2008

AN APPROACH TO MACHINE DEVELOPMENT OF MUSICAL ONTOGENY

GIMENES, MARCELO

http://hdl.handle.net/10026.1/1812

http://dx.doi.org/10.24382/3821
University of Plymouth

AN APPROACH TO MACHINE DEVELOPMENT OF MUSICAL ONTOGENY

by

MARCELO GIMENES

A thesis submitted to the University of Plymouth
in partial fulfilment for the degree of

DOCTOR OF PHILOSOPHY

School of Computing, Communications and Electronics
Faculty of Technology

November 2008

Copyright statement

An Approach to Machine Development of Musical Ontogeny

Marcelo Gimenes

Abstract

This Thesis pursues three main objectives: (i) to use computational modelling to explore how music is perceived, cognitively processed and created by human beings; (ii) to explore interactive musical systems as a method to model and achieve the transmission of musical influence in artificial worlds and between humans and machines; and (iii) to experiment with artificial and alternative developmental musical routes in order to observe the evolution of musical styles.

In order to achieve these objectives, this Thesis introduces a new paradigm for the design of computer interactive musical systems called the Ontomemetical Model of Music Evolution - OMME, which includes the fields of musical ontogenesis and memetics. OMME-based systems are designed to artificially explore the evolution of music centred on human perceptive and cognitive faculties.

The potential of the OMME is illustrated with two interactive musical systems, the Rhythmic Meme Generator (RGeme) and the Interactive Musical Environments (iMe), which have been tested in a series of laboratory experiments and live performances. The introduction to the OMME is preceded by an extensive and critical overview of the state of the art computer models that explore musical creativity and interactivity, in addition to a systematic

exposition of the major issues involved in the design and implementation of these systems.

This Thesis also proposes innovative solutions for (i) the representation of musical streams based on perceptive features, (ii) music segmentation, (iii) a memory-based music model, (iv) the measure of distance between musical styles, and (v) an improvisation-based creative model.

Contents Overview

Table of Contents

List of Illustrations

List of Tables

Acknowledgments

I would like to express my gratitude to

The Brazilian Government's Fundação Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) for granting the scholarship that allowed me to pursue my PhD.

My supervisors, Prof. Eduardo Reck Miranda and Dr Chris Johnson, for their continuous guidance and advice.

Prof. Jônatas Manzolli, with whom I had the opportunity to have some of the initial discussions that led to this Thesis.

All the colleagues I met at the Interdisciplinary Centre for Computer Music Research, in special Adolfo Maia Jr., Alexis Kirke, Andrew Brouse, Eduardo Coutinho, Jaime Serquera, Joao Martins, Leandro Costalonga, Martijn Bosma, Nigel Morgan, Patricio da Silva, Peter Beyls, Qijun Zhang, and Vadim Tikhanoff, for the helpful discussions, friendship, and encouragement.

Scott McLaughlin and Janet Price for proof-reading.

The external and internal examiners, for taking their precious time to assess this Thesis.

My parents, Benjamin and Josephina, for their love and whole-hearted support.

My wife, Fernanda, for her love, endless patience and encouragement.

My children, Vitoria, Isadora and Pedro, my greatest source of inspiration.

Author's Declaration

The research introduced in this Thesis is not the subject nor is it part of any other University degree pursued by the Author.

This study was financed with the aid of a studentship from the Brazilian Government's Fundação Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES).

A record of activities can be found in Appendix 2 (p. 239).

Word count: 59104

Plymouth, 5 December 2008

Introduction

Music students are frequently encouraged to listen to celebrated musicians in order to improve their knowledge of different aspects of music making. Aspirant performers, for instance, concentrate on how specific elements could be (re)interpreted according to variations of dynamics and tempo, how different musical ideas and structures are developed and chained and how they are used to convey particular musical intentions. As a matter of fact, just slight changes of these elements have the potential to entirely transform the meaning of music, something that reveals at the same time the beauty and the complexity of music communication.

From my own education I recall many moments in which the discovery of music and particularly of music performance were a source of enlightenment, even though sometimes these moments resulted in frustration given my inability to achieve the results I was looking for. On one of these occasions, while I was receiving formal instruction on jazz piano improvisation in Rio de Janeiro, I was concentrating on listening to several recordings by Bill Evans. This American pianist developed a comprehensive career and is considered to be one of the most influential musicians in jazz history. Some of the things that attracted me the most in Evans' style were the freshness of his harmonies and the lyricism of his melodies. During one of the improvisation classes, my tutor noticed that I was starting to develop some of Bill Evans' mannerisms. At that point I was not consciously committed to developing my own improvisational style. My task was, to a large extent, to listen to as many different musicians as possible in order to improve my own improvisational skills. However, it occurred to me that I was "sounding" like Bill Evans only because I was listening so much to his music.

Almost instantly a series of questions began to come to my mind: what if, instead of Evans, I had been listening to somebody else, maybe someone with a completely different style? Would my improvisation have been influenced as much as it had been by Evans' music? What sort of perceptive and mental mechanisms allowed this sort of things to happen?

Eventually, I decided to dig deeper into this subject and went to the State University of Campinas (Brazil) where I pursued my Master's degree (Gimenes, 2003b). The subject was precisely Bill Evans' vertical structures (harmonies and "voicings"): I chose some of his well-known transcribed performances[1] and, aided by computer applications, mostly spreadsheets, I registered, categorised and classified all the occurring vertical structures (Gimenes, 2003a). Figure 1 shows a small part of the findings, the left hand vertical structures with three notes that Bill Evans predominantly used. In this figure, numbers represent the frequency of each vertical structure compared to the totality of the sample.

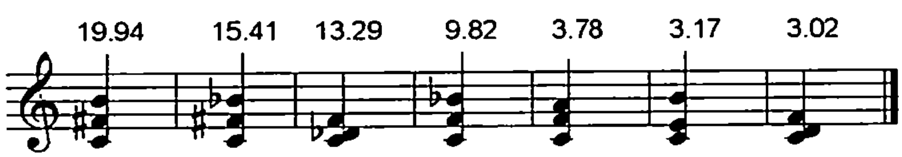


Figure 1: Bill Evans' left hand vertical structures.
(numbers = % of the sample)

In that moment I perceived that, by using a systematic method that basically included statistical analysis, it had been possible to describe some aspects of a

---

[1] Autumn Leaves, Peri's Scope, Waltz For Debby, Here's That Rainy Day, You Must Believe In Spring, Up With The Lark (recorded respectively in 18/12/1959, 28/12/1959, 25/6/1961, 30/9/1968, 23/8/1977, and 6/6/1980)

musician's style. There was something that Evans had used so consistently and repetitively that could ultimately be regarded as his own distinguishing mark. But, how did he achieve a point in his career where other people would listen to him and recognize his style? What mechanisms govern the way human beings interact with each other and absorb these traits? How are musical elements transmitted from one mind to another?

