

10 Intelligent Music Composition

*Maximos A. Kaliakatsos-Papakostas*¹,
*Andreas Floros*² and *Michael N. Vrahatis*¹

¹Department of Mathematics, University of Patras, Patras, Greece;

²Department of Audio and Visual Arts, Ionian University, Corfu, Greece

10.1 Introduction

Automatic music composition is an enchanting field of research, inspiring both researchers and musicians. The implementation of systems that perform this task, incorporates an algorithmic part which makes decisions on which notes/sounds will be heard, when they will be heard, for how long and how loud. The intriguing part of automatic composition is the process of formulating such algorithms, in such a manner that the composed music is not too simple or too complex. Referring to the terms “simple” and “complex,” someone may find her/himself sort of words when speaking about music. Nevertheless, almost everyone can more easily express an opinion about the complexity of a certain music excerpt. This fact raises the question of how we may be sure that an automatic composition system composes too simple or too complex music. Additionally, how may we be able to compare the compositional capabilities of two such systems? Is there an objective criterion to characterize how “pleasant” is the music produced by such a machine?

These questions among others have led the research community toward a direction to construct systems that have inherited some kind of *intelligence*, which allows them to construct music with a pleasant blend of simplicity and complexity. Specifically, these machines incorporate an intelligent algorithm, which is commonly able to create mathematical objects that may vary from numeric values to abstract string sequences. The underlying intelligence of these algorithms forges the aforementioned objects in a manner that they either produce present rich and complex structure or comply with certain target characteristics defined by humans. These objects could then simply be mapped to the sonic domain and produce subjectively pleasant music, or adjust the parameters of music composition systems and lead the composition process to certain stylistic and aesthetic directions. This chapter presents several theoretic and technical aspects of such systems and proposes their categorization according to their intelligence and the way that this

intelligence adheres to music composition. The main motivation for this categorization is the detailed orientation specifications of all the intelligent composition methodologies, which will hopefully facilitate the new-coming researchers and artists interested to enter this field and provide more robust guidelines to the production of new methodologies.

The functionality of the intelligent part of these automatic music composition systems is a parameter that can be used to categorize them into three main categories, depending on the manner that this intelligence is expressed and manipulated. To this end, we propose the following categorization for intelligent music composition (IMC) systems:

1. *Unsupervised intelligent composition*: The algorithmic composition intelligence is expressed through simple rules that produce complex, unpredictable, but yet structured output, a behavior that often resembles natural phenomena.
2. *Supervised intelligent composition*: Intelligent algorithms are utilized to modify the automatic composition system's parameters so that it may be able to compose music that meets some predefined criteria, not necessarily in real time.
3. *Interactive intelligent composition*: The system is acknowledging the human preference in real time and becomes adapted to it, by utilizing intelligent algorithms. Human preference is expressed either by a selection-rating scheme or by performing tasks (such as playing an instrument or adjusting target parameters in real time).

It has to be noted that this chapter focuses on intelligent algorithms that have fostered the creation of numerous automatic composition methodologies. There are some works that are eligible to be included in the IMC methodologies family and have produced interesting results, but their proposed direction has not been extensively used or has been abandoned for the last few years. For example, the utilization of artificial neural networks (ANNs) for music composition (Griffith and Todd, 1999) has not been extensively used in the last decade.[†] Additionally, there is not yet extensive literature concerning composition through models that utilize ant colony optimization as a music composition tool, i.e., in Geis and Middendorf (2008). Several methodologies that this chapter discusses have been implemented to software. Since the aspect of this chapter is purely methodological, the interested reader is referred to the bibliography for these implementations, although several of them are hardly accessible due to outdated web links or platform dependencies. Additionally, the bibliography incorporates many musical examples and links to audio material.

Based on the above considerations, this chapter aims to provide a categorization of IMC techniques in terms of the functionality of each intelligent algorithm employed and the compositional potentialities—limitations that they incorporate. Therefore, this chapter is mainly targeted to the researcher, the musician, and the student of computer science or music informatics. The researcher may find such a categorization interesting and hopefully increase the convenience of literature organization. The musicians, the option to integrate computational intelligence systems

[†]This comment refers to the utilization of ANNs as a means to create note sequences, not as automatic fitness raters as discussed later.

in their artistic arsenal, may be informed about the state of the art of IMC and furthermore be facilitated with the extensive amounts of literatures that a nonexpert in informatics has to face in practice. Finally, the student may directly decode the potentialities and limitations of IMC and incrementally obtain the required information to become an expert on the field of IMC or other related fields.

10.2 Unsupervised Intelligent Composition

This class of IMC systems incorporates algorithms inspired by natural phenomena, encompassing simple rules. If these rules are applied recursively, they tend to produce interesting results with rich and complex structure. The fundamental motivation toward using these systems as music composition tools mainly originates from the rich geometric structures that they produce and the corresponding visual representation outcome, which is often observed in nature. Thus the characterization of these systems as “intelligent” is based on the fact that their simple sets of rules trigger complex and sometimes unpredictable behavior, which exposes an underlying endogenous intelligence. The main question that the composing software has to face is the interpretation of this complex output to music objects.

An advantage of these systems is their ability to compose novel and quite intriguing music content. This ability is maintained almost regardless of the interpretation modeling that the composer–programmer is applying. This fact, however, creates a vast disadvantage, as it disallows any human involvement in the composition process. The composer is thus unable to have a sense of control over his compositions. Therefore, the integrity, the style, and the aesthetic value of the music produced by such systems are only dependent on the subjective taste of the listener. Since there are no specific target music characteristics, these compositions are rarely submitted to rigorous evaluation and the results mainly revolve around presenting scores of small music excerpts. Thus, for the majority of works in this field, there are no qualitative descriptions of the compositional capabilities of such systems.

