. The Interactive Evolutionary Computation GUI used for

generation, auditioning and selection of pieces, and iteration of the algorithm, is shown in Figure 21 (b).

Benefits.

- The system creates the music in a sufficiently interesting way.
- The user is not allowed to control all the musical data. For example, while performing, it is not possible to insert or delete arbitrary notes. This apparent drawback has two advantages. Firstly, since the system generates material continuously, with no wrong notes and precise timing, the user is freed from low-level details. Secondly, the user obtains a higher-level type of control despite the lack of low-level details. The user has control of which behavior to switch to and when to switch. The user performs the gestures which correspond to desired sonic results, reacting to the current musical context created by the ongoing algorithm and by his previous actions. This ability allows censoring undesired sections and creating complex musical syntax, such as call-and-response patterns among several behaviors. The users' control of timing is sufficiently fine to allow the behaviors to work together.

Limitations.

- The system is clearly lacking in the area of rhythm.
- It is not user-friendly: most musicians cannot perform symbolic regression in their heads while composing.
-While performing, the user can not insert or delete arbitrary notes.

3.2.14 Birdsongs

Birdsongs [For12] was implemented by Fornari.

Main Functionalities.

This paper presents a basic study on the development of an evolutionary algorithm for the generation of an artificial soundscape of birdsongs. This system is built by genetic operators. These operators dynamically generate sequences of control parameters for computational models of birdsongs, given by the physical model of a syrinx. This system can emulate a wide range of realistic birdsongs and generating with them a network of bird calls. For the generation of artificial soundscape, the EAs system was implemented as a patch. Individuals are instances of an auxiliary patch (ind.pd). Each instance of ind.pd generates an individual. Each instantiation is an independent physical model of the syrinx. They use a psychoacoustic distance as the fitness function, through which metric individuals inside the population are selected. Selection process measures the distance between each individual in the population. If the selected ones have a higher distance than the average distance for all individuals, they are eliminated. Thus, individuals in the population are similar to each other as far as the perception of their birdsongs concerns. They use crossover and mutation as genetic operators.

Benefits.

This study has the possibility of creating a similar yet variant sound texture. This system is able to generate artificial soundscapes compounded by synthesized birdsongs. It allows the interactivity of multiple users. This creates a feedback between users

and the EA system. Besides, it can be enhanced by the usage of a computer model to generate graphical objects corresponding to the sound objects created by the individuals into the population.

Limitations.

The included study provides no explicit limitations.

3.2.15 Neurogen

Neurogen [GB91] was implemented by Gibson and Byrne.

Main Functionalities.

In Neurogen, a set of Neural Networks are used to capture conceptual ideas. Those ideas build *good* music and this knowledge is then used to direct a search for the ultimate composition. The Neural Networks cooperate to produce a heuristic value. This value represents the worth of each of these musical fragments and is then used to evolve better compositions based on fragments with high fitness values. The software developers of Neurogen construct a Neural Network model; this model can learn the *good* characteristics from a set of *good* and *bad* model compositions. Once the Neural Network has completed its learning phase, it can be used as a guide for the Genetic Algorithm. After that, they apply genetic operators of reproduction, crossover and mutation to generate better compositions based on the heuristic values provided by the Neural Network.

Reproduction randomly selects two mates in the old population by the help of high fitness values. Crossover randomly selects a bit position within the two mates from where their bit information is to be exchanged. Mutation adds new random genetic information to prevent early saturation.

Benefits.

The system just focuses on a particular form and composition style. This form and style of musical composition has limited constraints. This means that there is less computational complexity.

Limitations.

The included study provides no explicit limitations.

3.2.16 GeNotator

GeNotator [Thy99] was implemented by Thywissen. It is a computer-assisted tool which uses evolutionary algorithms for composing music.

Main Functionalities.

- GeNotator has two different interactive levels: meta-composer and gardener. A meta-composer is interested in analytical understanding of the form and structure of a composition and what one wants to achieve, and uses the GA to assist the creative process by generating and refining ideas. In contrast to a meta-composer, a gardener is just interested in what she or he likes. A gardener does not need to know how the music is composed.

- Structurally, GeNotator has Genotype Structure Definition (GSD) in the centre of its architecture. The GSD is a data structure that packages a user-defined music grammar. The user specifies this grammar either textually or through a set of editing windows.
- Once defined, the GSD serves as input to the Form Space Manager. This consists of an interactive genetic algorithm that takes the chromosome structure of the GSD and allows the user to breed and mutate phenotype instances of it.
- In order to evolve favored instances, the user can judge and score results and continue to produce iteratively.
- GeNotator enables the user to mix and match between a text-based grammar and the graphical approach within the same project.
- The GUI components of GeNotator are very powerful.

Benefits.

- By the help of powerful GUI of the GeNotator, the user can interactively compose music.
- The user can modify genetic operators via this GUI.
- GeNotator is a fairly flexible tool for the user.

Limitations.

The included study provides no explicit limitations.

3.2.17 Mezzo

Mezzo [Bro12] was implemented by Brown. It creates soundtracks for computer games in real time.

Main Functionalities.

- Main motives of the game music are related with elements and game characters. These main motives are mapped into various musical forms. These forms are recognized by different amounts of harmonic tension and formal regularity.
- For each round of game, main motives of the game music were input to be related with game elements and characters, and a set of clues was written. These clues include a set of time points at which a new set of game data would be passed to Mezzo to demonstrate the action of the game trace.
- Music composition in Mezzo is made in two steps: 1) Build forms and 2) Deform them according to stochastic constraints. Both of these processes generate artistic properties in the music being composed. During the form building phase, chord progressions with the appropriate length, cadence type, formal organization and amount of harmonic tension are composed and forms are constructed by mapping proper motives and accompanying textures to them. In the second stage, each form that has been composed is mapped into a data structure that sets the stochastic constraints on its formal regularity, deciding which formal sections will be omitted or repeated. This stochastic model is a generalization of the approaches of deformation used by Classical and Romantic composers, and initiates to keep continuity with their methods while designing them in an interactive and modern context.
- The only time when Mezzo uses a genetic algorithm is when it makes harmonic pro-

gressions. The author mentions that he used this because making these progressions is a highly constrained problem over a very big search space, and it needs to be done quickly. At first he used the constructive algorithm that David Cope uses in EMI, but that approach takes too long.

- Moreover, there is some research that the ideas like narrative, form, expression, and meaning in music are biologically based:

Embodied music cognition is a direction within systematic musicology focusing on studying the role of the human body in relation to all musical activities. It studies the human body as the natural mediator between mind and physical environment [Wik13b]. Mezzo uses ideas of musical narrative to compose and then deform forms. In the interview the author mentions that he never spent much time looking into the research on whether these ideas are biologically based, however the author finds it very interesting.

Benefits.

- Mezzo has open-ended setting of a game. Each time a form is stated, it is organized differently from previous times. There is no pattern of the way the organization changes from statement to statement.
- It composes fully realized music (not just a single melody or a drum beat, but a full musical texture) in real time for a game, and this music is adapted to the game.
- Also, it uses musical theories of form and meaning that have not been used in computer-generated music till now.

Limitations.

- As it is stated in the benefits above, Mezzo has open-ended setting of a game. Each time a form is stated, it will be organized differently from previous times. This causes a certain quality of irregular formal organization.
- It does not orchestrate the music (yet), and so the sound output is not that interesting. The author mentions that he has been working on these issues.

3.2.18 Sonomorphs

Sonomorphs [Nel93] was implemented by Nelson.

Main Functionalities.

- The aim of this study was to find optimum methods for structuring musical organisms.
- An algorithm generates musical structures and interprets the genetic code. This genetic code is passed to each generation.
- The genetic code is embedded onto musical parameters. The composer uses this code for subjective aural evaluation.
- The genetic model of evolving rhythmic patterns uses a bit summing test. If a bit is switched on, a note is articulated; if a bit is switched off, a rest is made. The individuals play notes on approximately all of the pulses after a few generations.
- The bits are combined with crossover method. One of the two parents is chosen with a coin toss for beginning the breeding. A coin is tossed again when the first pair of bits is considered. If it is tails, no crossover occurs. The first bit for the child's genome is taken from the first parent. If the coin turns head up, a crossover occurs and a bit is

taken from the opposite parent.
- Selection of parents is performed by random walk method.
- As it is seen from Figure 22, the parent in the middle which is shown in the bold square box has been mutated and fourteen new sisters have been produced.

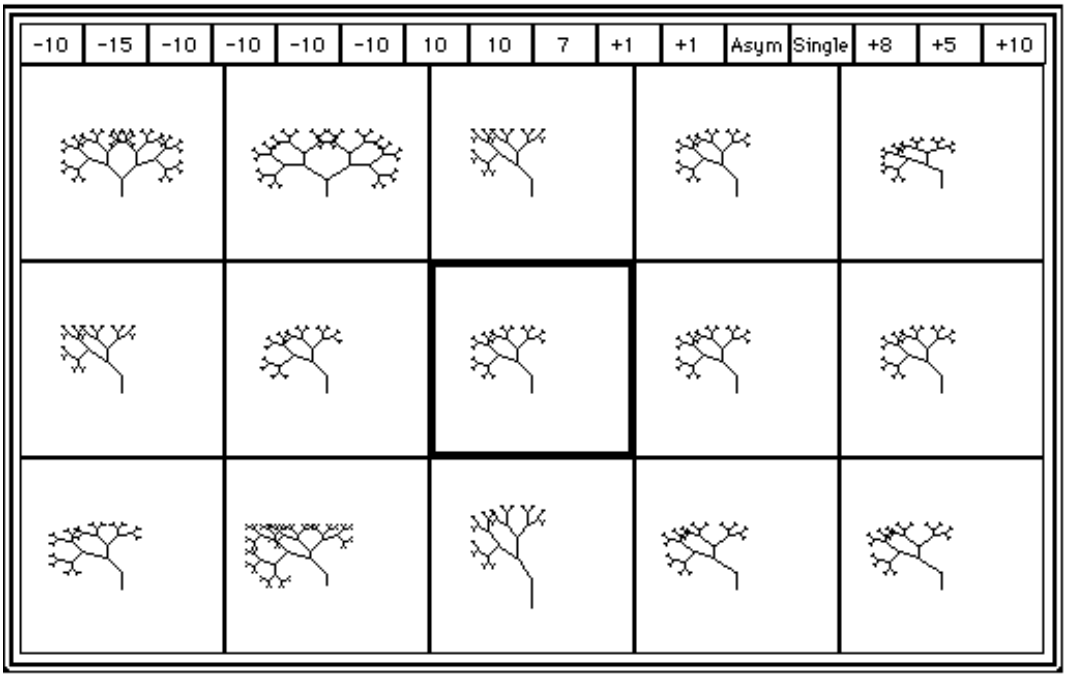


Figure 22: Dawkin's Breeding Window (Breeding With One Parent).

[***SMOB10***]

Benefits.
- They used mutation and migration operators in this study. These methods provide diversity on a gene pool.
- The proposed system has a lot of controls on graphical user interface; these controls provide a multiple toggle.

Limitations.
- The breadth of the field is so great that it is difficult to focus for very long on simple examples and the extraction of basic principles.
- This system is probably not a powerful tool for making large compositions. The operations are certainly too limited and too simple to make sophisticated musical utterances.

3.2.19 Vox Populi

Vox Populi [MMVG02] was implemented by Moroni, Manzolli, Zuben, and Gudwin.

Main Functionalities.

- Vox Populi is an evolutionary based system for composing music in real time. A population of chords is decently codified according to the MIDI protocol.
- A fitness function is defined to find the best chord in each generation. The best chord is selected as the next element in the sequence to be played. Each new generated chord is a new sound palette. Musicians can use this new sound palette to continue the music evolution.
- Vox Populi becomes a musical instrument, but unlike a traditional instrument, Vox Populi is able to create its own sound chord population and to provide choice criteria (music fitness) simultaneously.
- It allows the user to modify the fitness function by means of four controls: These controls are melodic criterion, the duration of the genetic cycle and musical rhythm, the set of octave ranges to be considered and the time segment for each selected orchestra.
- It uses the computer and the mouse as real-time music controllers and produces dynamic musical structures based on evolutionary models.
- This system is a new interactive computer-based musical instrument.
- It has a strong graphical interface to change the musical evolution.

Benefits.

- By graphic controls (pad and sliders), the system becomes user-friendly to manipulation of the fitness and of the sound attributes.
- Evolutionary computation is used to stimulate the user with novel sounds. It allows the user to respond.
- By the features of Vox Populi, this system enhances the user's music capabilities and marks this system as the state of the art in computer music.
- All controls of Vox Populi are available for real-time performance, allowing the user to play and interact with Vox Populi's music evolution.
- The interactivity of Vox Populi emphasizes aspects of musical practices in the scope of human/machine interaction.

Limitations.

- The included study provides no explicit limitations.

3.2.20 Rhythm Generation System

Rhythm Generation System [Hor94] was implemented by Horowitz. To be able to understand the benefits of this system, phenotype and genotype phenomenon should be described. Phenotype is the set of observable characteristics of an individual resulting from the interaction of its genotype with the environment [Fen]. Genotype is the genetic constitution of an individual organism [Ge1].

Main Functionalities.

- The set of rhythms satisfying the criteria of the user is represented by a Boolean formula.
- The rhythms which are created by the system are played for the user, who assigns fitness values to each generated rhythm according to his satisfaction. Then the system

uses GA selection, reproduction and mutation operators.
- There are **objective** functions through which the system automatically evolves generation of rhythms. These functions are syncopation, density, downbeat, beat repetition, cross rhythm and cluster. The user calls these functions to explore creative rhythms.

Benefits.

- In this system, rhythms are one measure long sequences of notes. The software developer only deals with a specific subset of the tremendous class of rhythms. The goal is to obtain a well defined domain for the application of the learning algorithm. The benefit of this minimization of the domain is that a rhythm phenotype can be viewed as a simple vector so that the set of rhythms satisfying the criteria of the user can be represented by a boolean formula.

Limitations.

- Although implementation of IGA is quite simple, the implementation of appropriate fitness functions is difficult and not efficient enough in this system. Fitness functions largely determine the musicality of the output.

3.2.21 Music composition system with human evaluation as human centered system

Music composition system [UO03] was implemented by Unehara and Onisawa.

Main Functionalities.

This system is an interactive music composition system. The human plays a major role in judgment, evaluation, recognition and emotion steps. The composition has 3 main procedures.

1- Two hundred chromosomes are generated based on general music theory.

2- A user listens to 20 musical works chosen from 200 works and performs three types of evaluations such as total evaluation, partial evaluation, and the choice of the best work.

3- The system performs EAs operations on 200 chromosomes reflecting these evaluations.

Procedures 2 and 3 are repeated until a musical work projecting users' evaluation is achieved. These steps show that this is a human centered system.