Something different also came to my mind: looking more globally, if it were possible to enumerate all the elements that describe today's state of cultural evolution and how humans perceive and transmit these elements, it would be possible to predict how culture would be in the future. Nonetheless, this task would certainly involve an extraordinarily large and unmanageable number of subjects, from Psychology to Physiology and Education, instrument design to music representation, just to name a few. Given these circumstances, "cultural prediction" is something not achievable today; maybe a naive idea, but still an inspirational thought.

An appropriate context to explore these questions is found in the Cognitive Sciences. In the words of Gozalez *et al* (2001), Cognitive Sciences are:

> "An area of interdisciplinary research on the nature and structure of cognitive processes" that "brings together various fields of knowledge, including philosophy, psychology, neuroscience, computer, language and robotics. Two central features differentiate it from other areas of cognitive research: its interdisciplinary angle, unified by philosophical thought, and the techniques for computer simulation employed in the preparation of explanatory models of the cognitive activity" (Gonzalez *et al.*, 2001).

Music Cognition is one of the many facets explored by the Cognitive Sciences and concentrates on the understanding of phenomena that can be described as the habits of the musical mind. Research in this field has been taking wide steps during the last decades, especially in domains related to computer synthesis of musical style, classification and recognition of structures and statistical analysis of patterns, among many others. A relevant topic for Music Cognition is precisely to identify the way musical intelligence evolves during creative processes.

Research objectives

In light of the above, this Thesis proposes the use computational modelling to focus on the role that the transmission of musical elements plays in the emergence and development of musical styles. The aim is to demonstrate that it is possible to artificially replicate these natural phenomena with specially designed interactive musical systems. Three main objectives are put forward:

(i) To use computational modelling to explore how music is perceived, cognitively processed and created by human beings;

(ii) To explore interactive musical systems as a method to model and achieve the transmission of musical influence in artificial worlds and between humans and machines; and

(iii) To experiment with artificial and alternative developmental musical routes in order to observe the evolution of musical styles

Each one of these subjects is surrounded by a vast amount of other topics. In this Thesis they will be encapsulated in a new model for the design of computer interactive musical systems called the "Ontomemetical Model of Music Evolution" (OMME), centred on the idea that music is essentially a human activity. In addition, this model represents a significant step towards the development of computer systems that explore the concept of machine musicianship and the communication between humans and machines.

General plan of the Thesis

This Thesis is divided into three parts. The first two, "Part I: The Field of Research" and "Part II: Interactive Musical Systems", gradually builds a critical knowledge on which "Part III: Ontomemetical Systems" - the core and main outcome of my research - is founded.

"Part I: The Field of Research" contains the following chapters:

> "Chapter 1 Music" introduces the viewpoint from which the main subject of this Thesis will be addressed: music as a human intelligent ability, culturally conditioned, and an act of communication.
>
> "Chapter 2 Human Faculties" explores the main perceptive and cognitive faculties that enable human beings to appreciate and communicate through music.
>
> "Chapter 3 Computer Models" reviews the state of the art computer models that investigate musical creativity.

"Part II: Interactive Musical Systems" contains the following chapters:

> "Chapter 4 Interactivity" explores the concept of interactivity, and extensively reviews a number of so-called interactive musical systems.
>
> "Chapter 5 Interactive Systems Design" is a systematic exposition of the major issues involved in the design and implementation of interactive musical systems and explains how many researchers have tackled them.

Finally, "Part III: Ontomemetical Systems" contains the following chapters

> "Chapter 6 The Ontomemetical Model" introduces the Ontomemetical Model of Music Evolution (OMME) aiming at the exploration of music evolution centred on human perceptive and cognitive faculties.

"Chapter 7 RGeme" introduces the Rhythmic Meme Generator (RGeme), the first implementation of a multi-agent interactive musical system based on the OMME specification.

"Chapter 8 iMe" introduces the second implementation of an OMME-based multi-agent musical system, the Interactive Musical Environments (iMe), especially designed to support interactivity from an improvisational point of view.

"Chapter 9 Conclusion" highlights the contributions to knowledge introduced in the Thesis and make recommendations for future advancements in the field.

Part I: The Field of Research

Chapter 1 Music

> "Music is one of the most intriguing phenomena of the human kind." ((Storr, 1993) in (Miranda, 1999, p. 4))

Music can be regarded from many different viewpoints and in many different contexts. Citing Storr (1993), Miranda (1999, pp. 4-5) mentions that our "sensitivity to timing and imitation, our tendency for imposing order on auditory information, our ability to categorise sound, our tendency to recognise and imitate sound patterning, and so on, are all unique to humans". Music is an intelligent activity and ubiquitously human as "there is no known human society which does not exhibit musical behaviour in some form" (Wiggins, 2006, p. 450).

Music is "the art or science of combining vocal or instrumental sounds (or both) to produce beauty of form, harmony, and expression of emotion" (Anon, 2005b). "Beauty of form", "harmony" and "expression of emotion" are concepts that many people may not find relevant, especially under contemporary aesthetical values. In the definition of music, other elements such as sound and organisation are however essential.

In this Chapter, three aspects of music will be addressed, namely, structure, style and communication, which are intrinsically related in mechanisms that allow the emergence and development of music. These elements will be consistently studied throughout the whole Thesis and will become referential ingredients of the model that will be proposed in Chapter 6 - The Ontomemetical Model (p. 125).

1.1 Structure

In physical terms, sounds are audible vibrations or oscillations and therefore, phenomena essentially dependent on the time domain. It is widely accepted, however, that not all sounds should be considered music but only those that are combined or organized. Music is "a pattern of sounds in time" (Blackwell, 2006, p. 1) and structure is precisely "what distinguishes music from noise" (Wiggins, 1999, p. 2).

Structures arise from the way we perceive or create things because, as humans, we are naturally inclined to apply structure to everything in the world, even when there is none. Experiments show, for instance, that people perceive a series of computationally generated identical pitches as groups of two or three elements (Handel, 1989; Wiggins, 1999).