10.2.1 *Unsupervised Composition with Cellular Automata*

Cellular automata (CA) represents a paradigm of systems that encompass simple rules of interaction between a “unit” with its neighboring “units,” which are yet able to produce complex behavior. CA can be viewed as simple forms of “discrete” societies, where each individual unit occupies a certain cell on a grid and may have a discrete state. This state is updated in successive time steps in accordance with the state of the unit’s neighbors. The dimensionality of the aforementioned grid determines the number of each unit’s neighbors (e.g., in a one-dimensional grid, each unit has two neighbors, the unit on its left and right), while the number of states determines the overall system’s dynamical behavior. All possible dynamics

that may be produced by CA have been classified by [Wolfram \(2002\)](#) in the following categories:

1. Patterns disappear with time or become fixed.
2. Patterns evolve to structures that repeat periodically, cycling through a fixed number of states.
3. Patterns become chaotic and never repeat, forming aperiodic and random states.
4. Patterns grow into complex forms, exhibiting localized structures moving both spatially and temporally.

Music composition has mainly been realized with the utilization of single-dimensional grid worlds with binary states known as “elementary cellular automata” (ECA), two- or three-dimensional worlds with binary states also known as “game of life” (GL), and two- or three-dimensional worlds with p discrete states, often referred to as “crystal growth” (CG) due to the crystal-like form that their visual interpretation exhibits. In the case of the binary states, the state numbers 0 and 1 are assumed, denoting a “dead” or “alive” cell, respectively. The state of each cell is updated in every tick of the clock and is determined by the cell’s own state, as well as the state of its neighbors. Similarly, the discrete states of the CG CA is updated with the “domination” of the cells in state $s + 1$ over its neighbors of state s .

Specifically, for the ECA, [Wolfram \(2002\)](#) has documented all possible 256 evolution rules and analyzed the dynamical behavior of these single-dimensional societies. Even though these rules are simple and deterministic, their application results in complex patterns often leading to chaotic behavior, like Rule 30[‡] the evolution of which is depicted in [Figure 10.1A](#).

The evolution of the GL CA abounds in dynamical phenomena, ranging from stable to oscillating and chaotic patterns, which derive from the simple application of the following rules:

1. *Death by underpopulation*: Any alive cell with fewer than two live neighbors dies.
2. *Death by overpopulation*: Any alive cell with more than three alive neighbors dies.
3. *Survival*: Any alive cell with two or three alive neighbors lives on to the next generation.
4. *Birth by reproduction*: Any dead cell with exactly three alive neighbors becomes alive.

The application of these rules is graphically presented in [Figure 10.1B](#), beginning from a random initial population of cells. In a similar fashion, the state of a cell (an integer value ranging from 0 to s) in the CG CA determines the state of its neighbors. If a neighboring cell is in state $s - 1$, then, at the next tick of the clock, it will be converted to s . This rule acts circularly, meaning that the state 0 dominates over the state corresponding to the greatest integer. The application of these rules results in crystal-like oscillating patterns, an example instance of which is depicted in [Figure 10.1C](#). The rules of the GL and CG CA can also be applied on hypercubes of arbitrary dimensions; for music composition though, only two- and three-dimensional versions have been tested ([Burraston and Edmonds, 2005](#); [Miranda and Biles, 2007](#)). The dynamics of the emergent social behavior of these discrete societies has attracted

[‡]Rule 30 is also used as a random number generator in the mathematical software Mathematica.

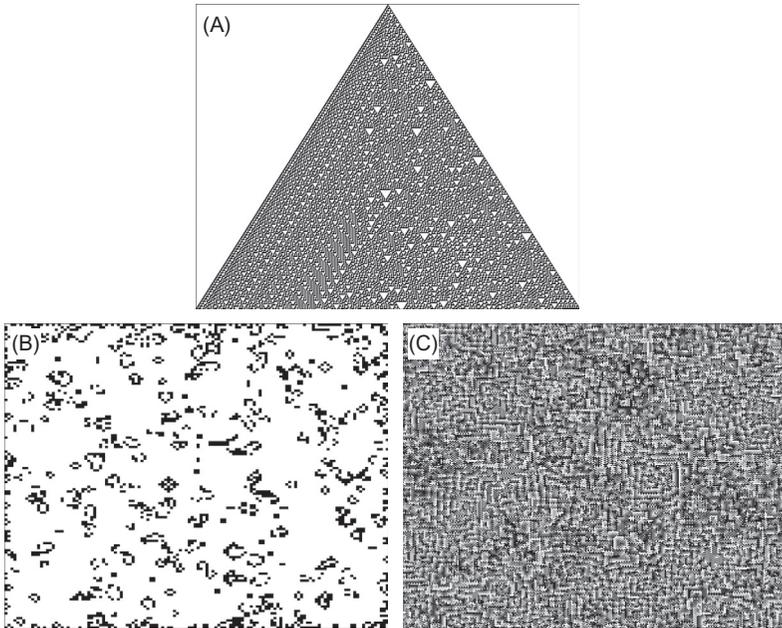


Figure 10.1 Graphical representation of CA. (A) ECA: Evolution of Rule 30 with a single alive cell at the first step. (B) GL: Evolution from a random initial set of alive cells. (C) CG: Evolution of a random initial six-states cell society, the states of which are mapped in a grayscale image.

the attention of composers ever since the epoch of the pioneers of automatic computer composition, like Xenakis (Solomos, 2005). For an extended and thorough review of the methodologies for music production through CA, the reader is referred to Burraston and Edmonds (2005) and Miranda and Biles (2007).

The systems suggested so far for music composition and sound synthesis using CA either utilize the states of the CA to trigger Musical Instrument Digital Interface (MIDI) commands or map these states directly to a sound generator in order to structure a waveform or to perform granular synthesis. In the first case, the graphical representation of the CA is mapped to notes (or groups of notes called chords), onsets, durations, and intensities, with this information assigned to certain instruments. In the latter case, the output of the CA activates specific oscillators with certain frequency–amplitude pairs, or it is projected to certain sound granules and their properties, i.e., sound material of short duration and information regarding their amplitude, length, and repetitiveness.