This paper uses the IEAs to construct a music composition system. Human evaluation system is embedded into the EAs process as a fitness function in this system. The system generates chromosomes using the EAs operations based on user evaluation. Then, the composed music gradually becomes the user's best one with each generation. Unehara and Onisawa used two types of chromosome structures; A and B. The information of the whole music structure is held in structure A and melodic information is held in structure B.

Benefits.

This system is a human centered system and the human has an important role in construction of musical compositions. Because of this, the system can be easily personalized according to the choices of the user.

Limitations.

In addition to several advantages of human centered system, there are two important limitations of this system: One is the users' fatigue. In this approach, users have to listen to the music one by one. Users have to take a lot of time listening almost the same melody as he or she already listened. The other limitation is the solution space, that is, musical variation. Users have to choose many aspects of music, such as instrument, background music, tempo, and so on. This is a human depended system and does not guarantee perfect melodies. It depends on musical background of the user. The researchers have to solve at least above two, by using somewhat ingenuity or breakthrough.

3.2.22 Beads

Beads [Bow11] was implemented by Bown.

Main Functionalities.

The approach of the author is very ambiguous with respect to specific functionalities, e.g., tracking beat, finding key, continuing on the style of the performer. None of these things are done by the system. The author approaches the final system more as a creative composition of a system by himself. Performers have reported to have a great experience of interaction with it in terms of the surprising responses it generates.

Benefits.

The author has reported that the system is beneficial in terms of design methodology. He does not believe in creating an ultimate live algorithm, and he thinks the innovation in design methodology is currently required. In this case because the behavior is the product of evolution towards a targeted fitness function, its design is detached from the designer (the author), so it has a kind of functional autonomy (sort of).

Limitations.

The author explains that the system is hugely limited in many dimensions. Since he treats making the algorithms a form of composition, it is like asking what the limitations of a given melody are. However, it is relatively flexible as a kind dynamic behavior that can be adaptively used in different contexts.

3.2.23 ChaOS

ChaOS [Mir02] was implemented by Miranda.

Main Functionalities.

- The research is about musical forms originated and evolved in artificial worlds. The music making term is used for both creating and listening to music. Natural selection in biology is the effect of new music making.
- Social evolution, a very complex phenomenon, is considered. Music is the interaction of agents engagement in music making.
- In transformation, the entity information is preserved. Co-evolution used for interaction between various transformation and selection causes the system to be more

complex.
- In self organization, with feedback, a fluctuation is strengthened and becomes more predominant.
- Computer sound synthesis technology has allowed sound control fundamentally. Granular synthesis involves tiny sound granules and exhibits sensible movement and flow. Self organization is used for controlling the evolution of the granules.
- Cellular Automata are modeling techniques being used in systems in which space and time are represented discretely. CA is implemented as an array or matrix of cells and associated with a color.

Benefits.
- ChaOs is an acronym for Chemical Oscillator, an adapted version of a Cellular Automata used to model the behavior of a number of oscillatory and reverbatory phenomena. An oscillator needs three parameters to function: frequency, amplitude and duration.
- In order to produce sounds, ChaOs resembles evolution of harmonics of acoustic instruments that converge from a wide distribution to oscillatory patterns.

Limitations.
- It does not work in real-time.
- The last version is for an older operational system which does not run anymore.

3.2.24 Sound Gallery

Sound Gallery [WT02] was implemented by Woolf and Thompson.
Main Functionalities.
- The Sound Gallery has two important algorithms:

1 Hill-Climbing Phase

Four initialization genotypes are generated, one for each of the four sub populations. Hill climbing then commences, with each sub-population working in parallel to the other three. The initialization genotypes undergo repeated mutations, generating new genotypes which present new TRAC configurations. The Sound Gallery uses the Zetex Totally Reconfigurable Analog Circuit (TRAC). Each new genotype is evaluated and assigned a fitness value. When a mutation is evaluated to be fitter than, or equally fit to, the parent genotype from which it was derived then this mutant genotype is stored as the next member of its sub-population and will be used as the source for subsequent mutations.

2 Island Model Genetic Algorithm Phase

A linear rank based selection is used to select two parent genotypes from each island sub-population. Child genotypes are derived from each pair of parents through the application of mutation, replication and crossover genetic operators. The TRAC development board is reconfigured so the circuit specifications represented by each of the new child genotypes are physically manifested in silicon. Each of the four circuits on the TRAC development board are then allocated fitness values, and the new genotypes replace the least fit members of their respective island sub-populations. This sequence

of events presents one iteration of the algorithm.

- To test Sound Gallery, software named galSim which simulates the movements of a number of people as they navigate an artificial gallery space was written. This software provides genetic algorithm with automatically generated fitness values, making it possible to run experiments with the system outside of the gallery setting.

Benefits.

- Volunteers made experiments about Sound Gallery. They enjoyed and found it enthusiastic while exploring different, varied and interesting sounds. Sound Gallery captured the attention and imaginations of a group of volunteer participants. The project proved itself capable of producing a lot of large repertoire of interesting distortion effects and sounds.

Limitations.

- The included study provides no explicit limitations.

3.2.25 DOT

DOT [Ros13] was implemented by Roscoe.

Main Functionalities.

- The artist and the invited guests are able to create all sounds and images live.
- All the live audiovisual parameters can be controlled using joysticks.
- Each part of the performance has a special score, where the functionalities of each button are shown to the players.
- The system is autonomous and does not need a computer to work.
- All images and sounds are linked to the performance concept through metaphorical relationships.
- There are no pre-recorded images or sounds. All the content is generated in real time.

Benefits.

- The players can participate without previous knowledge of the main principles of the performance.
- Winning the game is not the main objective, but participating in and contributing to the success of the performance is the main goal.
- The instrument is autonomous and does not need a computer to work.

Limitations.

- The generated images have low resolution (400x300).
- The sounds are limited to 64 voices that can be chosen from a sine wave or noise.
- The number of sprites and colors on the images are limited.
- All images have to be converted into binary information in order to enter the Arduino Environment.

3.2.26 MusicBlox

MusicBlox [GJ03] was implemented by Gartland-Jones.

Main Functionalities.

- The MusicBlox is a project which uses blocks like children's wooden building blocks. These blocks are combined together in a similar way to make physical structures.
- When the first block is created, it sends its result to the second block and the second block recomposes itself, and then the second block sends its result to the third one and the third block recomposes itself. This process continues iteratively. At the end, the collective music of the structure is transformed.
- The home music of the block is used to create a population of identical phenotypes, which build the initial population.
- Mutation and crossover operations are performed on the selected population member.
- The similarity between the phenotype and the target is defined as fitness function. If the fitness value for the mutated population member is higher than the lowest fitness found in the population, it replaces the low fitness population member, and is stored as a musical point on the evolutionary path to the target; otherwise, it is discarded.
- By each block possessing its own GA, and passing the output of one as the target input of another, the whole system becomes a kind of distributed IGA.

Benefits.

- This system is a kind of distributed IGA because each block possesses its own GA, and passes the output of one as the target input of another. This means that MusicBlox has an interactive learning process. And learning process depends on user's needs.

Limitations.

- The included study provides no explicit limitations.

3.2.27 GenBebop

GenBebop [SA94] was implemented by Spector and Alpern.

Main Functionalities.

- In this study they proposed a system which produces bebop jazz melodies from a case-base of melodies with genetic programming.
- Their fitness function is based on user-provided critical criteria. Aesthetic judgment is a problem of constructing artists systems. In this study, they tried to by-pass aesthetic judgment. For this purpose, the proposed system takes user-provided criteria and guaranties to produce proper melodies.
- Spector and Alpern just used reproduction and crossover functions of genetic programming. The reproduction operator selects the best individual and keeps it into the next generation. In addition, by crossover operation; they provide variations of the population.

Benefits.

- As mentioned before, the major problem of this kind of automatically generated melodies is aesthetic judgment. The proposed study takes criteria and constricts them as input, so they do not need to judge the output.

Limitations.
- The lack of robustness is a limitation of this study.

3.2.28 GenJazz

GenJazz [BD08] was implemented by Bäckman and Dahlstedt.

Main Functionalities.
- The aim of this study was to create computer based jazz improvisation solos.
- They used interactive evolution in their study.
- The computer has generated solos. Then these solos have been imported into a musical environment. After importation, the result can be listened and evaluated.

Benefits.
- This system provides interesting and unexpected artistic outputs.
- Throughout history, there have been a lot of rules for jazz music solos and all outputs of this study fulfill the conditions.
- Using computers in producing jazz music helps your mind to welcome new thinking.

Limitations.
- The included study provides no explicit limitations.

3.2.29 GenDash

GenDash [Was07] was implemented by Waschka.

Main Functionalities.
- GenDash is a program which has been revised several times according to needs of the authors. In general it has been used to help compose pieces of music. For some pieces, GenDash provided the total algorithmic support. For other pieces, the author might have used GenDash for one aspect of the work, such as the instrumental part of a composition, while employing a different program and algorithm for the electronic portion.
- GenDash has ten attributes; An individual consists of a measure of music, all individuals that are born in any generation are performed, the fitness function is random, only one crossover point is used for each breeding, space is set aside for individuals that are unheard in the current generation but may appear and/or breed in a later generation, space is set aside for an intact individual that may breed in the current generation and in a succeeding generation, individuals within a single generation can mate with more than one other individual and/or mate with the same individual more than once, mutations can occur and finally, the composer chooses the initial population.

Benefits.
- GenDash is a flexible program and the user can use this program according to his/her needs.
- The user is able to create a significant body of new art music based on evolutionary computation.

Limitations.
- After a regular concert series have been arranged, the concert hall has been rented and the performers have been paid to play, the amount and cost of rehearsal time have to be considered as an important budget factor. This factor can be a limitation for the composer.

3.2.30 MaestroGenesis

MaestroGenesis [SA94] was implemented by Hoover, Szerlip, Norton, Brindle, Merritt, and Stanley. Amy Hoover has a mainly contribution of implementation for both NEAT Drummer, the reference number of the included study [7], and MaestroGenesis. She mentions that NEAT Drummer is most certainly a generative software system for music composition and it specifically composes drum patterns for existing songs, but its successor, MaestroGenesis, can also add pitched accompaniments to as little as a single monophonic starting melody. Homepage of MaestroGenesis is at [atUoCF12].

Main Functionalities.
- This study enhances the state of the art for a computer-assisted approach to music generation called Functional Scaffolding for Musical Composition (FSMC). It is a method for generating music which concentrates on harmonization and accompaniment. It is based on the idea that music can be represented as a pattern which is a function of time, i.e. f(t) [atUoCF11].
- FSMC has a powerful representation like creative combination, exploration, and transformation of musical concepts.
- FSMC represents music as a functional relationship between an existing human composition, or scaffold, and a generated accompaniment.
- This relationship is encoded by Compositional Pattern Producing Network (CPPN). CPPN is a type of artificial neural network.
- A human user can generate polyphonic compositions from a single, human-composed monophonic starting track.
- FSCM facilitates creative exploration by helping the user construct and then navigate a search space of nominee accompaniments through a breeding process similar to animal breeding. This process is also called interactive evolutionary computation.
- MaestroGenesis can add pitched accompaniments as little as a single monophonic starting melody.

Benefits.
- The user can easily generate polyphonic compositions from only a single original monophonic track provided by the user.
- A human user without any musical expertise can use the FSCM system effectively and then gain accepted results.

Limitations.
- The included study provides no explicit limitations.

Summary of the results is presented in Appendix F. It is structured mentioning the reference number of the software system, title of the included study, name of the software, biologically inspired computational method used, different music types, programming language, programming environment, main functionalities, benefits, and limitations for each system respectively. Moreover, the background information about the authors of the included studies and the places which the programs are used are mentioned in Appendix B.

4 Discussion

As previously stated, this systematic review is about examining existing work for methods and approaches used, main functionalities, benefits and limitations of 30 generative music composition software systems (GMCSS). The research questions are: 1) What methods and approaches have been used in implementation of generative music composition software systems so far?, and 2) What is possible to know from the literature about main functionalities, benefits, and limitations of generative music composition software systems?

The results of this SLR are based on 30 selected included studies. The databases which are ACM Digital Library, IEEE Xplore, ScienceDirect, Springer Link, Web of Knowledge, and Wiley Interscience Journal Finder provided studies necessary for this thesis. This thesis is the first stage of the research about GMCSS and can obviously be studied more in depth. Further research can be conducted on these research questions.

As previously introduced in Exclusion Criterion No 1 and No 2 the studies which are not computer science related and the studies which do not offer a method were excluded. If the second exclusion criterion had been computer music related papers, biologically inspired algorithms would have been searched more in-depth. Searching what methods used in the music systems did not allow in-depth research about the algorithms being used.

This paragraph gives information to the reader about how the results should be read. Tables 5, 6, 7, and 8 show the results of biologically inspired computational methods used, music types, programming language, and programming environment respectively. In all these tables the second column named **Ref** involves the included studies that are systematically reviewed. To observe which number refers to which software name and paper please see Appendix D. The third column, which is **Total**, represents the summation of the included studies belonging to a specific method (e.g. evolutionary algorithms are one of the specific biologically inspired computational method). For example, in Table 6 rhythm generation is done by the included studies (which are software systems as well) numbered [7] and [20] and summation of these studies are 2.

An SLR, which is more comprehensive than conventional surveys, is presented to make the study valuable for people who are working on generative music composition systems for both practitioners (software developers) and researchers. Software developers will get inspiration by reading main functionalities, benefits, and limitations of 30

software systems in terms of requirement specifications (functional and non functional requirements) and design decisions of the software that they will design. Moreover, they will get an idea about which programming languages, programming environments, music types, and biologically inspired computational methods can be used during implementation process. Researchers who are interested in the area will be aware of the current limitations (e.g. fitness bottleneck) and solutions about the specific problems. By considering the current limitations of the 30 included studies, they can find appropriate solutions for these limitations.

Based on the results of this research, 5 suggestions to further field are mentioned below:

1) The beginner users who expect Complete Composition from the software are suggested to use [3], [5], [17] or [29] because they all use evolutionary algorithms without user interaction.

2) The advanced users who expect Complete Composition from the software are suggested to use [10], [13], [16], [19], [21], [26] or [30] because they use interactive evolutionary algorithms which require user interaction in a rapid way.

3) The musicians are suggested to use C, C++ or C++ based software like [4], [5], [15], [16], [18], [20], [23] and [24] for their real-time needs such as live performance because when the limitations are considered, applications implemented with higher level languages are found to lack of real-time performance.

4) The biological inspired computational methods or algorithms which are commonly used for developing GMCSS can be implemented modularly, easily and rapidly by an object oriented language. In this sense, the developer is suggested to use C++ programming language, since it is the most appropriate object oriented language for developing cross platform and real-time applications. It is why most of reviewed software is based on C++.