Another expression that refers to the same idea is pattern. Whitehead (Andrews, 1987, p. 19) once declared that "art is the imposing of a pattern on experience, and our aesthetic enjoyment is recognition of the pattern". Patterning implies the sensitive perception of the world and its categorisation into forms and classes of forms through cognitive activity, "the mental action or process of acquiring knowledge and understanding through thought, experience and the senses" (Anon, 2005a).

Musicologists have addressed the issue of musical structures in a number of different ways, starting from simple elements such as notes and chords and moving to higher-level structural concepts that define the overall form of a musical piece. In general, scholars look at "semantic units" (Boroda, 1992), which embody "a kind of musical molecule consisting of a number of integrated musical events, possessing a certain completeness, and well adapted to combination with other similar units" ((Schoenberg, 1983, pp. 3-8) in (Boroda, 1992)).

This Thesis will address (p. 31) the idea of musical structures taking into consideration exactly what allows the understanding of organized sounds and thus of music, the human perceptive and cognitive faculties. This issue will then be incorporated in the computational model proposed in Chapter 6 - The Ontomemetical Model (p. 125).

1.2 Style

In general terms, style is the expression of an identity or of a personal worldview and we associate it with fashion, and music, among many other things. The central idea is that the frequent occurrence of some traits or features produces a number of "fingerprints" through which someone or something becomes recognizable. In music, styles result from the recurrent use of many different elements, also known as "signatures" (Cope, 1991). Again, it is not the occasional but the recurrent use of these elements that makes them identifiable.

Rowe explains that styles are a "network of patterns, higher-level structures, and melodic, rhythmic, and timbral conventions that characterize identifiable musical genres" (Rowe, 1996, p. 58). "When listening to a piece of music for the first time, one usually can detect previously heard patterns even though the music is generally new to the ears (Cope, 1991, p. 24). In fact, "stylistic mimicry has been Europe's most enduring method for instructing young composers as well as writers, painters, and other artists" (Muscutt, 2007, p. 12).

For Meyer (1961, p. 45), musical styles are "more or less complex systems of sound relationships understood and used in common by a group of individuals". In a well-known definition, he adds that styles are "a replication of patterning, whether in human behaviour or in the artefacts produced by human behaviour, that results from a series of choices made within some set of constraints" (Meyer, 1989, p. 3).

Meyer understands that musical styles could be organized as a system of families: "The relationships between different styles can be characterized in a manner akin to the genealogy of languages. He refers to "style-system families" - groups of related styles. For example, Meyer notes that Bach and Beethoven represent different styles within a single style system or style family, whereas Mozart and Machaut belong to different style systems" (Huron, 2008). Jan (2000) classifies musical styles into five hierarchical levels, based on Nattiez (1990) and Meyer's (1989) ideas about style organization:

1. Intra-opus refers to patterns within a single work,
2. Idioms consist of the whole of composers' intra-opus contexts,
3. Dialects are the style of a community defined geographically and/or chronologically,
4. Rules refer to organizations in large-scales such as the modal system, and
5. Laws are invariable attributes belonging to human perception and cognition.

A fundamental element that arises from Meyer's (1961) definition on musical style is the fact that many sorts of constraints - that are ultimately determined by the environment in which the composer dwells - condition the emergence and disappearance of musical styles. Among these constraints, personal background and technology are essential to the understanding of styles. Addressing the development of his own musical style, Bill Evans once declared that he never really struggled for identity: "That's something that just has happened automatically as a result, I think, of just putting things together, tearing things apart and putting it together my own way, and somehow I guess the individual comes through eventually" (Stevens, 2008).

The model proposed in this Thesis (Chapter 6 - The Ontomemetical Model) is specifically concerned with these issues, particularly with the emergence and development of musical styles, and will define a number of conditions that will allow that interactive musical systems be used as tools for the manipulation and observation of these phenomena in artificial environments.

1.3 Communication

Being at the same time organised sound and a human activity, another fundamental aspect of music is that it has an intrinsic communicative role. As observed by Evans:

> "All people are in possession of what might be called a universal musical mind. Any true music speaks with this universal mind to the universal mind in all people. The understanding that results will vary only in so far as people have or have not been conditioned to the various styles of music in which the universal mind speaks. Consequently, often some effort and

exposure is necessary in order to understand some of the music coming from a different period or a different culture than that to which the listener has been conditioned" (Transcribed from Evans, 2005).

It follows that the success of the communication - or the understanding of the message - must be conditioned to one's previous knowledge. If someone is not familiar with a specific style, some "effort and exposure" is often necessary in order to enhance the chances that the communication will be successful.

However, even though music is an act of communication, it is not obvious what it communicates, the subject matter, or the semantics of the message. In fact, the understanding of the musical message would barely benefit from comparisons with verbal communication, as they possess different natures. Music can be (and is often) ambiguous as the composer, performer and/or listener can have different perceptions regarding particular musical structures (Wiggins, 1998).

Meyer (1961) understands that there would be three types of musical meaning: (i) hypothetical, which arises after a given stimulus, in expectation to something, (ii) evident, which connects antecedent and consequent stimuli, and (iii) determinate, which is unrelated to time (Titchener *et al.*, 1973). In addition, it is normally incontrovertible that emotion or affect is at the heart of the musical message. In fact, affective responses to music are physiologically measurable and have been reported by many researchers (Koelsch *et al.*, 2005; Luck, 2008).

Musical communication leads to musical influence, "the capacity to have an effect on the character, development or behaviour of someone or something, or the effect itself" (Anon, 2005a). An interesting analytical perspective is to identify connections amongst different styles according to the possible influences composers could have had over each other along their career.

Assayag refers to the process of musical influence as "stylistic reinjection" (Assayag *et al.*, 2006, p. 2): "The idea behind stylistic reinjection is to reify, using the computer as an external memory, this process of reinjecting musical figures from the past in a recombined fashion, providing an always similar but always innovative reconstruction of the past". In a similar context, Pachet uses the expression reflexive interaction: "The idea here is to show how the user can progressively feed the system with his own music material ... and get, in real time, an exploration of the accumulated material" (Pachet, 2006a, p. 8).

Analysis of musical influences, although inspiring, is not an easy task because there are many and complex factors that must be taken into account (e.g., human perceptive and cognitive issues, multidimensionality of musical parameters, sociological constraints). Although it may be extremely difficult in the real world to establish chains of musical influence, it is an inspiring idea to use influence mechanisms as tools to create music and to generate and observe evolutionary paths in artificial worlds.