10.2.2 *Unsupervised Composition with L-Systems*

The L-systems are parallel generative grammars (Prusinkiewicz and Lindenmayer, 1990) with some variations that allow the production of interesting patterns that

Table 10.1 Example of Simulation of a DOL-System for Three Iterations

V :	$\{A, B\}$
ω :	AB
P :	$A \rightarrow AB$ $B \rightarrow A$
(0)	AB
(1)	ABA
(2)	$ABAAB$
(3)	$ABAABABA$

resemble plant-like forms and fractals. Thus, they incorporate a form of intelligence that assimilates natural creativity, a fact that makes these systems eligible for music composition. L-systems generate sequences of symbols (or words in an alphabet) that are then interpreted to music or sound elements, projecting the rich structure that these strings encompass to the sonic domain. The L-systems belonging to the simplest form are called *deterministic context-free* (DOL-systems). In these systems, a set of symbols called alphabet is defined, V , and each symbol is associated with a rewriting rule in a set of rules, P , which are in turn applied to a nonempty word in the above-mentioned alphabet, $\omega \in V^+$, creating a new word. A DOL-system can be described as a triplet $G = (V, \omega, P)$. This procedure is applied recursively, creating new words with new length which may then be transcribed into graphical or music/sound entities. An example of an L-system with the above form is demonstrated in [Table 10.1](#).

The consideration of special symbols which are interpreted into special graphical or musical functionality produces special representations like that illustrated in [Figure 10.2](#). These symbols are translated into straight lines, turning angles of some predefined degrees and symbols that are not considered in graphical projection but function instead as “helping” variables. Among the pioneering works on music production with the L-systems is the work of [Prusinkiewicz \(1986\)](#), who created music scores by traversing the curves produced by the graphical projection of various L-systems. Since then, various models have been proposed for the transcription of the L-systems’ rules to music, from note-to-note composition to the creation of chord progressions based on the neo-Riemannian music space ([Gogins, 2010](#)).

Variations of the L-systems have also produced interesting musical results. For example, the introduction of more variation was achieved by the utilization of probabilistic grammars ([McCormack, 1996](#)), where each symbol is associated to several rewriting rules according to some probability. Furthermore, musical diversity has also been accomplished with genetic evolution of the rules in [Lourenc and Brand \(2009\)](#). In this work, no fitness function was provided, thus the production of music is considered “unsupervised,” in contrast to some similar techniques incorporating evolution of L-systems in [Section 10.3.1](#). For these systems, however, there is lack of a rigorous evaluation and the results revolve around presenting

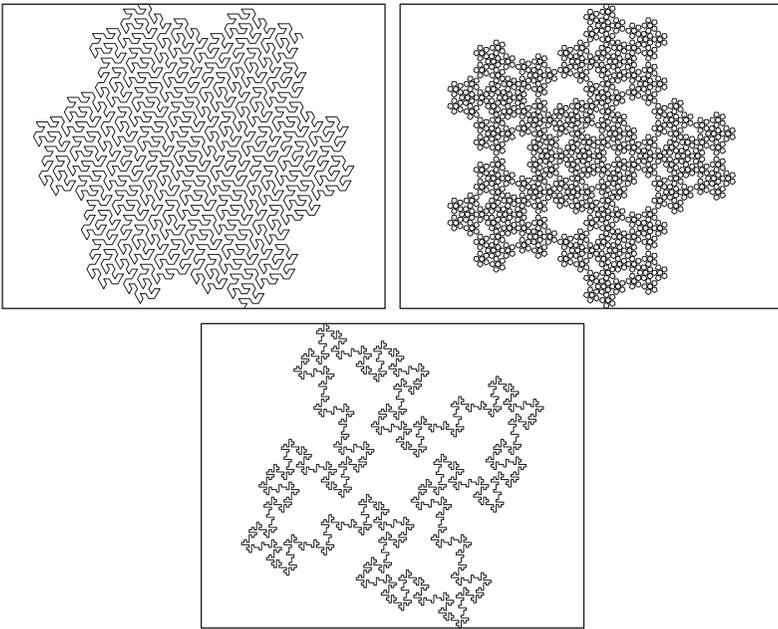


Figure 10.2 Graphical interpretation of the evolution of some well-known L-systems.

isolated music excerpts. In [Kaliakatsos-Papakostas et al. \(2012c\)](#), a study has been presented where some rhythmic attributes of the L-systems indicated extensive randomness in the produced music material. This led to an extension of the L-systems with the finite L-systems (FL-systems), which are able to produce rhythms with a wider spectrum of rhythmic characteristics, i.e., from more stable and predictable to more vague and complex rhythms.

10.3 Supervised Intelligent Composition

In contrast to the methodologies for unsupervised music composition, supervised methodologies aim to produce music that meets some predefined criteria. The systems that pertain to this category utilize intelligent algorithms in order to obtain the ability to compose music under some qualitative guidelines, which are often called “target features” or simply “features.” The underlying model that these systems utilize to produce music is either tuned or created from scratch with the utilization of intelligent algorithms toward the directions dictated by the features that the music output has to satisfy. An advantage of the supervised IMC systems is their ability to produce music with certain aesthetic and stylistic orientation ([Manaris et al., 2007](#)), in contrast to unsupervised composition systems. This fact, however, may arguably introduce a contraction to the novelty of the composed music, since the

aforementioned features are imposing restrictions to the composition process. Therefore, the formulation of these systems incorporates the following challenges: (a) create an *interpretation* of mathematical objects to music, (b) apply an *intelligent algorithm* to optimally traverse the search space of the mathematical objects, and (c) select a *proper set of features* that describe the desired music characteristics.