5) The developer is suggested to use Evolutionary or Interactive Evolutionary Algorithms for the software which is capable of making complete composition as it can be seen from the distribution tables.

There is a communication gap between the researchers and software developers within the field. For example, researchers are aware of the fitness bottleneck problem and they mention it in their research papers; however, software developers are not aware of the fitness bottleneck problem. For this reason, not much software system is implemented, which points the problem of fitness bottleneck.

Our results show a clear way for software developers. Considering the results, software developers can select the material and methods for their future systems in a more conscious way.

For further research it is suggested to conduct a survey on users in order to understand their preferences among 30 GMCSS by studying the results of mentioned survey statistically.

Oliver Bown, the author of Experiments in Modular Design for the Creative Composition of Live Algorithms, which is one of the included studies, shares his opinion

about this research:

> *It is not easy to reproduce the system from the paper, which is a problem for this kind of research. There are many tweaks and idiosyncratic design decisions that make it too hard to fit the details into a paper and keep it interesting. However, this should be an ambition of all such research and I hope to do this in my future papers.*

As Oliver Bown mentions, there are many challenges and many possibilities of design decisions which make this thesis hard to perform to fit the details into this study. I strongly believe that this research study was successful in terms of making appropriate design decisions and fitting the details into this study.

Presenting personal comments especially in benefits and limitations sections is not allowed in SLR. Therefore, a better reviewing method could be found instead of an SLR.

4.1 Discussion about Biologically Inspired Computational Methods

When the included studies were reviewed for biologically inspired computational methods used for music generation, 30 out of 30 software systems were extracted for this specific information. Thirteen systems use evolutionary algorithms, 9 systems use interactive evolutionary algorithms, 1 system uses cellular automata, 2 systems use neural networks, 1 system uses ant colony optimization algorithm, 2 systems use combination of neural networks and evolutionary algorithms, and 2 systems use combination of interactive evolutionary algorithms and neural networks. According to reviewed studies for biologically inspired computational methods used for music generation, **evolutionary algorithms**, which use mechanisms inspired by biological evolution, such as reproduction, mutation, recombination, and selection [Wik13c], are the **most used** biologically inspired algorithms in 30 software systems. Because evolutionary algorithms are the most used biologically inspired method within 30 software systems, it is observed that evolutionary algorithms are the **most popular** biologically inspired algorithms for implementing generative music composition software system. **Interactive evolutionary algorithms**, which allow the user to select which individuals to breed for the next generation, are the second most used biologically inspired algorithm within 30 included studies. Subjective evaluation of the user leads the sound generation or generated piece of music to be in the way that the user likes. Based on this aspect, when interactive evolutionary algorithms are used, the system becomes under the control of the user.

4.2 Discussion about Music Types

When the included studies were reviewed for the types of generative music composition software systems, 30 out of 30 software systems were extracted for this specific

information. Fourteen systems generate complete composition, 8 systems generate music melody, 2 systems generate rhythm, 2 systems generate music harmony, 1 system generates electronic music, 1 system generates soundscape, and 2 systems are sound synthesizers. According to reviewed studies for the types of generative music composition software systems, **complete composition**, which refers to musical melody that has more than one musical instrument, is the **most used** music type within 30 included studies. Because complete composition is the most used music type within 30 software systems, it is observed that complete composition is the **most preferred way** to generate music. **Melody generation** is the second most used music type within 30 included studies.

4.3 Discussion about Programming Languages

When the included studies were reviewed for programming languages of generative music composition software systems, 19 out of 30 software systems were extracted for this specific information. Six systems were implemented by using C++, 5 systems were implemented by using Java, 2 systems were implemented by using C, 1 system was implemented by using C#, 2 systems were implemented by using Objective C, 1 system was implemented by using Python, 1 system was implemented by using THINK C version 5, and 1 system was implemented by using Common Lisp. According to reviewed studies for programming languages of generative music composition software systems, **C++** is the **most used** programming language within 30 software systems. For this reason, that **C++** can be considered as one of the **most preferred** programming language for implementing generative music composition software systems. **Java** is the second most used programming language for implementing generative music composition software systems. It should be noted that 16 of software systems are implemented in **C based languages**, which are C, C++, Objective C, Java, C#. Moreover, 16 of the systems are implemented with **object oriented programming languages**, which are C#, C++, Java, Objective C, Python and THINK C.

4.4 Discussion about Programming Environment

When the included studies were reviewed for programming environment of generative music composition software systems, 17 out of 30 software systems were extracted for this specific information. Four systems are implemented by using Pure Data, which is a user friendly IDE, and Max/MSP, 4 systems are implemented by using Eclipse, 2 systems are implemented by using NetBeans, 1 system is implemented by using CodeWarrior IDE, 1 system is implemented by using Microsoft Visual Studio 6.0 on Windows, 1 system is implemented by using Moxc, 1 system is implemented by using MATLAB, 1 system is implemented by using Arduino Environment, 1 system is implemented by using .NET with DirectX, and 1 system is implemented by using Macintosh Common Lisp. According to the reviewed studies for programming environment of generative music composition software systems, **Pure Data** and **Max/MSP**, which are real-time

graphical programming environments for audio, video, and graphical processing, and **Eclipse** are the **most used** programming environments within 30 software systems. Therefore, **Pure Data** and **Max/MSP**, and **Eclipse** can be considered as one of the **most popular** programming environments for implementing generative music composition software systems. Moreover, it is worth mentioning that recent studies have been conducted with Pure Data and Max/MSP in general while earlier ones have been performed with Eclipse.

4.5 Discussion about Main Functionalities

The main functionality of the included studies, which were reviewed, is to produce sound or music by using various techniques. The variety of the techniques comes from the tested methods for increasing the quality of the music or sound that is generated. Moreover, an interface which is user friendly can facilitate programmers' work.

4.6 Discussion about Benefits

The benefits are the facilities which are provided to the end users for the techniques used and the quality of the resulting product of the generated music. Moreover, the benefits of the systems are that they help users to create high quality music more easily. For instance, in the systems that are using interactive evolutionary algorithms instead of evolutionary algorithms, the process being under control of the user is one of the most important and invaluable benefit.

4.7 Discussion about Limitations

One of the biggest, most noticeable and striking limitation in the area of evolutionary computation is the fitness bottleneck, that is, manual evaluation of the user takes too much time for many individuals. Users need to hear each individual and set a score for those individuals such as good or bad [Bil07]. To solve this problem, a lot of research has been done. One of them is the **temporary storage** which was created by Dahlstedt. Temporary storage is mentioned in his two studies which are [Dah07] and [Dah09a]. He created a software tool named Patch Mutator. In the tool there is a section named temporary storage which has number of slots, which serves as a temporary storage for promising sounds and potential sounds to be kept. Temporary storage solves the problem of fitness bottleneck. Another common solution is the automated choice in parallel or in series with human choice. There is a person in the loop, but the person is not in the loop all the time. Automated fitness functions model the behavior of the user.

According to John A. Biles [Bil94], in the evolutionary computation area, the problem of fitness bottleneck often manifests itself in the fitness function used to guide the evolutionary process of a composing or improvising system. Objective fitness functions always seem to lead to low-quality music. The most musically successful systems tend

to be **interactive** systems, where a human mentor listens to the music and provides feedback on its value. The bottom line is that music is an artistic domain, not an engineering discipline.

5 Conclusions

In this thesis, the results of the systematic literature review of generative music composition software systems in terms of methods and approaches used and main functionalities, benefits, and limitations are presented.

2040 studies were identified from searches of the literature and 30 studies were found to have acceptable relevance. These 30 studies were included in this thesis. The way of selecting 30 studies out of 2040 studies was to apply 2 exclusion criteria to 2040 studies. The first exclusion criterion is eliminating the studies which do not refer to computer science. The second exclusion criterion is eliminating studies which do not refer to a method.

5.1 Concluding Remarks

It is observed that evolutionary algorithms are the most used biologically inspired computational method; however, interactive evolutionary algorithms can be better depending on the goal of the user. If the user wants to reach a specific sound or music in his mind and wants to control the flow of breeding then IEAs are better. If the user does not know where to go in sound or music exploration then EAs are incredibly great choice for the user. EAs being more than IEAs in the result are not by chance because the systems that are implemented with EAs are easier for the users to get results faster and play with. The software which is easy to use attracts users more.

It is observed that complete composition is the most used music type however melody generation can be better depending on the goal of the user. For example if a piano player get stuck in producing a melody and his focus is only producing a piano melodies but not the bass guitar, drum partitions then melody generation can be a better option.

C++ is the most used programming language in these studies. This indicates that the software developers of the programs for the included studies are using the right programming language since C++ is the most appropriate object oriented language for developing cross platform and real-time applications. It is why most of reviewed software is based on C++. Moreover, the finding considering the result of this thesis is that most of the reviewed software systems are implemented using object oriented programming languages and C based programming languages.

For programming environment it is observed that 4 software systems are implemented in Eclipse and 4 software systems are implemented in Pure Data and Max/MSP. Eclipse and Pure Data and Max/MSP are the most used programming environments. However, it is an important point that the recent studies have been conducted with Pure

Data and Max/MSP in general while earlier ones were performed with Eclipse. Considering this information, future software developers are strongly suggested to create the **sound engine** of their system by using Pure Data or Max/MSP when implementing generative music composition system instead of Eclipse. Moreover, the best option for the future software developers is to focus on using Pure Data when implementing the sound engine of the generative music composition software system.

As it is mentioned in the discussion section the main functionality of the included studies which are reviewed is to produce sound or music by using various techniques. Future software developers can analyze the main functionalities of the reviewed systems and take the ones that they want to include in their generative music composition software systems, collate the functionalties and specify the requirements of their software systems.

According to reviewed studies in terms of benefits, it is observed that the aim is to increase the quality of the software system and provide user friendly environment for the user. The future software developers, after creating the requirements of their main functionailities, they must understand the pointing benefit of the main functionality so that they can apply this specific benefit to their computer programs. For example, if the main functionality of a software is users being able to interactively evolve music then the benefit can be creating a automated fitness raters so that the automated fitness raters learn to rate in a similar way to the user.

The most striking limitation in 30 included studies is fitness bottleneck, that is, manual evaluation of the user takes too much time for many individuals. Palle Dahlstedt is one of the researchers who overcome this problem by implementing temporary storage. As a solution of the fitness bottleneck the other researchers avoid the data loss of the new generated melody by saving it. The focus for both future software developers and researchers must be solving the problem of fitness bottleneck.

5.2 Limitations and Constraints

All of the information that is given in main functionalities, benefits and limitations and biologically inspired computational methods are extracted from the included studies by systematically reviewing them. In some of the included studies music types, programming languages, programming environments of the systems are **not** mentioned. Therefore, to get this information the authors of the included studies are interviewed to gather this information. The authors are contacted via email. Although contacting them via email, some of them could not be reached. In these cases N/A is written. It is written when the information could not be reached via reviewing the included study or the information could not be reached by interviewing the author. N/A is used in the Table 5, 6, 7, and 8 and in Appendix F which is Summary of the Results. Moreover in Appendix B Background Information About the Authors of the Included Studies *Authors could not be contacted* is written when there is no reply from the authors.

5.3 Future Work

This SLR can be used to observe the current research as well as contemporary practice by presenting the most important software systems related to the field. This study both helps **practitioners** (software developers) who want to implement a generative music composition software systems and **researchers** for further improvement.

According to the 30 included studies Pure Data and Max/MSP are the most used programming environments. If the software developer wants to implement a mobile application, he should implement software for iOS, which is a mobile operating system developed by Apple, because to be able to make a generative music composition system application, libpd, which is one of the Pure data library [fPD13], should be used. Android, which is a mobile operating system developed by Google, is notorious for its audio problems and libpd has a lot to build upon that fixes this. For iOS using libpd could most of the time be the way if the software developer is performing dynamic patching. The software developer can easily make iOS apps with libpd by using openframeworks [Ope13], which is s an open source C++ toolkit for creative coding, and the ofxpd addon. Having experimented a bit with genetic evolution of sound programs, an idea could be to follow one of two ways: 1) Have a static architecture where the chromosomes correspond to various parameters in the synthesizer. Using libpd, this can be done by storing all parameters as a part of one single array and when performing crossover a series of random numbers is created and the settings from two organisms is spliced using a threshold (if the number is over a certain number use parameter value of from B, otherwise A), and 2) Have dynamic modular organism: this is harder to implement compared to the first one. The software developer would not easily be able to do it using libpd or rather it would just take a lot of work to do it this way using libpd. If the software developer is working with iOS then he can go the native route, use ofxSynth in openframeworks (it has dynamic patching through code) as a basis and create a binary tree representation of your audio graph and splice by looking at similar branches. If the software developer has access to a Macintosh computer it is easy to compile openframeworks for android as a native activity.

As it is mentioned previously both future software developers and researchers must solve the problem of fitness bottleneck.

Moreover, for further research it is suggested to perform a survey on users in order to understand their preferences among 30 GMCSS by studying the results of the mentioned survey.

References

[AM11] Torsten Anders and Eduardo R. Miranda. A survey of constraint programming systems for modelling music theories and composition. *ACM Computing Surveys*, 2011.

[app95] Composing with genetic algorithms. In *International Computer Music Association*, pages 452–455, 1995.

[art13] Generative art international conference. `http://www.generativeart.com`, 2013.

[atUoCF11] Evolutionary Complexity Research Group at the University of Central Florida. Functional scaffolding for musical composition. `http://eplex.cs.ucf.edu/fsmc/`, dec 2011.

[atUoCF12] Evolutionary Complexity Research Group at the University of Central Florida. Maestrogenesis. `http://maestrogenesis.org`, December 2012.

[BD08] Kjell Bäckman and Palle Dahlstedt. A generative representation for the evolution of jazz solos. In *Proceedings of the 2008 Conference on Applications of Evolutionary Computing*, number 10 in Evo'08, pages 371–380, Berlin, Heidelberg, 2008. Springer-Verlag.

[BE05] Dave Burraston and Ernest Edmonds. Cellular automata in generative electronic music and sonic art : Historical and technical review. *Digital Creativity*, 16(3):165–185, jan 2005.

[BGGP04] Wolfgang Banzhaf, David Goldberg, Erik Goodman, and Riccardo Poli. Human-competitive awards in genetic and evolutionary computation 2004. `http://www.genetic-programming.org/gecco2004hc.html`, September 2004.

[Bil94] John Biles. Genjam: A genetic algorithm for generating jazz solos. pages 131–137, 1994.

[Bil07] John A. Biles. Improvizing with genetic algorithms: Genjam. In *Evolutionary Computer Music*, pages pp 137–169. Springer London, 2007.