This Thesis will tackle these issues under the scope of a new paradigm for the design and implementation of interactive musical systems, the Ontomemetical Model of Music Evolution - OMME (p. 125). By using the propositions contained in this model, this Thesis will demonstrate how music communication can be incorporated in a computational model, allowing the transmission of musical structures, and the emergence and development of musical styles in an artificial and controlled environment.

Summary

This Chapter introduced the viewpoint from which the main subject of this Thesis will be addressed: music as a human intelligent ability, culturally conditioned, and an act of communication. The way human beings perceive, organize, and generate sounds is intimately connected to structures in music. The replication of these structures defines one's personal musical identity, i.e., one's musical style.

It is not clear what exactly music communicates once music and verbal communication are of different nature. It seems incontrovertible however that emotion is an important part of this communication. Music communication enables the transformation of people's musical styles.

In order to understand from the human perspective how music communication occurs, the next Chapter will explore some of the human faculties that have a direct connection with the perception and cognition of music.

Chapter 2 Human Faculties

> "... in music that has communication as its goal, the structure of the music must take into consideration the structure of memory - even if we want to work against that structure" (Snyder, 2000, p. 3).

Because music is an ability that enables communication, it is important to explore which human faculties are associated with music. Some of these are physiological abilities provided by our sensory organ, the ear, while others are mental processes that help us to understand and to create music. In fact, they all work together within an integrated system.

Attention, perception and memory are all components of the "cognitive system" and must be jointly addressed to be understood (Styles, 2005). These, among other issues, are studied by Cognitive Psychology, a branch of Psychology that originated from the Gestalt school. Cognition involves all "processes by which the sensory input is transformed, reduced, elaborated, stored, recovered, and used. It is concerned with these processes even when they operate in the absence of relevant stimulation, as in images and hallucinations" ((Neisser, 1967) in (Flanagan, 1991, p. 178)).

New techniques (e.g., fMRI - Functional Magnetic Resonance Imaging) have been introduced to follow the execution of specific tasks (e.g., seeing, hearing) by tracking the blood oxygen levels in the brain. As a result, Cognitive Neuroscience has been able to evaluate several theories about the functioning of the biological mechanisms that underlie cognition, some of which will be presented in this Chapter.

2.1 Memory

According to Snyder (2000, p. 3), "the organisation of memory and the limits of our ability to remember have a profound effect on how we perceive patterns of events and boundaries in time". Welch (2000, p. 1) explains that the "mind's basic design is such that we are able to make sense of our sonic world through the utilisation of a hierarchical signal processing capability that appears to progress from the perception of psycho-acoustic features (such as pitch, loudness, duration, timbre), to structures (detecting/constructing patterns and regularities in the sounds), and (subsequently) to music's syntactic and communicative elements (being the potential for musical sounds to be characterized by a grammatical function within the musical context: music as a form of language)".

From a physiological perspective, memory is the "ability of nerve cells (neurons) in the brain to alter the strength and number of their connections to each other in ways that extend over time" (Snyder, 2000, p. 4). Hebb ((1949) in (Feng, 2004, pp. 305-306)) explained how neurons contribute to learning: "When an axon of cell A is near enough to excite cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased".

A number of factors can influence the way we perceive the world and store information into our memories. Our attitude in relation to the environment is one of them and defines our listening experience, a field explored by Huron under the concept of "listening modes" (e.g., distracted, tangential, metaphysical) (Huron, 2002a). Surprisingly, sleep could also play an important role on what memory retains and what is forgotten. Recent research (Payne *et al.*, 2008) demonstrates that sleep would help to preserve emotional experiences and to eliminate background details.

In fact, our attention is affected by a phenomenon called habituation as it is drawn to elements that are not stable, constant or predictable such as, in music, patterns that entail surprise: unexpected changes in harmony, melodic contour, etc. In Snyder's words, habituation results from

> "... the fact that when many types of neurons are stimulated repeatedly with an identical stimulus, their output of impulses does not remain constant, but instead decreases over time (...) Because habituation applies to aspects of experience that are fairly stable and constant or repetitive, aspects of the environment that are the most predictable or constant (and hence that match recent memory) will tend to move into the background, out of the focus of conscious awareness (Snyder, 2000, p. 24)".

Figure 2 below provides an overview of a generally accepted model (Snyder, 2000) of the human auditory cognitive system from the perception of sound to the formation of memory. This model will be closely followed by one of the interactive musical systems that have been developed in my research during the Ph.D., the Interactive Musical Environments - iMe, where artificial agents process musical information captured by their "sensory organs".

Initially, sound is captured by the ears from the environment via specialized neurons responsible for extracting a number of features ("feature extraction"). Next, the output of these neurons is taken in parallel and combined into coherent events through various types of correlation or synchronization in time ("perceptual binding"): events that are similar or closer in time constitute conceptual categories ("perceptual categorisation"). These processes are reductive in the sense that information is lost from what is captured by the ears to what is effectively stored in the memory.