The selection of proper features is of vital importance for the supervised systems' performance. In an abstract sense, these features should provide landmarks for the system to compose music with certain characteristics, but at the same time allow it to introduce a considerable amount of novelty to the music it composes. The selection of proper features is thus crucial: on the one hand, they should capture the *essence* of the music to be composed; on the other hand, they should not *overdetermine* the compositions. Research on supervised intelligent composition has mainly been driven toward establishing effective intelligent methodologies that tune the composition models, restraining from the formulation of novel features that describe music in a global sense.

10.3.1 Supervised Composition with Genetic Algorithms

Automatic music composition is realized through the utilization of a “model” that is constructed by a composer—programmer. The aforementioned model may incorporate a set of parameters that define its compositional style and capabilities. It is thus crucial that a proper combination of these parameters is defined for the automatic creation of music, so that it may exhibit certain characteristics and aesthetic value of high quality. The utilization of genetic algorithms (GA) provides proper values for these parameters, given a qualitative measure of what the produced music should sound like. Thus, the problems that the composer—programmer faces are related not only to the formulation of a proper parametric model, but also, equally importantly, to the formalization of measurements that accurately describe the target music style.

The GA is a class of algorithms inspired by natural evolution, which iteratively produce better solutions to a problem. These algorithms belong to the wider class of “heuristics,” meaning that they initially “guess” a set of random solutions and produce new ones grouped in generations, by utilizing information only from their current generation and their product candidate solutions. Specifically, the candidate solutions within a generation in a GA scheme are evolved using operators resembling natural genetic phenomena (such as crossover and mutation) and a selection procedure that propagates the consecution of generations toward better solutions. A better solution means that the set of the model's parameters that this solution describes gives a more satisfactory result with regard to a qualitative measurement that is called “fitness function.” Among the most popular genetic operators are the crossover and mutation. Crossover incorporates the combination of the parameter between two “parent” solutions for the creation of two “children” solutions, while mutation is the random reassignment of certain parameter values of a “parent” solution to produce a new “children solution.” The progression from one generation of

“parent” solutions to the next is finally accomplished with a selection process that allows the “better” fitted solutions among parents and children to form the new generation.

The key notion in the GA is the *precise* and *informative* description of what a “better” solution is. In the case of supervised IMC, *precise* denotes the correct demarcation of the target musical attributes that the automatically composed music should encompass. The term *informative* expresses the necessity to model the target musical attributes in a “continuous” manner as possible, abstaining the hazard to create nonsmooth and discretized error surfaces that abound in local optima.

Among the first works for supervised composition using “objective” musical criteria for the assignment of fitness evaluations was the work of [Papadopoulos and Wiggins \(1998\)](#). In this work, a system was presented which was composing jazz solos over a given chord progression. The solutions to the problem were the melodies themselves, specifically pitch-duration pairs, and after a random initialization, GA with some special genetic operators with “musical meaning” was applied, fostering new generations of possible solutions-melodies. The evaluation process of each candidate solution-melody was based on eight evaluation indicators, borrowed by music theory. The results were reported to be promising; however, a thorough statistical examination of the results was not realized, according also to the authors’ opinion in the concluding section of their work. A similar system was introduced by [Özcan and Ercal \(2008\)](#), who also provide a downloadable application called *AMUSE*. In this work, a set of 10 music features were used for fitness evaluation. Furthermore, experimental results of a questionnaire-based research on a group of 20 participants indicated that these features are linked to human preference at some extent. The utilization of a fitness function based on music theoretic criteria was also utilized for the creation of four-part melodies from scratch ([Donnelly and Sheppard, 2011](#)).

On the other hand, features not related to music theory have also been utilized. These features are related to informatics and statistics, measuring aspects of melodic information capacity either through compressibility or through various characteristics of music that can be translated into discrete probabilities. In [Alfonseca et al. \(2007\)](#), the normalized compression distance (NCD) is used to allow a GA composes music pieces that resemble the pieces of a certain style. The sum of NCDs between each possible solution-melody and a set of target pieces in a certain style is computed, and a solution-melody with a better fitness is the one for which a smaller sum of NCDs is obtained. Systems that genetically evolve CA ([Lo, 2012](#)) and FL-systems ([Kaliakatsos-Papakostas et al., 2012e](#)) for music and rhythm composition have also been presented, where again fitness is calculated with the utilization of probability measures, such as n-Grams (Markov transition tables), Shannon information entropy (SIE), and compression rate.

10.3.2 Supervised Composition Genetic Programming

Genetic programming (GP) works under the same evolutionary principle with the GA that is evolving initially random possible solutions to new ones that are better

fitted to the problem at hand. The difference between GP and GA is the problem's formulation. In GA, the optimization process targets at the model's parameters, while the utilization of GP allows the optimization of the model *per se*, since the populations of possible solutions incorporate entire programs that actually form the model. These programs are constituted of short subprogram segments, hierarchically structured in a tree-like representation. The genetic operators are similar to that used by the GA; however, they act on tree branches instead of string-like or numeric chromosomes. The crossover operator, for example, exchanges subtrees of the parents' trees, creating children that comprise combined subprogram parts of their parents. Through a similar selection process as in the GA, new populations of possible solutions-programs are created which are better or equally fitted to the problem at hand.

Although GP offers an entirely new aspect regarding the problem formulation, the fitness evaluation of these programs remains an important issue. In [Spector and Alpern \(1994\)](#), a GP scheme which acted on an *initial* melody-string by performing several musical operations, such as retrograde and transposition, produced novel melodies. The fitness used in this work incorporated measurement on how different certain tonal and rhythmical characteristics are between the initial and the GP-created melodies. In an extension of this work ([Spector and Alpern, 1995](#)), an ANN has been presented for the automatic assignment of fitness based on the knowledge that this ANN has been trained on by several target pieces. Some works have also addressed the issue of fitness evaluation by training machine learning tools on features derived by a set of target pieces. For example, in [Manaris et al. \(2007\)](#), ANNs were trained on a set of features that were taken by sets of pieces belonging to a certain style. These features incorporated the fractal dimension (FD) of several music attributes, such as pitch, interval, and duration, and some of their local variability measurements (i.e., higher order derivatives and relative moving averages). The FD can be obtained by any music feature that may be expressed as a discrete probability density function (PDF) and reveals its information capacity by measuring the ratio of the successively ranked probabilities. Among others, self-organizing maps ([Phon-Amnuaisuk et al., 2007](#)) have also been trained on specified symbolic music features as automatic fitness raters.