[BKB+07] Pearl Brereton, Barbara A. Kitchenham, David Budgen, Mark Turner, and Mohamed Khalil. Lessons from applying the systematic literature review process within the software engineering domain. *Journal of Systems and Software*, 80(4):571–583, 2007.

[Bow11] Oliver Bown. Experiments in modular design for the creative composition of live algorithms. In 35, editor, *Comput. Music J.*, volume 3, pages 73–85, september 2011.

[Bra98] Riccardo Poli Brad Johanson. Gp-music: An interactive genetic programming system for music generation with automated fitness

raters. In *Proceedings of the Third Annual Conference*, pages 181–186. MIT Press, 1998.

[Bro12] Daniel Brown. Mezzo: An adaptive, real-time composition program for game soundtracks. Technical report, Artificial Intelligence and Interactive Digital Entertainment Conference, 2012.

[Bur67] Alex Burnard. *Harmony and composition for the student and the potential composer / by Alex Burnard.* Allan's Music (Australia) Melbourne, 1967.

[BV99] Anthony R. Burton and Tanya R. Vladimirova. Generation of musical sequences with genetic techniques. *Comput. Music J.*, 23(4):59–73, December 1999.

[Col09] Nick Collins. Musical form and algorithmic composition. *Contemporary Music Review*, 28(1):103–114, 2009.

[CPKK93] Erick Cantu-Paz, Chandrika Kamath, and Rika Kamath. On the use of evolutionary algorithms in data mining. In *Proc. 2nd International Conf. on Simulation of Adaptive Behavior*, pages 384–392. MIT Press, 1993.

[Dah07] Palle Dahlstedt. Evolution in creative sound design. In EduardoReck Miranda and JohnAl Biles, editors, *Evolutionary Computer Music*, pages 79–99. Springer London, 2007.

[Dah09a] Palle Dahlstedt. On the role of temporary storage in interactive evolution. In *Proceedings of the EvoWorkshops 2009 on Applications of Evolutionary Computing: EvoCOMNET, EvoENVIRONMENT, EvoFIN, EvoGAMES, EvoHOT, EvoIASP, EvoINTERACTION, EvoMUSART, EvoNUM, EvoSTOC, EvoTRANSLOG*, 2009.

[Dah09b] Palle Dahlstedt. Thoughts on creative evolution: A meta-generative approach to composition. *Contemporary Music Review*, 28:43–55, 2009.

[Dah12] Palle Dahlstedt. Autonomous Evolution of Complete Piano Pieces and Performances. *A-Life for music: music and computer models of living systems*, 2012.

[Dan96] Roger B. Dannenberg. The cmu midi toolkit. `http://www.cs.cmu.edu/~rbd/papers/cmticmc.ps.Z`, 1996.

[Doo04] Paul Doornbusch. Computer sound synthesis in 1951: The music of csirac. *Comput. Music J.*, 28(1):10–25, mar 2004.

[evo13] Evostar - the leading european event on bio-inspired computation. `http://www.evostar.org`, April 2013.

[Fen] Biology online. `http://www.biology-online.org/dictionary/Phenotype`.

[For12] José Fornari. A computational environment for the evolutionary sound synthesis of birdsongs. In *Proceedings of the First international conference on Evolutionary and Biologically Inspired Music, Sound, Art and Design*, EvoMUSART'12, pages 96–107, Berlin, Heidelberg, 2012. Springer-Verlag.

[fPD13] Community Portal for Pure Data. Libpd. `http://puredata.info`, April 2013.

[FV13] Jose D. Fernandez and Francisco J. Vico. Ai methods in algorithmic composition: A comprehensive survey. *J. Artif. Intell. Res. (JAIR)*, 48:513–582, 2013.

[GB91] P.M. Gibson and J.A. Byrne. Neurogen, musical composition using genetic algorithms and cooperating neural networks. In *Artificial Neural Networks, 1991., Second International Conference*, pages 309–313, november 1991.

[Ge1] Biology online. `http://www.biology-online.org/dictionary/Genotype`.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. 0716710447. W. H. Freeman & Co., New York, NY, USA, 1979.

[GJ03] Andrew Gartland-Jones. Musicblox: a real-time algorithmic composition system incorporating a distributed interactive genetic algorithm. In *Proceedings of the 2003 international conference on Applications of evolutionary computing*, EvoWorkshops'03, pages 490–501, Berlin, Heidelberg, 2003. Springer-Verlag.

[Hil81] Lejaren Hiller. Composing with computers: A progress report. *Computer Music Journal*, 5(4):7+, FebApr 1981.

[Hor94] Damon Horowitz. Generating rhythms with genetic algorithms. In *Proceedings of the twelfth national conference on Artificial intelligence (vol. 2)*, AAAI'94, Menlo Park, CA, USA, 1994. American Association for Artificial Intelligence.

[HS07] Amy Kathryn Hoover and Kenneth O. Stanley. Neat drummer: Interactive evolutionary computation for drum pattern generation. Technical report, The AMALTHEA REU Program, 2007.

[In 95] In Proceedings of the Brazilian Symposium on Computer Music. *Fifteen Years of Computer-Assisted Composition*, 1995.

[Inc13] PG Music Inc. Band in a box. `http://www.pgmusic.com/`, September 2013.

[IO12] E Intermorphic. and OE. Generative music 1 and brian eno. `http://www.intermorphic.com/tools/noatikl/generative_music.html`, 2012.

[Jac95] Bruce Jacob. Composing with genetic algorithms. In *International Computer Music Association*, pages 452–455, 1995.

[Jac96] Bruce L. Jacob. Algorithmic composition as a model of creativity. *Organised Sound*, 1:157–165, 1996.

[Jin05] Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Comput.*, 9(1):3–12, Jan 2005.

[KC07] Barbara Kitchenham and Stuart Charters. Guidelines for performing systematic literature reviews in software engineering. Technical report, Keele University and Durham University Joint Report, 2007.

[KPB+09] Barbara Kitchenham, O. Pearl Brereton, David Budgen, Mark Turner, John Bailey, and Stephen Linkman. Systematic literature reviews in software engineering - a systematic literature review. *Inf. Softw. Technol.*, 51(1):7–15, January 2009.

[Mac13] Wiktor K. Macura. Ant colony algorithm. `http://mathworld.wolfram.com/AntColonyAlgorithm.html`, November 2013.

[Mal77] William P. Malm. *Music cultures of the Pacific, the Near East, and Asia*. Prentice-Hall History of Music Series. Prentice-Hall, Englewood Cliffs, N.J., 1977.

[MIL05] M. MILLER. *The Complete Idiot's Guide to Music Theory*. The Complete Idiot's Guide Series. Alpha Books, 2005.

[Mir02] Eduardo Reck Miranda. Creative evolutionary systems. In *On the origins and evolution of music in virtual worlds*, chapter On the origins and evolution of music in virtual worlds, pages 189–204. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002.

[MM11] Jon McCormack and Peter McIlwain. Generative composition with nodal. In Eduardo R. Miranda, editor, *A-Life for Music: Music and Computer Models of Living Systems*, Computer Music and Digital Audio, pages 99–113. A-R Editions, Inc., 2011.

[MMVG02] Artemis Moroni, Jonatas Manzolli, Fernando Von Zuben, and Ricardo Gudwin. Vox populi: Evolutionary computation for music evolution. In *Creative evolutionary systems*. Morgan Kaufmann Publishers Inc., 2002.

[Nak11] T. Nakano. Biologically inspired network systems: A review and future prospects. *Trans. Sys. Man Cyber Part C*, 41(5):630–643, Sep 2011.

[Nat90] Jean J. Nattiez. *Music and discourse : toward a semiology of music*. Princeton University Press, 1990.

[Nel93] G. L. Nelson. Sonomorphs: An application of genetic algorithms to the growth and development of musical organisms. In *Proceedings of the Fourth Biennial Art Technology Symposium*, volume 155, 1993.

[OE08] Ender Ozcan and Türker Ercal. A genetic algorithm for generating improvised music. In Nicolas Monmarche, El-Ghazali Talbi, Pierre Collet, Marc Schoenauer, and Evelyne Lutton, editors, *Artificial Evolution*, volume 4926 of *Lecture Notes in Computer Science*, pages 266–277. Springer Berlin Heidelberg, 2008.

[Ope13] OpenFrameworks. Openframeworks. `http://www.openframeworks.cc/`, November 2013.

[Pas13] Philippe Pasquier. 1st international workship on musical metacreation (mume 2012). `http://www.metacreation.net/mume2012/`, June 2013.

[PPPPTPSHTM13] Inc. Preventing Piracy Privacy Policy Trademarks Patents Site Help The MathWorks. Matlab the language of technical computing. `http://www.mathworks.com/products/matlab/`, November 2013.

[Pro89] Proceedings of the International Computer Music Conference. *Six Techniques for Algorithmic Music Composition.*, San Francisco: Computer Music Conference Association, 1989. Computer Music Conference Association.

[PW99] George Papadopoulos and Geraint Wiggins. Ai methods for algorithmic composition: A survey, a critical view and future prospects. In *IN AISB SYMPOSIUM ON MUSICAL CREATIVITY*, pages 110–117, 1999.

[Roa85] Curtis Roads. Research in music and artificial intelligence. *ACM Comput. Surv.*, 17(2):163–190, jun 1985.

[Ros13] Henrique Roscoe. Dot, a videogame with no winner. Technical report, Hol, 2013.

[SA94] Lee Spector and Adam Alpern. Criticism, culture, and the automatic generation of artworks. In *Proceedings of the twelfth national conference on Artificial intelligence (vol. 1)*, AAAI '94, pages 3–8, Menlo Park, CA, USA, 1994. AAAI '94.

[SAD+00] Antonino Santos, Bernardino Arcay, Julián Dorado, Juan Romero, and Jose Rodriguez. Evolutionary computation systems for musical composition. In *In Mathematics and Computers in Modern Science*, pages 97–102. Society Press. 2000, 2000.

[SG99] Naresh K. Sinha and Madan M. Gupta. *Soft Computing and Intelligent Systems: Theory and Applications*. Number 0126464901. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 1999.

[SGR09] Florian Schulz, Chris Geiger, and Holger Reckter. Antracks: generative mobile music composition. In *Proceedings of the International Conference on Advances in Computer Enterntainment Technology*, pages 302–305, 2009.

[SMOB10] Jianhua Shao, James McDermott, Michael O'Neill, and Anthony Brabazon. Jive: A generative, interactive, virtual, evolutionary music system. In Cecilia Chio, Anthony Brabazon, GianniA. Caro, Marc Ebner, Muddassar Farooq, Andreas Fink, Jörn Grahl, Gary Greenfield, Penousal Machado, Michael O'Neill, Ernesto Tarantino, and Neil Urquhart, editors, *Applications of Evolutionary Computation*, volume 6025 of *Lecture Notes in Computer Science*, pages 341–350. Springer Berlin Heidelberg, 2010.

[SP94] M. Srinivas and Lalit M. Patnaik. Genetic algorithms: A survey. *Computer*, 27(6):17–26, Jun 1994.

[SPVP07] Ángel Sánchez, JuanJosé Pantrigo, Jesús Virseda, and Gabriela Pérez. Spieldose: An interactive genetic software for assisting to music composition tasks. In José Mira and JoséR. Álvarez, editors,

Bio-inspired Modeling of Cognitive Tasks, volume 4527 of *Lecture Notes in Computer Science*, pages 617–626. Springer Berlin Heidelberg, 2007.

[Tak01] H. Takagi. Interactive evolutionary computation: fusion of the capabilities of ec optimization and human evaluation. *Proceedings of the IEEE*, 89(9):1275–1296, 2001.

[Thy99] Kurt Thywissen. Genotator: an environment for exploring the application of evolutionary techniques in computer-assisted composition. *Org. Sound*, 4(2):127–133, jun 1999.

[TW09] W. J. Tang and Q. H. Wu. Biologically inspired optimization: a review. *Transactions of the Institute of Measurement and Control*, 31(6):495–515, Dec 2009.

[UO03] Muneyuri Unehara and Takehisa Onisawa. Music composition system with human evaluation as human centered system. *Soft Computing*, 7:167–178, 2003.

[US01] Tatsuo Unemi and Manabu Senda. A new musical tool for composition and play based on simulated breeding. In *Proceedings of Second Iteration*, pages 10–9, 2001.

[Was07] Rodney Waschka. *Composing with Genetic Algorithms: GenDash*. Springer London, 2007.

[Wik12] Wikipedia. Interactive evolutionary computation. `http://en.wikipedia.org/wiki/Interactive_evolutionary_computation`, November 2012.

[Wik13a] Wikipedia. Algorithmic composition. `http://en.wikipedia.org/wiki/Algorithmic_composition`, august 2013.

[Wik13b] Wikipedia. Embodied music cognition. `http://en.wikipedia.org/wiki/Embodied_music_cognition`, April 2013.

[Wik13c] Wikipedia. Evolutionary algorithm. `http://en.wikipedia.org/wiki/Evolutionary_algorithm`, November 2013.

[Wik13d] Wikipedia. Evolutionary algorithms. `http://en.wikipedia.org/wiki/Evolutionary_computation#Evolutionary_algorithms`, October 2013.

[Wik13e] Wikipedia. Melody. `http://en.wikipedia.org/wiki/Melody`, April 2013.

[Wik13f] Wikipedia. Microtonal music. http://en.wikipedia.org/wiki/Microtonal_music, April 2013.

[Wik13g] Wikipedia. Synthesizer patch. http://en.wikipedia.org/wiki/Synthesizer#Patch, august 2013.

[Wik13h] Wikipedia. Theme (music). http://en.wikipedia.org/wiki/Theme_(music), February 2013.

[Wik13i] Wikipedia. Think c. http://en.wikipedia.org/wiki/THINK_C, March 2013.

[WT02] Sam Woolf and Andrian Thomas. Creative evolutionary systems. In *The sound gallery an interactive A-life artwork*, chapter The sound gallery an interactive A-life artwork, pages 223–250. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002.

[ZBT11] He Zhang, Muhammad Ali Babar, and Paolo Tell. Identifying relevant studies in software engineering. *Inf. Softw. Technol.*, 2011.