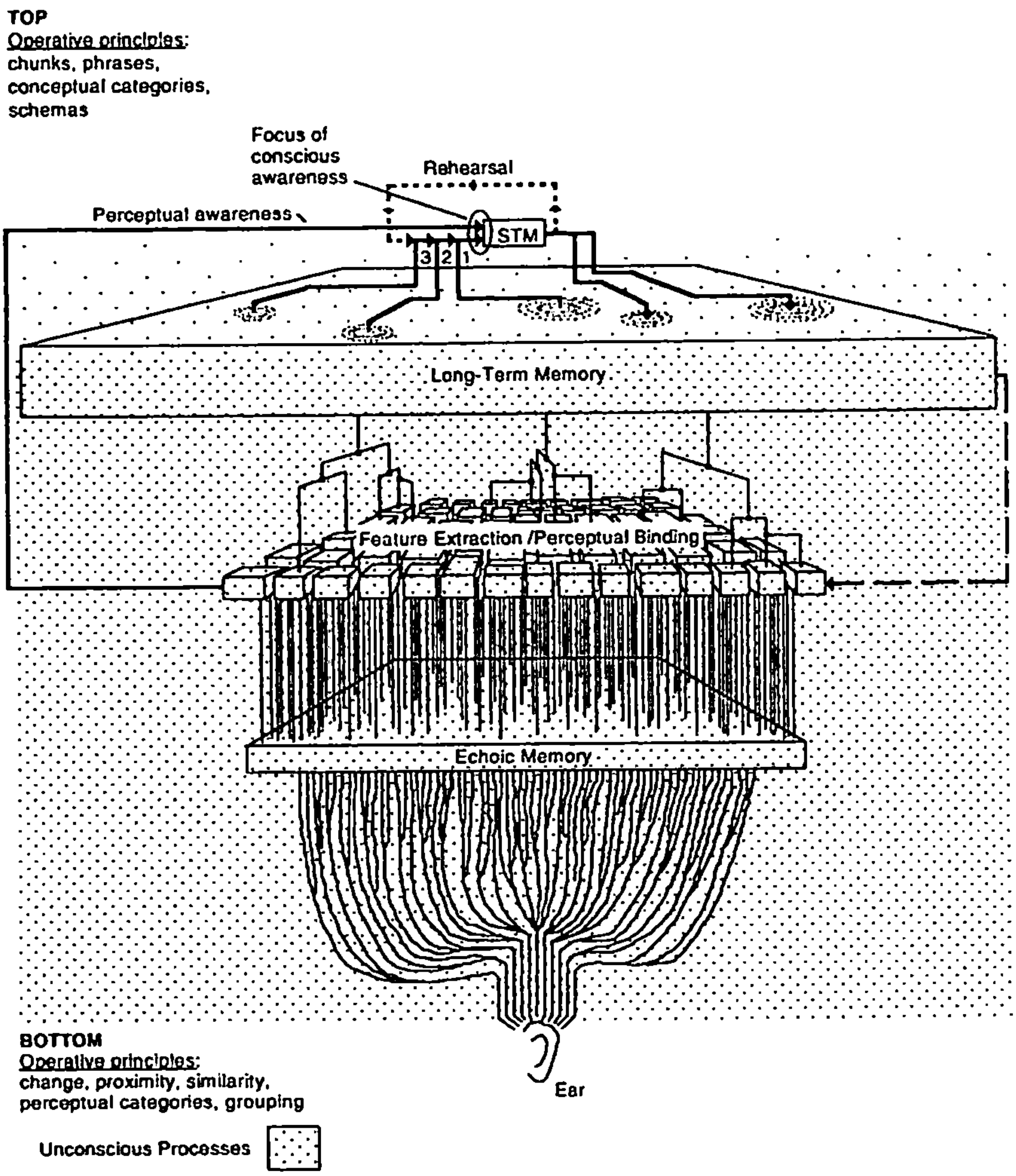


Figure 2: Some aspects of auditory memory. [2]

The above-mentioned model represents the memory on the basis of functional processes roughly classified by temporal duration of memory retention: the echoic (or sensory), short-term (or working) and long-term memories. Echoic

[2] The reproduction of Figure 2 from Bob Snyder's book, Music and Memory: An Introduction (Cambridge, MA: MIT Press, 2000) was generously authorised by the author.

memory corresponds to an early processing of the sound flow (200 to 500 milliseconds after the sound is perceived) where "event fusion" occurs and single events such as pitch are identified. Short-term memory is responsible for melodic (e.g., range, motion) and rhythmic (e.g., timing, intensity) groupings or patterns of events. Lastly, long-term memory has the ability to retain a larger time-scale, such as entire sections of music.

The degree of consciousness involved in each one of these processes is different. In Figure 2 the number of dots in the background illustrates this fact: the decrease of density from bottom to top represents the increase of consciousness. Only a small part of the memory ("focus of conscious awareness ") is fully conscious at any given time.

2.1.1 Short-Term Memory

The short-term memory - the memory of the immediate past - is sometimes also called "working memory" and consists not only of "immediate perceptions" but also of "related activated long-term memories, as well as contextual information that is semi-activated but not in consciousness and information that has just been in consciousness. Because it includes things both on the fringe and at the center of consciousness, working memory is not entirely identical to consciousness" (Snyder, 2000, p. 49)

Short-term memory would not cause enduring changes in the connections between neurons. It retains information immediately available for recall from a few (3 to 5) seconds to as much as one minute. Not only is the duration of the information limited but also the number of items that the short-term memory is able to retain, normally between five to nine (7 ± 2) elements (Miller, 1956). An element here means not only the "basic elements" of a sequence but also to collections of other elements that result from a process called chunking (or grouping). The capacity of the short-term memory is therefore increased without necessarily increasing the memory load (Snyder, 2000).

Chunking and association are interconnected processes, "whereby events that happen close together in time or seem similar form memories that are connected together (...) anything activating one of the associated memories may also activate the other memory" (Snyder, 2000, pp. 69-70). Groupings happen because of "the natural tendency of the human nervous system to segment acoustical information from the outside world into units, whose components seem related forming some kind of wholes" (Snyder, 2000, p. 31). They can be primitive or learned.

Primitive grouping (or "schema-driven" grouping) is innate to human kind, a "bottom-up" process that involve echoic and short-term memory and factors such as proximity, similarity and continuity (Snyder, 2000, pp. 39-42).

> "When some aspect of the acoustical environment changes sufficiently, a boundary is created. This boundary defines where a grouping begins or ends, and is the most basic kind of feature that is detected in the earliest stages of perception. Often referred to as "closure," the establishment of grouping boundaries is the quality that makes a grouping seem relatively self-contained and separate from other groupings" (Snyder, 2000, p. 33).

Learned grouping is a "top-down" process that happens after primitive grouping. It involves the long-term memory, relationships over longer time spans and factors such as intensification, parallelism and the occurrence of tonality (Snyder, 2000, p.43).

Groupings can occur in different levels or time scales. Melodic or rhythmic grouping include a small number (5 or less) of events and would take place early in perception. In the case of melodic groupings, boundaries can be defined by factors such as changes in relative pitch intervals, direction of motion or both. Rhythmic grouping can be defined by changes in time intervals, accents, etc. Changes in loudness, articulation and tone colour can also contribute to define boundaries. At a larger time-scale, groupings (or group of groupings) define phrases, which are the lengthiest elements that short-term memory can accommodate (Snyder, 2000, p. 38). Concurring boundaries usually happen at the phrase level and help to define the sense of closure.

Cognitive psychologists have addressed the issue of groupings since the advent of the German Gestalt Psychology school founded by Max Wertheimer, Kurt Koffka, and Wolfgang Kohler in the early 20th century. By definition, the German word "gestalt" means "shape" and contains the idea that the human senses are driven by the perception of the whole (e.g., a physical, psychological or symbolic entity) before the perception of the parts.