10.4 Interactive Intelligent Composition

Interactive evolution (IE) is a general class of algorithms that transform the problem to formulate a proper fitness function, into the problem to allow the user to assign fitness evaluations based on her/his taste or actions. In this section, we propose the presentation of the IE algorithms in two subcategories: the *subjectively driven* and *performance driven* evolution systems. The *subjectively driven* systems evolve their parameters (or the system *per se*) based on the fitness evaluation provided directly by the user, either in a "select-the-best" fashion or by following a rating procedure. The underlying evolutionary mechanisms remain intact with an

exception on the fitness evaluation scheme that is performed by the user. In the *performance driven* systems, on the other hand, the interaction between the user and the system is not limited to the aforementioned rate-and-evolve scheme. Instead, the user feeds the system with her/his actions; the system analyzes these actions and produces a proper musical response. This response in turn, in most cases, affects the human's actions, creating an interactive "dialogue" between the human performer and the system itself.

The subjectively driven IE systems have the advantage to capture the human preference directly, obtaining feedback from the user's responses either in a user-selection fashion or in a user-rating fashion. However, such a fitness assignment scheme demands extensive effort by the user, who would ideally have to rate thousands of musical excerpts in order to drive the evolution to a compositional optimum. Additionally, as the hearing-selecting or rating trials evolve, the compositions "converge" to an optimum, thus obtaining great similarities. Therefore, the user is expected to be hearing and rating irritatingly similar music compositions. This, combined with the fact that a great number of iterative selection or rating turns are necessary, creates an effect that is common to IE called "*user fatigue*." User fatigue restrains the user from focusing on the material under evaluation, forcing her/him to provide inconsistent ratings, a fact that misleads evolution to suboptimal solutions.

The advantage of performance driven interactive systems is that the human and the automatic performer combine their compositional "virtues" in real time, creating music that is both structured and surprising. The human performer does not have direct control over the composition but has the ability to control several aspects of the produced music by transferring feedback to the system through her/his performance. The system on the other side acknowledges this feedback and responds by producing music which encompasses novelty and spontaneity, hopefully inspiring the human performer. Thus, this interactive dialogue integrates the characteristics of the human's structured understanding of music and the IMC's unpredictability. Although the rationale behind the formulation of performance driven systems is solid, their pragmatic realization has to deal with the following question: *how does the system understand the human's performance?* The implementation of such systems therefore faces the same problem as in [Section 10.3](#), which concerns the creation of proper *features* which capture the human's performance without being over- or under-descriptive. Additionally, the responses of these systems have to be prepared in real time, thus the underlying intelligent algorithms have to be as less time-consuming as possible.

It also has to be noted that in several works ([Millen \(2005\)](#), [Pestana \(2012\)](#), among others), the term "interactive" is used to describe the intervention of the human user to the composition process of the underlying system (CA and L-systems, respectively). This term does not meet the specifications that this work utilizes. Nevertheless, we do not argue that this characterization is wrong in any sense; however, it highlights our motivation that interaction is "intelligent" when it incorporates modification of the "intelligent mechanisms" of the system and not the parameter assignment. This fact has also been noted in a previous work

(Blackwell, 2007), by introducing the notion of “live algorithms” and describing them as follows: “The idea that interaction involves state change rather than parameter selection is an important aspect in the design of live algorithms.”

10.4.1 Composing with Swarms

Swarm intelligence (SI) is a branch of computational intelligence that discusses the *collective behavior* emerging within self-organizing societies of *agents*. SI was inspired by the observation of the collective behavior in societies in nature such as the movement of birds and fish. The collective behavior of such ecosystems, and their artificial counterpart of SI, is not encoded within the set of rules that determines the movement of each isolated agent, but it emerges through the interaction of multiple agents. Although several variations have been proposed in the literature, the fundamental set of rules used for music composition is based on the “*boids*” algorithm (Reynolds, 1987). These rules define the movement of each agent by adjusting its acceleration within short time intervals according to some conditions in the environment that it perceives. Specifically, these rules incorporate the following guidelines for the movement of each agent:

1. *Shoaling*: Move toward the center of mass of the agents that you perceive.
2. *Collision avoidance*: Move away from the agents that are too close to you.
3. *Schooling*: Move toward aligning your velocity to the mean velocity of the agents you perceive.

For these rules to be applied, some constants have to be predetermined, such as the radius of agent perception, i.e., at which distance does an agent perceive an object in its environment, and the radius of collision avoidance. These rules define the behavior of each agent, which begins forming organized groups called “swarms,” which present collective behavior (combined intelligence as if they were a single organism).

Several parts of the aforementioned social characteristics have been embodied to agents which produce music and sound output, in combination with human interaction. Using the terminology described earlier, these systems are *performance driven*, since the agents receive feedback from the human user which defines some characteristics either of their behavior or of their environment. A thorough review of various methodologies that compose music on the symbolic level can be found by Blackwell (2007). Performance driven interaction in these systems captures the human performance of an instrumentalist or a vocalist, creating “attractors” in the space that the agents move. The agents are gathered around these attractors forming clusters, several spatial characteristics of which are translated into MIDI music information (including pitch, duration, intensity, polyphony, and several motif aspects).