A Appendix: Definitions and Abbreviations

Abbreviation	Description
AI	Artificial Intelligence
CA	Cellular Automata
CS	Computer Science
EAs	Evolutionary Algorithms
EBSE	Evidence Based Software Engineering
EC	Evolutionary Computation
EMA	Evolutionary Music and Art
GAs	Genetic Algorithms
GP	Genetic Programming
GUI	Graphical User Interface
GMCSS	Generative Music Composition Software Systems
IEAs	Interactive Evolutionary Algorithms
IEC	Interactive Evolutionary Computation
IGAs	Interactive Genetic Algorithms
IGP	Interactive Genetic Programming
NIME	New Interfaces for Musical Expression
NMG2	Nord Modular G2
NNs	Neural Networks
N/A	The information which could not be reached via reviewing and interviewing
OF	Open Frameworks
OSC	Open Sound Control
Pd	Pure Data
SE	Software Engineering
SLR	Systematic Literature Review

B Appendix: Background Information About the Authors of the Included Studies

Ref	Software Name	Author(s) Name	Background of Authors	Where is the Software used?
1	Nodal	J. McCormack	Jon is an electronic media artist and researcher in computing. His detailed biography is provided below: http://jonmccormack.info/~jonmc/sa/about/biography-jon-mccormack/	Nodal is a published software which is used by millions of users
		P. McIlwain	Peter is a composer who specializes in orchestral music, electronic music and computer generated music. He has collaborative projects in sound installations for museums, theatre productions, live music performance. He is an audio technician. He has broad experience in the use of multimedia applications including web design. He is a teacher specializing in composition and electronic music at a University level. Moreover he is a graphic designer and a researcher. His web page is provided below: http://sonicdesign.com.au/about.html	
		A. Lane	Aidan is software engineer and research assistant at Faculty of Information Technology, Monash University.	
		A. Dorin	Dr Alan Dorin is an Associate Professor in the Faculty of Information Technology, Monash University. His web page is provided below: http://www.infotech.monash.edu.au/research/profiles/profile.html?sid=806&pid=2797	

Ref	Software Name	Author(s) Name	Background of Authors	Where is the Software used?
2	GP-Music	B. Johanson	Brad finished working as a post-doc in Computer Science at Stanford University. In his Masters degree at the University of Birmingham, England, he worked in the field of Genetic Programming, and has a paper on using GP and neural networks to automatically compose music.	Authors could not be contacted
		R. Poli	He is a Professor in the Department of Computing and Electronic Systems of the University of Essex. In PhD he worked in biomedical image analysis. He later became an expert in the field of Evolutionary Computation. His home page is provided below: http://cswww.essex.ac.uk/staff/poli/	
3	SBEAT	T. Unemi	Tatsuo Unemi is a professor in Department of Information Systems Science, Soka University. His research included Natural Language Processing, Knowledge Engineering, Machine Learning, Genetic Algorithm, Reinforcement Learning, Distributed Autonomous Robot System, and Artificial Life. Current interests include artistic, sociological, and humanities applications of these technologies. His home page is provided below: http://www.intlab.soka.ac.jp/~unemi/	SBEAT is a published free software which is used by millions of users http://www.intlab.soka.ac.jp/~unemi/sbeat/
		M. Senda	Manabu is a professor at Department of Information Systems Science, Soka University	

Ref	Software Name	Author(s) Name	Background of Authors	Where is the Software used?
4	ANTracks	F. Schulz	Florian is a software developer who is implementing applications in Berlin, Germany.	Antracks ist not used anymore but the hexagonal structure of notes is still used in another interface for musical expressions, a 2D theremin-based approach in a virtual studio environment. Citation of the article is provided below: Dionysios Marinos, Björn Wöldecke, Chris Geiger, and Tobias Schwirten. 2011. Design of a touchless multipoint musical interface in a virtual studio environment. In Proceedings of the 8th International Conference on Advances in Computer Entertainment Technology (ACE '11), Teresa Romão, Nuno Correia, Masahiko Inami, Hirokasu Kato, Rui Prada, Tsutomu Terada, Eduardo Dias, and Teresa Chambel (Eds.). ACM, New York, NY, USA, , Article 33 , 7 pages. DOI=10.1145/2071423.2071464 http://doi.acm.org/10.1145/2071423.2071464
		C. Geiger	Chris Geiger is professor for Mixed Reality and Visualization at University of Applied Sciences Düsseldorf.	
		H. Reckter	Holger Reckter is professor at University of Applied Sciences Mainz, Germany	
5	GenJam	J. A. Biles	Jazz musician who is a trumpet player and computer scientist	GenJam and the author have performed all over the world. He performs with the software a couple of times a month, on the average.

Ref	Software Name	Author(s) Name	Background of Authors	Where is the Software used?
6	Patch Mutator	P. Dahlstedt	Associate Professor of Applied Information Technology at Gothenburg University Palle Dahlstedt is a composer, scholar, pianist, improviser, instrument builder and sound artist, as well as a professional programmer. After studying composition in Stockholm, Dahlstedt's often very intense music ranges from music for films, theatre, dance and installations to instrumental, orchestral and prize-winning electroacoustic pieces, performed all over the world. He is also a lecturing co-founder of the international master's program in Art & Technology at Chalmers University of Technology, where he is also doing research in creative algorithms. Palle specializes in computer-aided creativity, and his research is partly about developing new technologies for musical improvisation and composition, and on understanding the inner mechanisms of artistic creation processes, in order to get computers to behave more creatively. He is also a teacher in the composition of electronic music and electronic music composition at the Academy of Music and Drama, as well as artistic director of the Lindblad studio. His web page is provided below http://www.ait.gu.se/kontaktaoss/personal/palle_dahlstedt/	Patch Mutator is a tool which is embedded in Nord Modular G2. Patch Mutator is a published system and it is used by millions of people to help musicians with their creative process.

Ref	Software Name	Author(s) Name	Background of Authors	Where is the Software used?
7	NEAT Drummer	A. K. Hoover	Amy is a Ph.D. candidate in computer science at the University of Central Florida. Her research focuses on artificial intelligence in a musical domain encompassing topics in computational creativity, computer-generated music, music cognition, NeuroEvolution of Augmenting Topologies (NEAT), HyperNEAT, and Compositional Pattern Producing Networks (CPPNs). Amy plays all kinds of instruments and taking private music lessons her whole life, however she does not have formal education in music theory. She currently spend most of her time playing bass guitar, she also plays and has competed on all sorts of woodwinds, including a world competition for bagpipes in Glasgow, Scotland. Her web page is provided below: http://amykhoover.com/	NEAT Drummer was never released; It is used in Evolutionary Complexity Research Group at the University of Central Florida in laboratory environment. The drum tracks which is generated by NEAT drummer can be found from the URL below: http://eplex.cs.ucf.edu/neatmusic/
		K. O. Stanley	Ken Stanley is an Associate Professor, in Department of Electrical Engineering and Computer Science, The University of Central Florida. Moreover he took a music class in college and played throughout middle and high school. His web page is provided below http://www.cs.ucf.edu/~kstanley/	
8	Ossia II	P. Dahlstedt	The information about the author is already provided in page 80.	Ossia has been used in Universium which is a museum in Gothenburg for several years and music festivals.
9	Application of Genetic Algorithms	B. Jacob	Bruce Jacob is a professor in Department of Electrical and Computer Engineering University of Maryland at College Park. His web page is provided below http://www.ece.umd.edu/~blj/	The author just made the software for himself. He presented its music at ICMC and posted it on his website.

Ref	Software Name	Author(s) Name	Background of Authors	Where is the Software used?
10	Spieldose	A. Sanchez	He is an associate professor in Universidad Rey Juan Carlos (URJC). He works in GAVAB Group (GAVAB), department of computer science, area of computation and artificial intelligence, Higher Technical School of Computer Engineering (ETSII). His web page is provided below: http://www.escet.urjc.es/~ansanche/	Spieldose is only used in the laboratory of University Rey Juan Carlos, Department of Computer Science
		J. Pantrigo	He is associate professor at the URJC (Spain) where he is a member of the GAVAB and CAPO research groups in the Departamento de Ciencias de la Computación - DCC (Department of Computer Science). His main research interests focus on the interface among Computer Science, Artificial Intelligence, Computer Vision and Operations Research. Specifically, he is mainly interested in combinatorial optimization, high performance computing, knowledge modeling his web page is provided below: http://www.escet.urjc.es/~jjpantrigo/	
		J. Virseda	Virseda was a student and he does not work in research nowadays.	
		G. Perez	Gabriela obtained her PhD in 2011 and she does not work in research nowadays.	

Ref	Software Name	Author(s) Name	Background of Authors	Where is the Software used?
11	AMUSE (A MUSical Evolutionary assistant)	E. Özcan	He is a "Science and Innovation" Lecturer in the School of Computer Science at the University of Nottingham, working in the Automated Scheduling, Optimisation and Planning Research Group (ASAP). his web site is provided below: http://www.cs.nott.ac.uk/~exo/	Ercal is using AMUSE in his compositions.
		T. Ercal	He is working in Hitit Computer Services as a project leader. He is a half professional guitarist who plays in a band. He created AMUSE for himself and the ones who are interested in the area of biologically inspired computational methods. His Linkedin profile is provided below: http://www.linkedin.com/in/turkerercal	
12	Variations	B. L. Jacob	His background information is already given in page 81	Author could not be contacted

Ref	Software Name	Author(s) Name	Background of Authors	Where is the Software used?
13	Jive	J. Shao	He is a PhD candidate in Digital Economy in University of Nottingham. His Linkedn profile is provided below: http://www.linkedin.com/in/jianhuashao	Jive was mostly used for composition. Obviously it was also used in the laboratory for testing for publications.
		J. McDermott	Research interests of Dr James McDermott BSc PhD are in machine learning. He uses evolutionary computation to create music, graphics, and 3D designs. He studies representational issues in evolutionary computation in order to make it work better. His web page is provided below: http://www.ucd.ie/cba/members/jamesmcdermott/	
		M. O´Neill	Dr. O'Neill is Director of the UCD Complex & Adaptive Systems Laboratory (CASL), a founding Director of the UCD Natural Computing Research & Applications group, and is a Senior Lecturer in the UCD School of Computer Science & Informatics. He is the inventors of Grammatical Evolution. He is independently ranked as one of the top 5 researchers in Genetic Programming. His web page is provided below http://www.ucd.ie/casl/people/principalinvestigators/oneillmichael/	
		A. Brabazon	Anthony is currently Associate Dean, Smurfit Graduate School of Business UCD and Professor of Accountancy. His primary research interests concern the development of natural computing theory and the application of natural computing algorithms to real-world problems, including the domain of business and finance His web page is provided below http://www.ucd.ie/cba/members/anthonybrabazon/	

Ref	Software Name	Author(s) Name	Background of Authors	Where is the Software used?
14	Birdsongs	J. Fornari	Jose Fornari (Tuti) is a full-time PosDoc researcher at the Nucleus of Sound Communication (NICS) at the University of Campinas (UNICAMP). He has two PosDoc degrees: one in Music Cognition and the other in Evolutionary Computation; Ph.D. and M.S. in EE; B.S. in Music (piano) and EE. More info: https://sites.google.com/site/tutifornari/	Birdsongs were used in the presentation I gave at TEDTalentSearch of TEDxSummit in Doha/Qatar 2012. http://www.youtube.com/watch?v=o8LtGbRa-FI
15	Neurogen	P. Gibson	He is at Staffordshire Polytech., Stoke-on-Trent, UK	Authors could not be contacted
		J. Byrne	John Byrne is a Professor of Photocatalysis in the School of Engineering His profile URL is provided below http://www.nibec.ulster.ac.uk/staff/j.byrne	
16	GeNotator	K. Thywissen	Kurt Thywissen works at Tesla in California where he spends his time leading a team responsible for the in-car Infotainment and Instrument Cluster UI.	GeNotator is not commercially available or even available in any public form. It was distributed to a small group as the author worked on it. It was presented at ICMC97 in Greece, and at the Brazilian Symposium on Computer Music in 1999. The author stopped working on it in this timeframe, although he has thought many times about bringing it up to date and releasing it again. It informed some of the work the author did with Brian Eno on the Generative Music the author release he did in 1996 using the Koan generative environment.
17	Mezzo	D. Brown	Daniel Brown is a composer, cellist, and computer music researcher living in Santa Cruz, California. He researches and develops real-time, computer-composed music. His web site is provided below http://www.danielbrownmusic.com/	Mezzo is for Bown´s Doctoral Dissertation in at the University of California at Santa Cruz. It will be embedded to the computer games.

Ref	Software Name	Author(s) Name	Background of Authors	Where is the Software used?
18	Sonomorphs	G. L. Nelson	Nelson is a pioneer in the field of computer music. Nelson earned his composition doctorate at Washington University in Saint Louis. He has taught at Purdue University and Bowling Green State University. Since 1974, He has been a faculty member at the Oberlin Conservatory of Music. At Oberlin, Nelson is a Professor of Electronic and Computer Music. He is also chair of the TIMARA Department The link to his web page is provided below: http://www.timara.oberlin.edu/gnelson/gnelson.htm	Boston Museum of Science Boston Museum of Science created a special keyboard for exploring the Sonomorphs that they renamed "Musical Animals." The author got some feedback from the 10-year-old son of a composer friend who lives in Boston. In the words of this boy and his friends, Sonomorphs is awesome. At present, Sonomorphs is no longer works at Bostom Museum of Science because they upgraded all of their Macintosh computers. Nelson discussed an update to the software but they have yet to come through with funding.
19	Vox Populi	A. Moroni	Artemis Sanchez Moroni is a Researcher at Robotics and Computer Vision Division at Center for Information Technology Renato Archer	Authors could not be contacted
		J. Manzolli	Jônatas Manzolli is a mathematician and composer, tenured professor of the Music Department and head of the Interdisciplinary Nucleus for Sound Studies (NICS) of the University of Campinas (UNICAMP) Brazil. His web page is provided below http://www.nics.unicamp.br/jonatas/information.htm	
		F. Von Zuben	Fernando is a Associate Professor at the Department of Computer Engineering and Industrial Automation (DCA) of the School of Electrical and Computer Engineering (FEEC), State University of Campinas (Unicamp), Campinas, São Paulo, Brazil. His web page is provided below http://www.dca.fee.unicamp.br/~vonzuben/	

Ref	Software Name	Author(s) Name	Background of Authors	Where is the Software used?
		R. Gudwin	Ricardo is an associate professor in University of Campinas - Unicamp Campinas, Brazil. His research fields are Computer and Information Science, Artificial Intelligence Cognitive Systems, Cognitive Architectures, Machine Consciousness, Cognitive Science His web page is provided below http://faculty.dca.fee.unicamp.br/gudwin/	
20	Rhythm Generation System	D. Horowitz	Dr. Damon Horowitz is a Philosophy Professor and Serial Entrepreneur. His work explores what is possible at the boundaries of technology and the humanities. His profile can be reached from the URL below http://berlinsymposium.org/damon-horowitz	Horowitz wrote the system while a Researcher at the MIT Media Lab. It was presented in several conferences and festivals in the US and Europe, and was distributed free online to users everywhere. Though it is no longer available.
21	Music composition system with human evaluation as human centered system	M. Unehara	Muneyuki Unehara Ph.D., is an Assistant Professor in Department of Management and Information Systems Science in Nagaoka University of Technology	The software is used in the laboratory environment.
		T. Onisawa	Onisawa is proffesor in Institute of Engineering Mechanics and Systems in University of Tsukuba	
22	Beads	O. Bown	Oliver Bown is Lecturer in the Design Labs. Bown´s research areas are Digital music, music software and performance systems. Computational creativity, biologically-inspired computing, complex systems and ecosystems. Multi-agent modeling, models and theories of cultural dynamics and human evolution, particularly with respect to human artistic behavior. His web page is provided below http://sydney.edu.au/architecture/staff/homepage/oliverbown.shtml	Beads is used in the laboratory environment.