The fundaments and principles of Gestalt Psychology share many points in common with the recent theories of Cognitive Psychology described in this Chapter. One of the Gestalt's cornerstones is the law of "prägnanz" (or conciseness), which states that perception is organized at its "best, simplest and most stable shape". Huron (2008) mentions that, for Meyer (Meyer, 1961, pp. 86 and 128), the law of Prägnanz "states that psychological organization will always be as "good" as the prevailing conditions allow (...) the mind, governed by the law of Prägnanz, is continually striving for completeness, stability, and rest".

In order to understand what a "good" organization would be, other principles are put forward (Huron, 2008). The "law of good continuation", for instance, states that a "shape or pattern will, other things being equal, tend to be continued in its initial mode of operation. (...) Among other things this law helps to account for our being able to hear separate, discrete stimuli as continuous motions and shapes" (Meyer, 1961, p. 92). According to the "law of return", "other things being equal, it is better to return to the starting point whatsoever, than not to return (...) the term "return" need not be taken literally; that is, the opening materials may indicate what the final tone of a piece is to be without explicitly presenting it in the opening moments" (Meyer, 1961, p. 151). These and other principles (e.g., closure, similarity, proximity, symmetry) have been addressed in a number of publications in the musical field (Tenney *et al.*, 1980; Cambouropoulos, 1997; Bod, 2001; Thom *et al.*, 2002; Dhande, 2003; Jan, 2004; Pearce *et al.*, 2004).

2.1.2 Long-Term Memory

Contrasting with the other (echoic and short-term) memories, the long-term memory has the ability to store large quantities of information and be recalled after years. Short-term memories that are consciously rehearsed can become part of permanent long-term memory (Snyder, 2000). This is so because long-term memories produce actual changes in the strength of neuronal connections.

However, because most of long-term memory's content is unconscious, it must be re-activated in order to get to the focus of conscious awareness (the highest state of activation or one's immediate conscious experience). If long-term memories are not re-activated after a long period, associative connections with other memories become weaker and the result is "forgetting" things (Snyder, 2000, p. 71).

Retrieval from long-term memory can be unconscious (e.g., recognition and reminding) as well as conscious (recollection):

> Which "incoming information actually gets selected for our consciousness is determined by factors such as its novelty, and its relation to our goals, values, and so on (...) LTM acts as something like a filter, determining which aspects of our environment we are aware of at a given time. The presence of this type of connection between the early stages of perception and LTM essentially means that perception and memory are often impossible to separate; there may be little difference between memory processing and memory storage" (Snyder, 2000, p. 11).

Finally, all the memory processes affect our musical experience:

> "Echoic memory and early processing provide our immediate experience of the present moment of music in the focus of conscious awareness, and help to segment it into manageable units; short-term memory establishes the continuity and discontinuity of that moment with the immediate past; and long-term memory provides the context that gives it meaning, by relating the moment to a larger framework of ongoing experience and previous knowledge" (Snyder, 2000, p. 15).

2.2 Creativity

I have mentioned above (p. 25) that people express their musicality not only through highly skilled activities such as composition and instrument performance. When someone chooses to buy a particular CD in a music store or to tune an FM station on the radio, these choices necessarily indicate personal preferences, tastes, interests, something that I prefer to call "musical worldviews", an idea very close to musical style. The fact that someone plays a CD or a FM station to somebody else implies the idea that the second person will somehow react to (e.g., enjoy, dislike) and receive influence from these media. The concept of influence studied in this Thesis is more restrict than these examples, however.

My concern is with the continuous cycle through which certain musical structures end up in composer's minds and compositions. Other composers will hear these structures and use them in their own compositions. In order to understand this process, it is important to address some aspects of the cognitive system associated with musical creativity, an extremely complex subject. Some of these issues will explain the reasons for which this Thesis is centred on the paradigm of improvisation.

The ability to create radically new things is seen as the climax of human intelligence (Wiggins, 2006, p. 450), yet creativity is very difficult to define and understand. Creativity implies the idea of novelty; some aspect or property of the new creation must be previously non-existent and cause surprise in the receiver.

Recently, creativity has been addressed from a computational perspective. Computational creativity would be "the study and support, through computational means and methods, of behaviour exhibited by natural and artificial systems, which would be deemed creative if exhibited by humans (...) This "study and support" may, of course, include simulation" (Wiggins, 2006, p. 451).

Boden (1994a; 1994b; 2003), a well-known name in the field of computational creativity, understands that there would be two distinct levels of creativity: psychological and historical. Psychological (or p-creativity) exists when the creation is new at least for the one who creates. If the creation is new for humankind it should be called historical (or h-creativity). Departing from and formalizing Boden's ideas, Wiggins (2001; 2006) proposed a mathematical model that would allow objective comparisons between systems regarded as creative by humans. In practical terms, however, it is not clear how this model could be applied to assess the creativity of a musical piece.

Johnson-Laird (2002) advocates the idea that creative processes should be regarded as computable. His model NONCE defines creativity based on five components: "creativity is Novel for the individual, Optionally novel for society, Nondeterministic, dependent on Criteria or constraints, and based on Existing elements" (Johnson-Laird, 2002, p. 420).

According to the same author (Johnson-Laird, 2002), taking into consideration generative and evaluative stages, there are only three categories of computational algorithms that could be considered creative. The first one is neo-Darwinian: ideas are generated randomly and then critically evaluated, filtering out results that are not viable. Johnson-Laird understands that for real-time improvisation this alternative is not viable because it generates too many outputs.

Secondly, in neo-Lamarckian algorithms, an individual should master a set of criteria or constraints of a genre to generate new ideas. Where there is more than one possibility, selection should be done arbitrarily. Finally, the third algorithm is a compromise between the previous two and "appears to underlie many sorts of artistic and scientific creation" (Johnson-Laird, 2002, p. 439): initial ideas are generated under the guidance of some criteria and further criteria evaluate the results.

Enhancing creativity is another complex subject. Pachet (2003) asserts that interactive software would have the potential to enhance individual creativity in music improvisation, something that he demonstrates with his own interactive musical system, Continuator.