In a similar fashion, computer and human interaction on the sound level has been accomplished with the utilization of swarms (Blackwell, 2008). The human performer’s audio input is analyzed and attractors are generated in the space where the swarms move, which attract the swarms. Granulation parameters

(pitch, amplitude, duration, gain distance, attack, and decay) are then defined by the position of the agents within this space. An extension of this work with the location of agents determining the spatial localization of sound has been presented by Wilson (2008). An alternative approach, where the human user is not an instrumentalist or a vocalist, was presented by Jones (2008). In the latter work, the user has the ability to create, destroy agents, and deposit food on the agents' space, creating a simplified version of interaction, since the user does not necessarily need to be specialized in music performance.

10.4.2 Interactive Composition with GA and GP

Among the first works for *subjectively driven* IE was a simple rhythm rating scheme using GA evolution according to fitness provided by user ratings (Horowitz, 1994). In a similar fashion (Johanson and Poli, 1998), a system was presented where initially random note sequences were rearranged with the utilization of music function combinations (such as play mirror, play twice, and shift down). Driven by subjective user ratings, the combination of these music functions evolved, providing better combinations. Additionally, an ANN was trained on the positive responses of the user, which was subsequently used as an automatic fitness rater. This allowed some individuals to be discarded before the user had a chance to hear them, thus reducing the effect of user fatigue caused by hearing a great number of successive low-quality melodies. The former work was extended with a combination of GA and GP in the music composition mechanism by Tokui (2000). In this work, short individual melodies were considered as chromosomes and were evolved using GA, while in parallel, functions that determined their temporal arrangement were evolved using GP. Again, user rating was fed into an ANN so that low-rated individuals could be acknowledged and discarded before the user had a chance to hear them. More recent approaches incorporate the evolution of several musical aspects (such as rhythm, tonality, and style) based on separate human ratings for each aspect (Fortier and Van Dyne, 2011; Moroni et al., 2000).

A GP scheme was utilized for subjectively driven evolution of sounds with sound synthesis techniques by Putnam (1994). In this approach, waveforms were directly shaped by functions, which in turn were evolved according to fitness values provided by users. However, using the aforementioned class of functions was mentioned to “*produced little more than irritating noise and evolved (if at all) very slowly*” (Putnam, 1994). In Kaliakatsos-Papakostas et al. (2012b), a GP scheme was used that evolves a class of functions which create waveforms with pleasant and interesting sonic output (Heikkilä, 2011). This subjectively driven scheme successfully evolved sounds-melodies to more preferred ones for each user. This fact allowed to draw some conclusions about possible aesthetic measures for sound, by measuring their difference between unevolved and evolved sounds-melodies. This system has also been utilized to examine the application of depth-adaptive fitness indicators by Kaliakatsos-Papakostas et al. (2012a). This subjectively driven system was observed to converge more quickly to higher rated individuals for the users that utilized a modified version of the standard

GP crossover and mutation operators. This version incorporated a constraint to the depth of the tree representation that these operators acted, according to the fitness assigned by the user (lower fitness allowing action to lower height, thus inflicting greater variability).

The pioneering work of [Biles \(2002\)](#) yielded a hybrid *performance driven* system, called *GenJam*, which utilizes GA to generate musical responses to a human improviser, while being rated for these responses. GenJam receives the music phrases that a human improviser is playing, considers them as chromosomes, and genetically evolves them to generate novel music responses. These responses are then rated by the improviser and subsequently evolved toward better melodies. In an extension of this work, GenJam utilizes an initial database of jazz phrases, a combination of which fosters new generations of phrases after the application of the “crossover” operator ([Biles, 2001](#)). In this way, the “fitness bottleneck” is eliminated and the GenJam is allowed to improvise autonomously, without the constant improviser’s ratings. In a recent work ([Weinberg et al., 2008](#)) (similar to autonomous GenJam), the initial database of phrases was recorded by a jazz piano improviser and then stored to the system’s memory. These phrases were blended with the input phrases from the human improviser using GA, forming a music response that included a satisfactory combination of novelty and relativity to the human improviser’s phrases. The responses of this system was provided by a two-armed robot device performing on a xylophone, in contrast to most other approaches which use synthetic computer-generated sounds.

Different approaches to population initialization and fitness estimation have recently been suggested for *performance driven* system. In [Manaris et al. \(2011\)](#), Monterey Mirror is presented, which receives human phrases as input and creates music responses from scratch, using GA and a probabilistic population initialization scheme. The initial population is created by utilizing the *Markov transition probability* tables, which derive from the pitch and rhythm information extracted from the phrases provided by the human performer. This population is then evolved using as target, the FD of several features in the human improviser’s phrases.

An alternative approach for performance driven interactive composition was followed by [Kaliakatsos-Papakostas et al. \(2012d\)](#), where the user input was analyzed in terms of tonality (chord estimation), Shannon information entropy (SIE) of the pitch class profile (PCP), rhythm characteristics (using some rhythmic features describing note density, syncopation, etc.), and intensity variations. These characteristics were then used as fitness evaluators for three different submodules:

1. The *tone generation submodule* utilizes the differential evolution (DE) algorithm (DE is a continuous variation of the GA with operators that take advantage of possible solution differences) to define the parameters of a chaotic function, so that it may produce responses with similar PCP SIE to the one of the phrases the improviser is playing.
2. The *rhythm submodule* utilizes the genetic evolution of the FL-systems described in [Section 10.3.1](#), based on the rhythm that the improviser is playing within a sliding time window of fixed length.

3. The *intensity submodule* recognizes the mean value and standard deviation of the improviser's intensities and responds with values for intensity drawn by a uniform random number generator with the aforementioned mean and standard deviation.

This system provides responses with multiple “intelligent instrumentalists,” which “hear” and “tune-in” to the human improviser's playing style, forming an intelligent band that performs constraint-free improvisation.