Ref	Software Name	Author(s) Name	Background of Authors	Where is the Software used?
23	ChaOS	E. R. Miranda	Miranda is a composer working at the crossroads of music and science. His music is informed and inspired by my research into Artificial Intelligence (AI) in significant ways. He has composed music for symphonic orchestras, chamber groups, solo instruments - with and without live electronics - and electroacoustic music.	The author used ChaOS a lot in the compositions of Miranda. E.g. https://soundcloud.com/ed_miranda/olivine-trees
24	Sound Gallery	S. Woolf	Sam Woolf is a freelance producer, editor and flash developer with many years of production and post-production experience in film, video, new media and exhibition development. His page is provided below http://samwoolf.net/	It is used in the laboratory of University of Sussex
		A. Thompson	Dr Adrian Thompson was working in University of Suusex. He is interested in artificial intelligence and the new breed of biologically inspired robots.	
25	DOT	H. Roscoe	Henrique Roscoe is a digital artist, musician and designer.	DOT is a live audiovisual performance that he presents in festivals. For example, Roscoe performed live with DOT at WRO Biennale, in Wroclaw, Poland and LPM Rome and Athens video art.
26	Musicblox	A. Gartland-Jones	Drew Gartland-Jones was a lecturer in Computer Music in the department of Informatics. He was was a leading light in the computational modeling of creativity, especially with reference to musical composition. His web page is provided below http://www.atgj.org/drew/tributes.htm	Brighton Festival 2002, UK, and funded by South East Arts, may be downloaded at http://www.atgj.org/drew/, following the link to 'Musical Examples', then scrolling down to 'Example of OriGen Output'.

Ref	Software Name	Author(s) Name	Background of Authors	Where is the Software used?
27	GenBebop	L. Spector	Lee Spector is a Professor of Computer Science in the School of Cognitive Science at Hampshire College and an Adjunct Professor in the Department of Computer Science at the University of Massachusetts, Amherst. He received a B.A. in Philosophy from Oberlin College in 1984, where he also studied computer music with Gary Lee Nelson, and a Ph.D. from the Department of Computer Science at the University of Maryland in 1992.	It was used only in laboratory environment. Do note that there was a second publication on an extension to the system: Spector, L., and A. Alpern. 1995. Induction and Recapitulation of Deep Musical Structure. In Working Notes of the IJCAI-95 Workshop on Artificial Intelligence and Music. pp. 41-48. http://faculty.hampshire.edu/lspector/pubs/IJCAI95mus-toappear.ps
		A. Alpern	Senior software engineer with experience at the platform, application and application suite levels. Experienced in the entire software lifecycle from requirements gathering through delivery and maintenance. Deep experience in high performance multi-threaded C++ applications, server architecture, search, and multi-channel message processing. Specialties C++, multithreading, HTTP, network protocols, REST, XML, performance, infrastructure design & implementation, indexed search, development methodology, refactoring, service-oriented architecture, tools His public profile on LinkedIn is available at: http://www.linkedin.com/in/aalpern	
28	GenJazz	P. Dahlstedt	His background information is already given in page 80.	It is used in Bäckman´s experiments with new ways of improvising, to generate new musical impulses to his own playing and as ingredients in his acoustic jazz group.
		K. Bäckman	high school teacher in computer science and mathematics for 8 years. piano teacher at the Musical Academy of Gothenburg, especially jazz improvisation for 8 years. Teacher at high school for adults in computer science and mathematics for 8 years University teacher in Informatics at West University in Trollhättan for 10 years. Own company as IT consultant, whole time for 10 years. He performed a lot of concerts.	

Ref	Software Name	Author(s) Name	Background of Authors	Where is the Software used?
29	GenDash	R. Waschka	Doctor of Musical Arts, University of North Texas • Music Composition Specialization in Computer Music/ Media • Related fields: Music History and Music Theory Dissertation: Let Me Make It Simple For You: A Lecture-Recital of Three New Works. Major Professor: Larry Austin. Committee members: Thomas Clark and Cecil Adkins. He received the Sonology Certificate from the Royal Conservatory of the Netherlands/ Institute of Sonology (The Hague), where I studied with Paul Berg, Clarence Barlow, George Lewis, and Joel Ryan. He also studied music composition with Robert Ashley and computer music composition with Charles Dodge. He is currently a professor at North Carolina State University where he teaches computer music composition and other courses.	He has used the program to create a number of pieces. Some examples that have been recorded include: • Composer, Belgrade Overture, performed by the Brno Philharmonic Orchestra, Mikel Toms, conductor. Released by Ablaze Records, November, 2013. • Composer, Winter Concerto, for trumpet, piano, strings. Performed by the London Schubert Players chamber orchestra, Huw Morgan, trumpet soloist; released on the CD, "As You Like It". London: RMA, 2011 and on the Nimbus (Wyastone, England) 2013 re-release on a 3-CD set called "A European Odyssey", NI6195. • Composer, "Music for Strings" compact disc. The Nevsky String Quartet of Russia. Brooklyn, New York: Capstone Records CPS 8781, 2007 (UPC 759348078126) Genetic algorithms used to compose the following pieces on that CD: I. String Quartet: Laredo II. Six Folksongs from an Imaginary Country (viola alone) III. String Quartet: Ha! Fortune • Singing in Traffic, performed by Steve Duke, soprano saxophone, on the "Evolutionary Computation" compact disc. London: Springer, 2007. • Saint Ambrose, a chamber opera in one act for saxophonist/actor and recorded electronic computer music. Performed by Steve Duke. Brooklyn, New York: Capstone Records CPS 8708, 2002. • He also used GenDash to make the one-act opera "Sappho's Breath", a Piano Concerto, and other pieces. Performances of works made with GenDash have taken place in New York City (the premiere of "Sappho's Breath"), St. Petersburg, Russia (Piano Concerto, the string quartet pieces), Chicago (Saint Ambrose), London (Winter Concerto), Belgrade (Belgrade Overture), many other places throughout the world.

Ref	Software Name	Author(s) Name	Background of Authors	Where is the Software used?
30	Maestro Genesis	Amy K. Hoover,	Hoover´s background of information is already given in page 81.	MaestroGenesis is used in the laboratory only, but we are trying to encourage amateur musicians to use it in their own works. The website for MaestroGenesis is provided below: http://maestrogenesis.org/
		Paul A. Szerlip,	Paul is Ph.D Graduate Research Assistant at University of Central Florida. He has school experience with different musical instruments.	
		Marie E. Norton,	Marie Norton was undergraduate music major when Maestro Genesis was implemented. She graduated with music degree.	
		Trevor A. Brindle,	Trevor Brindle was undergraduate music major when Maestro Genesis was implemented. Trevor graduated with music degree.	
		Zachary Merritt,	Zachary Merritt was undergraduate music major when Maestro Genesis was implemented then he switched to actuarial science.	
		Kenneth O. Stanley	Background information of Kenneth Stanley is already given in page 81.	

C Appendix: Data Extraction Form

General Information	
Identifier	
Title	
Author(s)	
Source	
Abstract	
Motivation	
Is this paper academic study or industrial study?	
What is the triggering effect?	
Problem Statement	
How are the others relate two the motivation and problem?	
How is the problem related to the field of GMCSS?	
To what extend is the problem related to GMCSS?	
Methods and Approaches	
Which biologically inspired computational method has been used?	
Is the software system for rhythm generation or music composition or both?	
Which programming language and environment has been used to implement the software?	
Features of the software system	
What are the main functionalities of the software system?	
What are the benefits of the software system?	
What are the limitations of the software system?	
Results and Conclusion	
Main findings	
Statements for the future research	
Suggestions	

D Appendix: Included Studies

Ref	Software Name	Paper
1	Nodal	J. McCormack and P. McIlwain, "Generative composition with nodal," in A-Life for Music: Music and Computer Models of Living Systems (E. R. Miranda, ed.), Computer Music and Digital Audio, pp. 99–113, A-R Editions, Inc., 2011
2	GP-Music	B. Johanson, "Gp-music: An interactive genetic programming system for music generation with automated fitness raters," in Proceedings of the Third Annual Conference, pp. 181–186, MIT Press, 1998
3	SBEAT	T. Unemi and M. Senda, "A new musical tool for composition and play based on simulated breeding," in Proceedings of Second Iteration, pp. 10–9, 2001
4	ANTracks	F. Schulz, C. Geiger, and H. Reckter, "Antracks: generative mobile music composition," in Proceedings of the International Conference on Advances in Computer Enterntainment Technology, pp. 302–305, 2009
5	GenJam	J. A. Biles, "Improvizing with genetic algorithms: Genjam," in Evolutionary Computer Music, pp. pp 137–169, Springer London, 2007
6	Patch Mutator	P. Dahlstedt, "Evolution in creative sound design," in Evolutionary Computer Music (E. Miranda and J. Biles, eds.), pp. 79–99, Springer London, 2007
7	NEAT Drummer	A. K. Hoover and K. O. Stanley, "Neat drummer: Interactive evolutionary computation for drum pattern generation," tech. rep., The AMALTHEA REU Program, 2007
8	Ossia II	P. Dahlstedt, "Autonomous evolution of complete piano pieces and performances with Ossia II," A-Life for music: music and computer models of living systems, 2012
9	Application of Genetic Algorithms	B. Jacob, "Composing with genetic algorithms," in International Computer Music Association, pp. 452–455, 1995
10	Spieldose	A. Sanchez, J. Pantrigo, J. Virseda, and G. Perez, "Spieldose: An interactive genetic software for assisting to music composition tasks," in Bio-inspired Modeling of Cognitive Tasks (J. Mira and J. Alvarez, eds.), vol. 4527 of Lecture Notes in Computer Science, pp. 617–626, Springer Berlin Heidelberg, 2007

Ref	Software Name	Paper
11	AMUSE (A MUSical Evolutionary assistant)	E. Özcan and T. Ercal, "A genetic algorithm for generating improvised music," in Artificial Evolution (N. Monmarche, E.-G. Talbi, P. Collet, M. Schoenauer, and E. Lutton, eds.), vol. 4926 of Lecture Notes in Computer Science, pp. 266–277, Springer Berlin Heidelberg, 2008
12	Variations	B. L. Jacob, "Algorithmic composition as a model of creativity," Organised Sound, vol. 1, pp. 157–165, 1996
13	Jive	J. Shao, J. McDermott, M. O'Neill, and A. Brabazon, "Jive: A generative, interactive, virtual, evolutionary music system," in Applications of Evolutionary Computation (C. Chio, A. Brabazon, G. Caro, M. Ebner, M. Farooq, A. Fink, J. Grahl, G. Greenfield, P. Machado, M. O'Neill, E. Tarantino, and N. Urquhart, eds.), vol. 6025 of Lecture Notes in Computer Science, pp. 341–350, Springer Berlin Heidelberg, 2010
14	Birdsongs	J. Fornari, "A computational environment for the evolutionary sound synthesis of birdsongs," in Proceedings of the First international conference on Evolutionary and Biologically Inspired Music, Sound, Art and Design, EvoMUSART'12, (Berlin, Heidelberg), pp. 96–107, Springer-Verlag, 2012
15	Neurogen	P. Gibson and J. Byrne, "Neurogen, musical composition using genetic algorithms and cooperating neural networks," in Artificial Neural Networks, 1991., Second International Conference on, pp. 309 –313, nov 1991
16	GeNotator	K. Thywissen, "Genotator: an environment for exploring the application of evolutionary techniques in computer-assisted composition," Org. Sound, vol. 4, pp. 127–133, jun 1999
17	Mezzo	D. Brown, "Mezzo: An adaptive, real-time composition program for game soundtracks," tech. rep., Artificial Intelligence and Interactive Digital Entertainment Conference, 2012
18	Sonomorphs	G. L. Nelson, "Sonomorphs: An application of genetic algorithms to the growth and development of musical organisms," in Proceedings of the Fourth Biennial Art Technology Symposium, vol. 155, 1993
19	Vox Populi	A. Moroni, J. Manzolli, F. Von Zuben, and R. Gudwin, "Vox populi: Evolutionary computation for music evolution," in Creative evolutionary systems, Morgan Kaufmann Publishers Inc., 2002
20	Rhythm Generation System	D. Horowitz, "Generating rhythms with genetic algorithms," in Proceedings of the twelfth national conference on Artificial intelligence (vol. 2), AAAI'94, (Menlo Park, CA, USA), American Association for Artificial In- telligence, 1994

Ref	Software Name	Paper
21	Music composition system with human evaluation as human centered system	M. Unehara and T. Onisawa, "Music composition system with human evaluation as human centered system," Soft Computing, vol. 7, pp. 167–178, 2003
22	Beads	O. Bown, "Experiments in modular design for the creative composition of live algorithms," Comput. Music J., vol. 35, pp. 73–85, sep 2011
23	ChaOS	E. R. Miranda, "Creative evolutionary systems," in On the origins and evolution of music in virtual worlds (P. J. Bentley and D. W. Corne, eds.), ch. On the origins and evolution of music in virtual worlds, pp. 189–204, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2002
24	Sound Gallery	S. Woolf and A. Thomas, "Creative evolutionary systems," in The sound gallery - an interactive a-life artwork, ch. The sound gallery an interactive A-life artwork, pp. 223–250, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2002
25	DOT	H. Roscoe, "Dot, a videogame with no winner," tech. rep., Hol, 2013
26	Musicblox	A. Gartland-Jones, "Musicblox: a real-time algorithmic composition system incorporating a distributed interactive genetic algorithm," in Proceedings of the 2003 international conference on Applications of evolutionary computing, EvoWorkshops'03, (Berlin, Heidelberg), pp. 490–501, Springer- Verlag, 2003
27	GenBebop	L. Spector and A. Alpern, "Criticism, culture, and the automatic generation of artworks," in *Proceedings of the twelfth national conference on Artificial intelligence (vol. 1)*, AAAI '94, (Menlo Park, CA, USA), pp. 3–8, American Association for Artificial Intelligence, 1994
28	GenJazz	K. Backman and P. Dahlstedt, "A generative representation for the evolution of jazz solos," in Proceedings of the 2008 conference on Applications of evolutionary computing, no. 10 in Evo'08, (Berlin, Heidelberg), pp. 371– 380, Springer-Verlag, 2008
29	GenDash	R. WASCHKA II, "Composing with genetic algorithms: Gendash," in Evolutionary Computer Music (E. Miranda and J. Biles, eds.), 978-1-84628- 599-8, pp. 117–136, Springer London, 2007
30	MaestroGenesis	A. K. Hoover, P. A. Szerlip, M. E. Norton, T. A. Brindle, Z. Merritt, and K. O. Stanley, "Generating a complete multipart musical composition from a single monophonic melody with functional scaffolding," International Conference on Computational Creativity, p. 111, 2012

E Appendix: Activities in the SLR

Time	Planning	Realization	Reporting	Outcomes
December 2011	Protocol Development			Review Protocol
		Data Retrieval		Repository With Articles
		Study Selection Upon Titles		
January 2012		Study Selection Upon Abstracts		
		Pilot: Data Extraction, 4 papers		4 papers reviewed
February 2012	Process Improvement			Draft: Data Extraction Form
		Revisit Reviewed Papers		
March 2012		Pilot: Data Extraction, 5 papers		9 Papers Reviewed
	Process Improvement			Definition Dictionary Refined: Data Extraction Form
April 2012		Revisit Reviewed Papers		
		Pilot: Data Extraction, 6 papers		15 Papers Reviewed
July 2012		Pilot: Data Synthesis		
December 2012			Pilot Report	
		Review: Data Extraction		20 Papers reviewed
January 2013		Data Synthesis		
February 2013	Process Improvement			
March 2013			Pilot Report	25 Papers Reviewed
		Review: Data Extraction		
April 2013	Process Improvement			
		Data Synthesis		
Msy 2013		Data Synthesis		30 Papers Reviewed
June 2013			Final Report	

F Appendix: Summary Of The Results

PL refers to programming language
PE refers to programming environment\\
N/A represents the information which can not be reached via reviewing the included study and interviewing with the author or authors of the included study.