2.2.1 Improvisation

Improvisation is an unusual form of expertise (Johnson-Laird, 1991) and, a fertile soil for the study of musicianship and creativity, reason why the model of creativity adopted by this Thesis and, in special, by the interactive musical systems RGeme (p. 141) and iMe (p. 169), is intimately connected with this practice. According to a traditional definition, musical improvisation is the spontaneous creative process of making music while it is being performed and, to use a famous analogy, is like speaking or having a conversation as opposed to reciting a written text. From a physiological perspective, Pressing (1987) describes the improvisation process as a series of steps that happen roughly in this order:

> "1. Complex electrochemical signals are passed between parts of the nervous system and on to endocrine and muscle systems
> 2. Muscles, bones, and connective tissues execute a complex sequence of actions
> 3. Rapid visual, tactile and proprioceptive monitoring of actions takes place
> 4. Music is produced by the instrument or voice
> 5. Self-produced sounds, and other auditory input, are sensed
> 6. Sensed sounds are set into cognitive representations and evaluated as music
> 7. Further cognitive processing in the central nervous system generates the design of the next action sequence and triggers it. - return to step 1 and repeat" (Pressing, 1987, p. 2).

Actually, these steps are very similar to the ones of a "fixed performance", i.e., the performance of something that has been previously created. The major distinctions between these two forms of performances would occur in steps 6, 7 and 3. (Pressing, 1987)

As it encompasses musical performance, it is natural to observe that improvisation has a direct connection with performance-related issues such as instrument design and technique. It is known, for instance, considering the universe of musical elements played by improvisers, that certain musical ideas are more adapted to be played with polyphonic (e.g., piano, guitar) as opposed to monophonic instruments (e.g., saxophone, flute) or with keyboards as opposed to blown instruments, and so forth. Since instrument design and technique affect the ease or difficulty of performing certain musical ideas, different musical elements must affect the cognition of different players in different ways.

The technical or "performance part" of a musical improvisation is, at the same time, passionate and extremely complex but this subject is not the major concern in this Thesis. The interest here is primarily on how music is perceived, represented in the memory and the resulting cognitive processes relevant to musical creation conveys the emergence and development of musical styles. Improvisation is a particularly rich paradigm in which these issues can be explored.

One of the essential characteristics of improvisational music is the fact that once musical ideas have been created and played, it is not possible to go back in time and get them erased or changed. Each individual idea once performed is an accomplished fact, an "imposition" that requires completion that leads to other ideas, which themselves require completion. Newly played elements complete and re-signify the previous ones. If the whole reveals a unified structure it is because the exhibition of ideas follows a particular logic regarding a multitude of musical variables. But what defines this logic? How does it work during improvisation?

Johnson-Laird (2002, p. 417) once observed that musicians "can articulate only a limited answer, because the underlying mental processes are largely unconscious". This fact can be illustrated with a personal experience. Many years ago, after attending a jazz improvisation show, I approached the pianist - a friend of mine - and inquired about how he could build such wonderful lines. The reply was simple and straight: he said that he did not create those lines himself but it was the music that used him to create itself. An abstract and poetic thought; little informative, however.

Regarding the overall construction of the improvisation, jazz musicians normally follow certain conventions in order to develop their own ideas (Walker, 1997). Typically, the performance starts with the statement of the main theme, which is then followed by a number of choruses (a repetition of the entire piece structure) and ends with the re-statement of the theme. If the improvisation includes more than one musician, they normally decide beforehand who will play and in which order the solos will be played. Musicians can also take this decision during the performance itself by giving, for instance, visual hints.

The interest for this field from a cognitive perspective is that as the "structuring dynamics" of the musical improvisation progresses, decisions have to be taken in fractions of a second by the musician who chooses the outlines, directions and all the other elements that will be a part of the emerging performance. The solutions given to harmonization, vertical note distribution, largest or smaller concentration in this or in that area of the piano (timbre), rhythmic opposition offered by the left hand against the melodic ideas of the right hand, scales, are all issues that the musician has to address almost instantly. As stated by Johnson-Laird (2002) and explored in a number of computer systems, improvisation is a computable stochastic model "based on a greedy search through a highly constrained "what to play next" space" (Thom, 2001, p. 41).

The improviser creates a series of musical ideas (phrases, etc.) based on his own previous experience and, at the same time, adapts them to a lead sheet, the theme on top of which he improvises. In this continuous process two (or more) concurrent and different strata play inter-dependent roles: on one hand, a pathway (the "lead sheet") to which the generated ideas have to adapt and, on the other hand, the "flow of musical ideas" that is particular to each individual at each given moment. The chain of musical structures the improviser plays reveals instant reactions in respect to previously generated structures in such ways that the improviser's musical worldview is uncovered.

Summary

This Chapter explored some of the major faculties that enable human beings to appreciate and communicate through music. At the centre stage of the cognitive system, memory was defined as the human ability to store and retrieve information.

The organization of memory has a great impact on how humans perceive and understand music. Based on temporal duration, memory organization could be defined as three interconnected processes. Initially, echoic (or sensory) memory corresponds to the early processing of the sound flow. Short-term (or working memory) retains information immediately available for recall. Long-term memory produces actual changes in the strength of neuronal connections, which enables the storage of large quantities of information that can be recalled after years.

To conclude, in the last Section of this Chapter, the subject of musical creativity was introduced. Music improvisation, an unusual form of expertise and an especial case of creative behaviour, was chosen as the paradigm around which the interactive computer systems introduced in this Thesis will be based.

The next Chapter will continue to explore the main subjects that pave the way for the main outcome of this Thesis and extensively examine the state of the art of the computer models that investigate musical creativity.

Chapter 3 Computer Models

> "I do not believe that one can compose without using at least some algorithms. I could even argue that those who rely extensively on intuition for their music actually use algorithms subconsciously." (David Cope interview *in* (Muscutt, 2007, p. 20))

This chapter presents an extensive overview of computer systems that deal with music. Some of these systems are inspired by natural phenomena, trying to mimic, for instance, the processes executed by the human mind, while others don't have this special concern and mainly manipulate sounds algorithmically. The number of subjects is immense and an appropriate taxonomy difficult to reach, reason why it is possible that some of the systems presented here may look "misplaced" as they use different techniques at the same time. Therefore, rather than proposing a definite systematics for the classification of computer musical systems, the organization of this chapter roughly follows the flow of the musical information from (i) the perception of sounds to (ii) the acquisition of musical knowledge and (iii) the manipulation of this knowledge in generative processes. The sole objective is to address a number of approaches and paradigms in order to situate and introduce the Ontomemetical Model (Chapter 6, p. 125).