10.5 Conclusions

This chapter has presented a short review of a wide range of methodologies that may be considered within the generalized notion of *IMC*. In parallel with a short discussion about what would the term “intelligent” denotes when speaking about automatic music composition, a categorization of *IMC* methodologies was proposed that grouped methodologies not according to the underlying computational mechanisms, but according to the utilization of the *intelligent part* of the methodologies with regard to the *compositional aims*. The intelligent parts incorporated bioinspired methods, which either produce life-like structures or use notions borrowed by nature (such as evolution and collective behavior) to create human-like compositions. The compositional aims are defined by the composer/programmer and roughly pinpoint the rate of compositional freedom allowed to the underlying mechanism, in combination with the interactivity of the system's responses.

The intelligent part of the compositional methodologies would be composing music undestructively, without the imposition of restrictions or targets, forming the category of *unsupervised intelligent composition* methods. The imposition of compositional restrictions with the introduction of target features that derive from stylistic directions forms the category of *supervised intelligent composition*. In this category, the intelligent part of the methodologies drives the compositional process toward creating music with specified characteristics, which derive from sets of target pieces or desired music characteristics. Finally the category of *interactive intelligent composition* was presented, which incorporates methodologies that compose music based on the input provided by the human user. This category was further subdivided in the subcategories of *subjectively driven* and *performance driven* methodologies. The former drives the evolution solely according to the user's choices or ratings, whereas the latter decodes the human input into musical information and creates proper music responses.

More methodologies that pertain to the aforementioned categorization can be found within an overwhelming amount of excellent works that were not mentioned in this chapter. The authors hope that the reader was given a representative sample from the state-of-the-art systems among the most popular intelligent methodologies. Among the authors' motivations for this chapter is also the establishment of a commonly accepted terminology and categorization of intelligent music methodologies, based on the force that drives their creativity, whether it is nature-like complexity,

target-oriented evolution, or human interaction. Such a categorization could be of vital importance to the intelligent music research community and would benefit the researcher, the musician, and the student. The researcher would have immediate access to works that are relative to her/his research. The musician would be facilitated by quickly filtering out the works that are not among her/his artistic scope, without frustratingly going through a great amount of irrelevant material. Finally, the student could constructively understand the motivation behind each intelligent methodology and the mechanisms of the underlying intelligent methods, making the way to expertise less time-consuming and more creative.

References

- Alfonseca, M., Cebrian, M., Ortega, A., 2007. A simple genetic algorithm for music generation by means of algorithmic information theory. In: IEEE Congress on Evolutionary Computation, CEC'07, pp. 3035–3042.
- Biles, J.A., 2001. Autonomous GenJam: eliminating the fitness bottleneck by eliminating fitness. In: Proceedings of the 2001 Genetic and Evolutionary Computation Conference Workshop Program, GECCO'01, Morgan Kaufmann, San Francisco, CA.
- Biles, J.A., 2002. GenJam: evolution of a jazz improviser. In: Bentley, P.J., Corne, D.W. (Eds.), *Creative Evolutionary Systems*. Morgan Kaufmann Publishers Inc., San Francisco, CA, pp. 165–187.
- Blackwell, T., 2008. Swarm granulation. In: Romero, J., Machado, P. (Eds.), *The Art of Artificial Evolution*, Natural Computing Series. Springer, Berlin/Heidelberg, pp. 103–122.
- Blackwell, T.M., 2007. Swarming and music. In: Miranda, E.R., Biles, J.A. (Eds.), *Evolutionary Computer Music*. Springer, London, pp. 194–217.
- Burraston, D., Edmonds, E., 2005. Cellular automata in generative electronic music and sonic art: a historical and technical review. *Digit. Creativity*. 16 (3), 165–185.
- Donnelly, P., Sheppard, J., 2011. Evolving four-part harmony using genetic algorithms. Proceedings of the 2011 International Conference on Applications of Evolutionary Computation—Volume Part II, *EvoApplications'11*. Springer-Verlag, Berlin/Heidelberg, pp. 273–282.
- Fortier, N., Van Dyne, M., 2011. A genetic algorithm approach to improve automated music composition. *Int. J. Comput.* 5 (4), 525–532.
- Geis, M., Middendorf, M., 2008. Creating melodies and baroque harmonies with ant colony optimization. *Int. J. Intell. Comput. Cybern.* 1 (2), 213–238.
- Gogins, M., 2010. Score generating Lindenmayer Systems in the Generalized Contextual Group. Available Online: <http://michaelgogins.com/pdf/Lindenmayer_Systems_Based_on_Riemannian_Transformations.pdf> (accessed 10.15.12).
- Griffith, N., Todd, P.M. (Eds.), 1999. *Musical Networks: Parallel Distributed Perception and Performance*. MIT Press, Cambridge, MA.
- Heikkilä, V., 2011. Discovering novel computer music techniques by exploring the space of short computer programs. CoRR abs/1112.1368 (accessed 10.15.12).
- Horowitz, D., 1994. Generating rhythms with genetic algorithms. Proceedings of the Twelfth National Conference on Artificial Intelligence (vol. 2), AAAI'94. American Association for Artificial Intelligence, Menlo Park, CA, pp. 1459–1460.