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
1	Generative composition with nodal	Nodal	NNs	Complete Composition	N/A	N/A	• user creates a graph by using nodes and edges. • users control, structure processes in a compositional sense • user design dynamic graphic notation systems • user interpolate controller information as they travel along edges • change the musical composition while it is playing	• difficult tasks can be achieved by using conventional notation software • provides many possibilities for generating complex, emergent structures • simple interface • there are live performance tools in the system	• the system specifies the graphs in two-dimensions. • sequenced musical notes sounds mechanical

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
2	Gp-Music: An interactive genetic programming system for music generation with automated fitness raters	GP-Music	Combination of IEAs and NNs	Melody Generation	N/A	N/A	• user can derive short musical sequences • user can interactively evolve music • system achieves good results without the need of explicitly claryfing a lot of domain knowledge for a problem • user can rate individual sequences with the help of GUI	• with automated fitness raters, longer runs, operating both with and without user interaction, fully automated mode is provided • runs using only simple concatenation, using complex structuring functions is provided.	• before automated fitness raters, the user must have listened to and rated each musical sequence in every generation during a run • ratings of users are subjective, as a solution rating of individual is locked generation to generation • in previous versions, creation of some melodies were too long or too short. In the next version, individuals which do not meet the criteria are killed by the system.

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
3	A new musical tool for composition and play based on simulated breeding	SBEAT	EAs	Complete Composition	N/A	Pure Data	• population size • see, play and listen • migration and integration • genome editor • users select favorite individuals from a population as parents for the next generation • “Play all individuals” plays all individuals in the population sequentially • SBEAT can play eight instruments show notes of four instruments • user can edit chromosomes directly with the help of genome editor to have better results	• see, play, listen feature helps the beginners to learn to read the scores, helps musicians save time	N/A

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
4	Antracks: generative mobile music composition	ANTracks	Ant Colony Optimization Algorithm	Melody Generation	ObjC	Pure Data	• to generate harmonic musical expression virtual ants are used. • the user create ants. They start moving on the grid. • the user can place/change the objects on the grid and modify musical parameters. • the user can change parameters of tempo, pattern length, send notes on, note scaling.	• innovative • attractive • stimulates the user in a positive way	• WIMP, smaller interaction space • hardware restrictions on iPhone&iPod touch • usability is not good enough

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
5	Improvizing with genetic algorithms: Genjam	GenJam	EAs	Complete Composition	THINK C version 5	Moxc	• GenJam listens what the user plays during improvization then it maps the notes into the chromosomes. They help GenJam to generate a reply. The mutations develop idea for the user. • while GenJam is performing bass, piano, drums, string, and guitar is evaluated. • after GenJam generates a piece of music, the user evaluates the music.	• GenJam interacts with the user in real time in a effective way. • loudness threshold filters out the ambient noise in the room, so that GenJam pays attention to the close-miked trumpet. • when the user plays a good musical phrase, the mutations are also good. • GenJam does not play a wrong musical note unlike the human beings.	• fitness bottleneck • GenJam does not fit the criteria for the annual human competitive awards in evolution and genetic computation.

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
6	Evolution in creative sound design	Patch Mutator	EAs	Electronic Music Generation	N/A	N/A	• GUI of the Patch Mutator is divided into 5 different sections for usability • each sound has a visual representation to give a quick impression • for fast and efficient evolution all operations can be controlled with either through keyboard short-cuts or the mouse • temporary storage is one way of solving fittness bottleneck problem	• the actual synthesis parameter values are hidden from the user so that he focus on the sonic result and creative processes of the music • a new way of adjusting synthesis parameters by ear • modular architecture is more accesible to the beginner musicians • quick tweak tool • for sound design and synthesis interactive evolution is a feasible new paradigm	• it is frustrating when the user knows what exactly he wants because evolution does not take the user there • it is not the greatest tool for all sound design situations

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
7	Neat drummer: Interactive evolutionary computation for drum pattern generation	NEAT Drummer	Combination of IEAs and NNs	Rhythm Generation	C#	N/A	• NEAT Drummer is programmed without any expert musical knowledge. You can give it any MIDI and it will produce a drum pattern that follows the contours of the song.	• the benefit is that it can theoretically compose a drum pattern for any MIDI that you provide. The draw backs are that it does not work in real time and requires MIDI	• the draw backs are that it does not work in real time and requires MIDI

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
8	Autonomous Evolution of Piano Pieces and Performances with Ossia II	Ossia II	EAs	Melody Generation	N/A	N/A	• random variation is used during reproduction • after evolution the best score from the last generation is performed on acoustic grand piano • when the user plays melody on the piano keyboard, this melody is translated into a genome data structure • for each note the information of onset time, pitch, amplitude, note duration, articulated duration is stored • three types of mutations are used a genetic operators • the system supplies three mechanism for computing initial populations; random generation, human input, and recombination from a collection of “good examples”.	• formal methods help composer to prevent artistic stagnation • the system can generate and perform piano pieces • the recursive mechanism in the system provides clear thematic structures • the different sets of generative parameters and fitness target ranges create an interesting results • with the help of flexible representation and the different generative parameter sets and target ranges the musical output of Ossia II varies • system generates complex, vivid, musically convincing, interesting and novel results	• the system does not incorporate much • lack of correlation between superimposed structures • arbitrariness of the large-scale forms • statistical measures that form the basis for the fitness evaluation is not developed enough.

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
9	Composing with Genetic Algortithms	Application of Genetic Algorithms	EAs	Melody Generation	N/A	N/A	• the system uses three different EAs. The first EAs composes phrases (melodic material), the second EAs controls the harmony, valid chords and chord transitions (chord progression), and the third EAs generates a form out of that material (adjust this material). • this system is an example of making home compositions. • in this system EAs are used to generate a set of data filters. The goal of using EAs is to analyze the entire potential solutions to find one which fulfils the criteria. An important point is to arrange the set of all potential solutions. • algorithmic composition system variations are used. • this system applies to microtonal music.	• the system is very flexible. It should be noted that the general representation of valid combinations does not depend on the choice of a twelve-tone octave. Microtonal vertical pitch combinations can be represented by using a different number of bits.	• the biggest problem of using EAs is the size of the search space. Outstanding EAs music applications have restricted goals. The reason is that the problem domain becomes large instantly. Therefore, convergence to an adequate solution may take a lot of time. Researchers solve that problem by decreasing the amount of the problem domain. However this study handles the problem differently; EAs which are used in this application deal with larger building blocks.

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
10	Spieldose: An interactive genetic software for assisting to music composition tasks	Spieldose	IEAs	Complete Composition	N/A	MATLAB	• adapted GA is used as an optimization method. • during the genetic evolution, the user selects several good melodies according to musical subjective criteria. This process is repeated in each iteration. By this way, until the termination criteria, an initial population of automatically generated compositions is evolved. • Spieldose tries to add criteria of musical composition into the Interactive GA • main steps of Interactive GA are initialization, selection, crossover, mutation, improvement, and invasion.	• there is a variety and its effective implementation of the operators in the Interactive GA. • GUI of Spieldose offers the user appropriate functionality for the musical composition task and also the GUI hides the implementation details of the interactive GA. • generation of melody population • it enables the user to listen to generated music interactively • to select the best melody subset by roulette wheel • at the end of each iteration, the best melody is saved in wav or text format	N/A

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
11	A genetic algorithm for generating improvised music	AMUSE (A MUSical Evolutionary assistant)	EAs	Melody Generation	Java	Eclipse	• AMUSE is a system for generating improvised melodies over a musical piece given in a harmonic context. • to generate melodies automatically without a human feedback, AMUSE combines a modified representation scheme and different fitness objectives under a GA approach. • AMUSE tries to add criteria of musical composition into the Interactive GA • core objectives are; Chord note (f1), relationships between notes (f2), directions of notes (f3), beginning note (f4), ending note (f5), over fifth (f6) and drastic duration change (f7). Adjustable objectives are Rest proportion (f8), hold event proportion (f9) and pattern matching (f10).	• the advantage of using modified representation scheme is that it is impossible to generate non-scale notes. AMUSE does not need a human for getting feedbacks	N/A

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
12	Algorithmic Composition as a Model of Creativity	Variations	EAs	Harmony Generation	N/A	N/A	• Variations is an algorithmic composition system which initiates to model the hard work type of creativity. The system was implemented to reproduce creative melodies which the author uses when composing music. • the system works at the level of music motives. This simplifies the organization of the music. • the system allows the user to pay more attention to create harmonic progression. • there are two primary software components in the system which are COMPOSER and EAR modules. • The system uses an evolutionary algorithm, paired with domain-specific knowledge, and implemented as quite an interesting system.	• it allows to a human composer to work more quickly. Its creative success depends on two phenomena: 1) A great match between creative methodology of the composer and the implemented algorithm. 2) An accurate mechanism for making decision quickly about the viability of a specific musical phrase.	• because each composer has a unique compositional process, one algorithmic tool does not fulfill the demands and requirements of many different composers.

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
13	Jive: A generative, interactive, virtual, evolutionary music system	Jive	IEAs	Complete Composition	Java	Eclipse	• the Jive system has four components: generative, interactive, virtual, and evolutionary. • the system works at the level of music motives. This simplifies the organization of the music. • the system allows the user to pay more attention to create harmonic progression. • there are two primary software components in the system which are COMPOSER and EAR modules. • the system uses an evolutionary algorithm, paired with domain-specific knowledge, and implemented as quite an interesting system.	• the system creates the music in a sufficiently interesting way. • since the system generates material continuously, with no wrong notes and precise timing, the user is freed from low-level details. • the user obtains a higher-level type of control despite the lack of low-level details.	• the system is clearly lacking in the area of rhythm. • the system is not user-friendly: most musicians cannot perform symbolic regression in their heads while composing. • while performing, the user can not insert or delete arbitrary notes.

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
14	A computational environment for the evolutionary sound synthesis of birdsongs	Birdsongs	Combination of NNs and EAs	Melody Generation	N/A	Pure Data	• genetic operators dynamically generate sequences of control parameters for computational models of birdsongs, given by the physical model of a syrinx. • this system can emulate a wide range of realistic birdsongs and generating with them a network of bird calls. • psychoacoustic distance is used as the fitness function, through which metric individuals inside the population are selected. • selection process measures the distance between each individual in the population.	• this system is able to generate artificial soundscapes compounded of synthesized birdsongs. It allows the interactivity of multiple users. This creates a feedback between users and the EA system.	N/A

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
15	Neurogen, musical composition using genetic algorithms and cooperating neural networks	Neurogen	Combination of NNs and EAs	Complete Composition	C++	Max/MSP	• in Neurogen, a set of Neural Networks are used to capture conceptual ideas. Those ideas build good music and this knowledge is then used to direct a search for the ultimate composition. • the software developers of Neurogen construct a Neural Network model; this model can learn the good characteristics from a set of good and bad model compositions. • once the Neural Network has completed its learning phase, it can be used as a guide for the Genetic Algorithm. After that, they apply genetic operators of reproduction, crossover and mutation to generate better compositions based on the heuristic values provided by the Neural Network.	• the system just focuses on a particular form and composition style. This form and style of musical composition has limited constraints. And this means that there is less computational complexity.	N/A

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
16	GeNotator: an environment for exploring the application of evolutionary techniques in computer-assisted composition	GeNotator	IEAs	Complete Composition	C++	N/A	• GeNotator has two different interactive levels: meta-composer and gardener. A meta-composer is interested in analytical understanding about the form and structure of a composition. A gardener is just interested in the satisfaction of the user. • structurally, GeNotator has Genotype Structure Definition (GSD) in the centre of its architecture. The GSD is a data structure that packages a user-defined music grammar. • once defined, the GSD serves as input to the Form Space Manager. • in order to evolve favored instances, the user can judge and score results and continue to produce iteratively. • GeNotator enables the user to mix and match between a text-based grammar and the graphical approach within the same project. • the GUI components of GeNotator are very powerful.	• by the help of powerful GUI of the GeNotator, the user can interactively compose music. • the user can modify genetic operators via this GUI. • GeNotator is a fairly flexible tool for the user.	N/A

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
17	Mezzo: An adaptive, real-time composition program for game soundtracks	Mezzo	EAs	Complete Composition	Python	MAX/MSP	• main motives of the game music are related with elements and game characters. These main motives are mapped into various musical forms. These forms are recognized by different amounts of harmonic tension and formal regularity. • for each round of game, main motives of the game music were input to be related with game elements and characters, and a set of clues was written. These clues include a set of time points at which a new set of game data would be passed to Mezzo to demonstrate the action of the game trace. • music composition in Mezzo is made in two steps: 1) Build forms and 2) Deform them according to stochastic constraints. Both of these processes generate artistic properties in the music being composed. • the only time when Mezzo uses a genetic algorithm is when it makes harmonic progressions. The author mentions that he used this because making these progressions is a highly constrained problem over a very big search space, and it needs to be done quickly.	• Mezzo has open-ended setting of a game. Each time a form is stated, it is organized differently from previous times. There is no pattern of the way the organization changes from statement to statement. • it composes fully realized music in real time for a game, and this music adapts to game play. • Also, it uses musical theories of form and meaning that have not been used in computer-generated music till now.	• as it is stated in the benefits above, Mezzo has open-ended setting of a game. Each time a form is stated, it will be organized differently from previous times. This causes a certain quality of irregular formal organization. • it does not orchestrate the music (yet), and so the sound output is not that interesting. The author mentions that he has been working on these issues.

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
18	Sonomorphs: An application of genetic algorithms to the growth and development of musical organisms	Sonomorphs	EAs	Melody Generation	C	MAX/MSP	• the aim of this study was to find optimum methods for structuring musical organisms. • an algorithm generates musical structures and interprets the genetic code. This genetic code is passed to each generation. • the genetic code is embedded onto musical parameters. The composer uses this code for subjective aural evaluation. • the genetic model of evolving rhythmic patterns uses a bit summing test. If a bit is switched on, a note is articulated; if a bit is switched off, a rest is made. • the bits are combined with crossover method. One of the two parents is chosen with a coin toss for beginning the breeding. A coin is tossed again when the first pair of bits is considered. If it is tails, no crossover occurs. The first bit for the child´s genome is taken from the first parent. If the coin turns head up, a crossover occurs and a bit is taken from the opposite parent. • selection of parents is performed by random walk method.	• they used mutation and migration operators in this study. And these methods provide diversity on a gene pool. • the proposed system has a lot of controls on graphical user interface; these controls provide a multiple toggle.	• the breadth of the field is so great that it is difficult to focus for very long on simple examples and the extraction of basic principles. • this system is probably not a powerful tool for making large compositions. The operations are certainly too limited and too simple to make sophisticated musical utterances.

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
19	Vox populi: Evolutionary computation for music evolution	Vox Populi	IEAs	Complete Composition	Java	Eclipse	• Vox Populi is an evolutionary based system for composing music in real time. A population of chords is decently codified according to the MIDI protocol. • a fitness function is defined to find the best chord in each generation. The best chord is selected as the next element in the sequence to be played. Each new generated chord is a new sound palette. Musicians can use this new sound palette to continue the music evolution. • Vox Populi becomes a musical instrument, but unlike a traditional instrument, Vox Populi is able to create its own sound chord population and to provide choice criteria (music fitness) simultaneously. • Vox Populi allows the user to modify the fitness function by means of four controls: These controls are melodic criterion, the duration of the genetic cycle and musical rhythm, the set of octave ranges to be considered and the time segment for each selected orchestra. • Vox Populi uses the computer and the mouse as real-time music controllers. It produces dynamic musical structures based on evolutionary models. • this system is a new interactive computer-based musical instrument. • it has a strong graphical interface to change the musical evolution.	• by graphic controls (pad and sliders), the system becomes user-friendly to manipulation of the fitness and of the sound attributes. • evolutionary computation is used to stimulate the user with novel sounds. It allows the user to respond. • by the features of Vox Populi, this system enhances the users music capabilities and marks this system as the state of the art in computer music. • all controls of Vox Populi are available for real-time performance, allowing the user to play and interact with Vox Populis music evolution. • the interactivity of Vox Populi emphasizes aspects of musical practices in the scope of human/ machine interaction.	N/A

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
20	Generating rhythms with genetic algorithms	Rhythm Generation System	IEAs	Rhythm Generation	C++	Code Warrior IDE	• the set of rhythms satisfying the criteria of the user is represented by a boolean formula • the rhythms which are created by the system are played to the user. User assigns fitness values to the each generated rhythm according to his satisfaction. Then the system uses GA selection, reproduction and mutation operators • objective functions to explore creative rhythms	• in this system, rhythms are one measure long sequences of notes. The software developer only deals with a specific subset of the tremendous class of rhythms. The goal is to obtain a well defined domain for the application of the learning algorithm. The benefit of this minimization of the domain is that a rhythm phenotype can be viewed as a simple vector so that the set of rhythms satisfying the criteria of the user can be represented by a boolean formula.	• the implementation of appropriate fitness functions is not efficient enough

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
21	Music composition system with human evaluation as human centered system	Music composition system	IEAs	Complete Composition	N/A	N/A	• the human plays a major role in judgment, evaluation, recognition and emotion steps. The composition has 3 main procedures: 1) Two hundred chromosomes are generated based on general music theory. 2) A user listens to 20 musical works chosen from 200 works and performs three types of evaluations such as total evaluation, partial evaluation, and the choice of the best work. 3) The system performs EAs operations on 200 chromosomes reflecting these evaluations. • procedures 2 and 3 are repeated until a musical work projecting users' evaluation is achieved. These steps show that this is a human centered system.	• this system is a human centered system and the human has an important role in construction of musical compositions. Because of this, the system can be easily personalized according to the choices of the user.	• users' fatigue. In this approach, users have to listen to the music one by one. Users have to take a lot of time listening almost the same melody as he or she already listened. • the solution space, that is, musical variation. Users have to choose many aspect of music, such as instrument, background music, tempo, and so on. This is a human depended system and this system does not guarantee perfect melodies. It depends on musical background of the user.

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
22	Experiments in modular design for the creative composition of live algorithms	Beads	IEAs	Complete Composition	N/A	Max/MSP	The approach of the author is very ambiguous with respect to specific functionalities, e.g., tracking beat, finding key, continuing on the style of the performer. None of these things are done by the system. The author approaches the final system more as a creative composition of a system by himself. Performers have reported to have a great experience of interaction with it in terms the surprising responses it generates.	The author has reported that the system is beneficial in terms of design methodology. He does not believe in creating an ultimate live algorithm, and he thinks the innovation in design methodology is currently required. In this case because the behavior is the product of evolution towards a targeted fitness function, its design is detached from the designer, so it has a kind of functional autonomy.	The author explains that the system is hugely limited in many dimensions. Since he treats making the algorithms a form of composition, it is like asking what the limitations of a given melody are. However, it is relatively flexible as a kind dynamic behavior that can be adaptively used in different contexts.

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
23	On the origins and evolution of music in virtual worlds	ChaOS	CA	Sound Synthesizer	C and C++	NetBeans	• the research is about musical forms originated and evolved in artificial worlds. The music making term is used for both creating and listening music. Natural selection in biology is the effect of new music making. • social evolution, a much complex phenomenon is considered. Music is the interaction of agents engagement in music making. • in transformation, the entity information is preserved. Co-evolution used for interaction between various transformation and selection pushes the system to more complexity. • in self organization, with feedback, a fluctuation is strengthened and became more predominant. • computer sound synthesis technology allowed sound control fundamentally. Granular synthesis involved tiny sound granules and exhibits sensible movement and flow. Self organization is used for controlling the evolution of the granules. • cellular automata are modeling techniques being used in systems which space and time represented discretely. CA are implemented as array or matrix of cells and associated with a color.	• ChaOs is an acronym for Chemical Oscillator, an adapted version of a cellular automata used to model the behavior of a number of oscillatory and reverbatory phenomena. An oscillator needs three parameters to function: frequency, amplitude and duration. • in order to produce sounds ChaOs resembles the evolution of acoustic instrument's harmonics that converge from a wide distribution to oscillatory patterns.	• it does not work in real-time. • the last version is for an older operational system which does not run anymore.

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
24	The sound gallery - An interactive a-life artwork	Sound Gallery	EAs	Sound Synthesizer	C++	NetBeans	• the Sound Gallery has two important algorithms: 1) **Hill-Climbing Phase**: Four initialization genotypes are generated, one for each of the four sub populations. Hill climbing then commences, with each sub-population working in parallel to the other three. The initialization genotypes undergo repeated mutations, generating new genotypes which presents new TRAC configurations. The Sound Gallery uses the Zetex Totally Reconfigurable Analog Circuit (TRAC). Each new genotype is evaluated and assigned a fitness value. When a mutation is evaluated to be fitter than, or equally fit to, the parent genotype from which it was derived then this mutant genotype is stored as the next member of its sub-population and will be used as the source for subsequent mutations. 2) **Island Model Genetic Algorithm Phase**: Linear rank based selection is used to select two parent genotypes from each island sub-population. Child genotypes are derived from each pair of parents through the application of mutation, replication and crossover genetic operators. The TRAC development board is reconfigured so the circuit specifications represented by each of the new child genotypes are physically manifested in silicon. Each of the four circuits on the TRAC development board are then allocated fitness values, and the new genotypes replace the least fit members of their respective island sub-populations. This sequence of events presents one iteration of the algorithm.	• volunteers made experiments about Sound Gallery. Volunteers enjoy and find enthusiastic while exploring the different, varied and interesting sounds. Sound Gallery captures the attention and imaginations of a group of volunteer participants. The project proved itself capable of producing a lot of large repertoire of interesting distortion effects and sounds.	N/A

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
25	Dot, a videogame with no winner	DOT	EAs	Melody Generation	Java	Arduino environment	• the artist and the invited guests are able to create all sounds and images live • all the live audiovisual parameters can be controlled using joysticks • each part of the performance has a special score, where the functionalities of each button is shown to the players • the system is autonomous and doesn't need a computer to work • all images and sounds are linked to the performance concept though metaphorical relationships • there are no pre-recorded images or sounds. All the content is generated in real time	• the players can participante without previous knowledge of the mais principles of the performance. • winning the "game" is not the main objective, but participating and contributing to the success of the performance • the instrument is autonomous and doesn't need a computer to work	• the generated images have low resolution (400x300) • the sounds are limites to 64 voices that can be choosen from a sine wave or noise • the number of sprites and colors on the images are limited • all images have to be converted into binary information in ordem to enter the arduino enviroment

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
26	Musicblox: a real-time algorithmic composition system incorporating a distributed interactive genetic algorithm	MusicBlox	IEAs	Complete Composition	N/A	N/A	• the MusicBlox is a project which uses blocks like childrens wooden building blocks. These blocks are combined together in a similar way to make physical structures. • when the first block is created, it sends its result to the second block and the second block recomposes itself, and then the second block sends its result to the third one and the third block recomposes itself. This process continues iteratively. At the end, the collective music of the structure is transformed • the home music of the block is used to create a population of identical phenotypes, which build the initial population. • mutation and crossover operations are performed on the selected population member. • the similarity between the phenotype and the target is defined as fitness function. If the fitness value for the mutated population member is higher than the lowest fitness found in the population, it replaces the low fitness population member, and is stored as a musical point on the evolutionary path to the target; otherwise, it is discarded. • by each block possessing its own GA, and passing the output of one as the target input of another, the whole system becomes a kind of distributed IGA.	• this system is a kind of distributed IGA because each block possesses its own GA, and passes the output of one as the target input of another. And this means that MusicBlox has an interactive learning process. And learning process depends on the need of users	N/A

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
27	Criticism, Culture, and the Automatic Generation of Artworks	GenBebop	EAs	Melody Generation	Common Lisp	Macintosh Common Lisp	• in this study they proposed a system which produces bebop jazz melodies from a case-base of melodies with genetic programming. • their fitness function is based on user-provided critical criteria. Aesthetic judgment is a problem of constructing artists systems. In this study, they tried to by-pass aesthetic judgment. For this purpose, the proposed system takes user-provided criteria and guaranties to produce proper melodies. • Spector and Alpern just used reproduction and crossover functions of genetic programming. The reproduction operator selects the best individual and keeps it into the next generation. In addition, by crossover operation; they provide variations of the population.	• as mentioned before, the major problem of this kind of automatically generated melodies is aesthetic judgment. The proposed study takes criteria and constricts them as input, so they do not need to judge the output.	• the lack of robustness is a limitation of this study.

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
28	A Generative Representation for the Evolution of Jazz Solos	GenJazz	IEAs	Melody Generation	C++	.NET with DirectX	• the aim of this study was to create computer based jazz improvisation solos. • they used interactive evolution in their study. • the computer has generated solos. Then these solos have been imported into a musical environment. After importation, the result can be listened and evaluated.	• this system provides interesting and unexpected artistic outputs. • according to history, there are a lot of rules for jazz music solos and all outputs of this study fulfill the conditions. • by using computers in producing jazz music, it opens your mind to new thinking.	N/A

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
29	Composing with Genetic Algorithms: GenDash	GenDash	EAs	Complete Composition	ObjC	Pure Data	• GenDash is a program which has been revised several times according to authors needs. In general it has been used to help compose pieces of music. For some pieces, GenDash provided the total algorithmic support. For other pieces, the author might have used GenDash for one aspect of the work, such as the instrumental part of a composition, while employing a different program and algorithm for the electronic portion. • GenDash has ten attributes; An individual consists of a measure of music, all individuals that are born in any generation are performed, the fitness function is random, only one crossover point is used for each breeding, space is set aside for individuals that are unheard in the current generation but may appear and/or breed in a later generation, space is set aside for an intact individual that may breed in the current generation and in a succeeding generation, individuals within a single generation can mate with more than one other individual and/or mate with the same individual more than once, mutations can occur and finally, the composer chooses the initial population.	• GenDash is a flexible program and the user can use this program according to his/her needs. • the user is able to create a significant body of new art music based on evolutionary computation.	• after a regular concert series have been arranged, the concert hall has been rented and the performers have been paid to play, the amount and cost of rehearsal time have to be considered as an important budget factor. This factor can be a limitation for the composer.

Ref	Title of the Included Study	Name of the Software	Biologically Inspired Computational Method Used	Different Types	PL	PE	Main Functionalities	Benefits	Limitations
30	Generating a Complete Multipart Musical Composition from a Single Monophonic Melody with Functional Scaffolding	Maestro Genesis	IEAs	Complete Composition	N/A	N/A	• this study enhances the state of the art for a computer-assisted approach to music generation called functional scaffolding for musical composition (FSMC). It is a method for generating music which concentrates on harmonization and accompaniment. It is based on the idea that music can be represented as a pattern which is a function of time, i.e. f(t). • FSMC has a powerful representation like creative combination, exploration, and transformation of musical concepts. • FSMC represents music as a functional relationship between an existing human composition, or scaffold, and a generated accompaniment. • this relationship is encoded by compositional pattern producing network (CPPN). CPPN is a type of artificial neural network. • a human user can generate polyphonic compositions from a single, human-composed monophonic starting track. • FSCM facilitates creative exploration by helping the user construct and then navigate a search space of nominee accompaniments through a breeding process similar to animal breeding. This process is also called interactive evolutionary computation (IEC). • MaestroGenesis can add pitched accompaniments to as little as a single monophonic starting melody.	• the user can easily generate polyphonic compositions from only a single original monophonic track provided by the user. • a human user without any musical expertise can use the FSCM system effectively and then gain accepted results.	N/A