- Johanson, B.E., Poli, R., 1998. GP-Music: an interactive genetic programming system for music generation with automated fitness raters. Technical Report CSRP-98-13, University of Birmingham, School of Computer Science.
- Jones, D., 2008. AtomSwarm: a framework for swarm improvisation. *Proceedings of the 2008 Conference on Applications of Evolutionary Computing, Evo'08*. Springer-Verlag, Berlin/Heidelberg, pp. 423–432.
- Kaliakatsos-Papakostas, M.A., Eptropakis, M.G., Floros, A., Vrahatis, M.N., 2012a. Controlling interactive evolution of 8-bit melodies with genetic programming. *Soft Comput.* 16 (2), 1997–2008.
- Kaliakatsos-Papakostas, M.A., Eptropakis, M.G., Floros, A., Vrahatis, M.N., 2012b. Interactive evolution of 8-bit melodies with genetic programming towards finding aesthetic measures for sound. In: *Proceedings of the First International Conference on Evolutionary and Biologically Inspired Music, Sound, Art and Design, EvoMUSART'12*, Malaga, Spain, LNCS, vol. 7247. Springer-Verlag, pp. 140–151.
- Kaliakatsos-Papakostas, M.A., Floros, A., Vrahatis, M.N., 2012c. Intelligent generation of rhythmic sequences using finite L-systems. In: *Proceedings of the Eighth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, IHHMSP'12*, Piraeus, Athens, Greece, pp. 424–427.
- Kaliakatsos-Papakostas, M.A., Floros, A., Vrahatis, M.N., 2012d. Intelligent real-time music accompaniment for constraint-free improvisation. In: *Proceedings of the 24th IEEE International Conference on Tools with Artificial Intelligence, ICTAI'12*, Piraeus, Athens, Greece.
- Kaliakatsos-Papakostas, M.A., Floros, A., Vrahatis, M.N., Kanellopoulos, N., 2012e. Genetic evolution of L and FL-systems for the production of rhythmic sequences. In: *Proceedings of the Second Workshop in Evolutionary Music Held during the 21st International Conference on Genetic Algorithms and the 17th Annual Genetic Programming Conference (GP), GECCO'12*, Philadelphia, PA, pp. 461–468.
- Lo, M.Y., 2012. Evolving cellular automata for music composition with trainable fitness functions. Ph.D. Thesis, University of Essex, Essex.
- Lourenc, B.F., Brand, C.P., 2009. L-systems, scores, and evolutionary techniques. In: *Proceedings of the SMC 2009—Sixth Sound and Music Computing Conference*, pp. 113–118.
- Manaris, B., Roos, P., Machado, P., Krehbiel, D., Pellicoro, L., Romero, J., 2007. A corpus-based hybrid approach to music analysis and composition. *Proceedings of the 22nd National Conference on Artificial Intelligence—Volume 1*. AAAI Press, Vancouver, British Columbia, pp. 839–845.
- Manaris, B., Hughes, D., Vassilandonakis, Y., 2011. Monterey mirror: combining Markov models, genetic algorithms, and power laws. In: *Workshop in Evolutionary Music, IEEE Congress on Evolutionary Computation, CEC'11*, New Orleans, LA.
- McCormack, J., 1996. Grammar-based music composition. *Complexity Int.* 3.
- Millen, D., 2005. An interactive cellular automata music application in cocoa. In: *Proceedings of the 2004 International Computer Music Conference, ICMC'04*.
- Miranda, E.R., Biles, J.A., 2007. *Evolutionary Computer Music*. Springer-Verlag New York, Inc., Secaucus, NJ.
- Moroni, A., Manzolli, J., Zuben, F.V., Gudwin, R., 2000. Vox populi: an interactive evolutionary system for algorithmic music composition. *Leonardo Music J.* 10 (1), 49–54.
- Özcan, E., Ercal, T., 2008. A genetic algorithm for generating improvised music. In: *Monmarché, N., Talbi, E.-G., Collet, P., Schoenauer, M., Lutton, E. (Eds.), Artificial Evolution, Lecture Notes in Computer Science*, vol. 4926. Springer, Berlin/Heidelberg, pp. 266–277.

- Papadopoulos, G., Wiggins, G., 1998. A genetic algorithm for the generation of jazz melodies. In: Proceedings of the Finnish Conference on Artificial Intelligence (STeP), pp. 7–9.
- Pestana, P., 2012. Lindenmayer systems and the harmony of fractals. *Chaotic Model. Simul.* 1 (1), 91–99.
- Phon-Amnuaisuk, S., Law, E., Kuan, H., 2007. Evolving music generation with SOMFitness genetic programming. In: Giacobini, M. (Ed.), *Applications of Evolutionary Computing, Lecture Notes in Computer Science*, vol. 4448. Springer, Berlin/Heidelberg, pp. 557–566.
- Prusinkiewicz, P., 1986. Score generation with L-systems. In: Proceedings of the International Computer Music Conference, ICMA'86, pp. 455–457.
- Prusinkiewicz, P., Lindenmayer, A., 1990. *The Algorithmic Beauty of Plants*. Springer-Verlag New York, Inc., New York, NY.
- Putnam, J.B., 1994. Genetic Programming of Music. Available Online: <www0.cs.ucl.ac.uk/staff/ucacbb/ftp/papers/ep.ps.gz> (accessed 10.15.12).
- Reynolds, C.W., 1987. Flocks, herds and schools: a distributed behavioral model. In: Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH'87, pp. 25–34.
- Solomos, M., 2005. Cellular automata in Xenakis' music. Theory and practice. In: Proceedings of the International Symposium Iannis Xenakis, pp. 120–137.
- Spector, L., Alpern, A., 1994. Criticism, culture, and the automatic generation of artworks. Proceedings of the Twelfth National Conference on Artificial Intelligence (vol. 1), AAAI'94. American Association for Artificial Intelligence, Menlo Park, CA, pp. 3–8.
- Spector, L., Alpern, A., 1995. Induction and recapitulation of deep musical structure. In: Proceedings of the IFCAI'95 Workshop on Artificial Intelligence and Music, pp. 41–48.
- Tokui, N., 2000. Music composition with interactive evolutionary computation. *Communication.* 17 (2), 215–226.
- Weinberg, G., Godfrey, M., Rae, A., Rhoads, J., 2008. A real-time genetic algorithm in human–robot musical improvisation. *Computer Music Modeling and Retrieval Sense of Sounds*. Springer-Verlag, Berlin/Heidelberg, pp. 351–359.
- Wilson, S., 2008. Spatial swarm granulation. In: Proceedings of the 2008 International Computer Music Conference.
- Wolfram, S., 2002. *A New Kind of Science*. first ed. Wolfram Media, Champaign, IL.