In fact, computers are ubiquitous nowadays in virtually all fields of human knowledge. In music, many applications have been specially designed to generate, analyse and perform compositions. From purely logical devices, computers steadily shifted to new paradigms where the concept of "machine musicianship" became achievable. This idea, explored by Rowe (2004), synthesises at the same time the theoretical foundations of music perception and cognition and the ability to artificially analyze, perform and compose music with computers.

3.1 Music Perception

Modelling human perception involves the discovery of structures of different types and hierarchies as, while the music stream is sensed by the ears, and cognitively processed by the memory, we make associations between the structures that emerge from the sound flow. Concerned with these issues, a computational theoretical model called General Computational Theory of Musical Structure (GCTMS) was proposed by Cambouropoulos (1998), aiming at describing structural elements in music. The idea was to capture features and structures that would be recognized by a listener and, therefore, included concepts typical to human cognitive abilities (e.g., abstraction, recognition identity and/or similarity, categorisation). Listeners are able to make generalisations and to learn, which means that elementary musical concepts such as pitch space and scales could be induced from musical examples. Once this knowledge is acquired, it could "be used to facilitate further processing of new musical pieces" (Cambouropoulos, 1998, p. 31).

In the GCTMS, a number of components separately address each analytical task, from music representation (General Pitch Interval Representation - GPIR), and segmentation (Local Boundary Detection Model - LBDM) to the definition of structural models (Accentuation and Metrical Structure Models), among others.

GCTMS does not require music to be previously marked with any higher-level structural element; it may consist of just a sequence of symbolic events (notes, etc.) that are then translated into the system's own representation.

Segmentation is the next step, for which GCTMS takes into account Gestalt Psychology principles (p. 37), a paradigm adopted by several other researchers (Polansky, 1978; Tenney *et al.*, 1980; McAdams, 1984). Deutsch (Deutsch *et al.*, 1981; 1982a; 1982b), for instance, addressed how Gestalt rules could be applied to pitch combinations. Lerdahl and Jackendoff's Generative Theory of Tonal Music (1983) also uses Gestalt principles to define boundaries and groupings.

Music segmentation is in fact a general issue for many of the systems that explore music cognition or music analysis:

> "Most systematic theories of music suffer on the issue of surface segmentation (...) They all require some sort of pre-processing of the surface into segments which relies on explicit/implicit knowledge on the part of the human musician/analyst (Cambouropoulos, 1998, p. 25).

Segmentation very often represents a source of trouble and, to a large extent, despite the many progresses achieved so far, remains an unsolved problem. There are a multitude of parameters (melody, rhythm, etc.), and levels of hierarchy to be considered. Segmentation algorithms should take into account, for instance, "low-level discontinuities in the musical surface and higher-level emerging patterns due to musical parallelism" (Cambouropoulos, 1998, p. 25). The difficulty is that, in many cases, overlapping segments and various acceptable solutions exist for the same music.

In order to manage these problems, researchers frequently adopt simplified approaches. For instance, abstractions are used so that, instead of considering the pitch of notes, pitch intervals are taken into account (Deutsch, 1982b).

Overlapping segments and different levels of hierarchy are also disregarded (Thom *et al.*, 2002). Of course, these strategies have the potential to compromise the results of the segmentation, something that researchers must consider.

A number of authors (Hasty, 1978; Baker, 1989b; Baker, 1989a; Camilleri *et al.*, 1990; Chouvel, 1990) have proposed different algorithms to deal with segmentation. The Local Boundary Detection Model (LBDM), proposed by Cambouropoulos (1998) initially takes the sequence of notes from the musical surface and builds a representation of intervals. It then attempts to detect "perceptual discontinuities" or "perceptual boundaries" in parameters such as durations (long notes) or melodic leaps. The model tries to discover points of maximum local change based on two rules (identity-change and proximity rule) inspired by the gestalt principles of similarity and proximity. A discontinuity coefficient is assigned to each pair of notes in a melody, which determines the boundary strength.

Belinda Thom and colleagues (2002) presented a comprehensive review on several algorithms for melodic segmentation comparing the results with segmentation performed by musicians. Amongst the algorithms mentioned in this review are the LBDM, Tenney and Polansky's (1980) computer model for the segmentation of melodies and Bod's (2001) "memory-based" model, which uses a data oriented probabilistic grammar technique for the segmentation and analysis of melodies. In the Paradigmatic Analysis (Nattiez, 1990) music segmentation is done via decomposition into classes of "significant units".

The Implication-Realisation model (Narmour, 1990), also inspired on Gestalt principles, involves the analysis of processes that would occur in the perception of melodic structures. Implication structures are expectations that guide both music perception and creation and correspond to stylistic influences received through exposure to musical contexts. They imply realisations, which are archetypes for possible implied continuations.

In iMe, one of the musical systems introduced in this Thesis, an optimal solution is pursued by artificial agents who use a combination of the "perceptive information" (e.g. melodic direction, melodic leap, melodic inter-onset interval, etc.) extracted by their "sensory organs" in order to segment the musical flow (p. 175). Other systems that tackle music segmentation in different forms and contexts are Cypher, Grouper, and the Sonic Object Analysis. Cypher (Rowe, 2004) uses a number of collaborative intelligent agents specialized in different music parameters (harmony, register, dynamics, etc.) to locate phrases' boundaries. Grouper is Temperley's (2004) rule-based system adapted from Lerdahl and Jackendoff's (1983) Generative Theory of Tonal Music that defines a set of preferences (gaps, phrase length, metrical parallelism, etc.) to define segment boundaries. Lima (2000) introduced the Sonic Object Analysis, a multi-agent system for the automatic segmentation of musical flows. Finally, neural networks have also being used for rhythmic segmentation of melodies (Carpinteiro) and in the case of post-tonal music (Isaacson, 1996).

After segmentation, once structures are defined, one must compare the several resulting segments. It is relatively easy to establish exact matches between structures but finding "similar structures" is something much harder. During the course of this research, in cooperation with Martins, Manzolli and Maia Jr. (Martins *et al.*, 2005), I proposed an algorithm to measure the similarity between sub-sequences in a general rhythmic space using a structure called Similarity Coefficient Vector. In this model, all sub-sequences of a given rhythm are compared: a hierarchical subdivision of rhythm sequences is done in several levels, and a Distance Matrix for each level is computed using the "block distance". The information about the similarity of the rhythmic substructures is retrieved from the matrices and coded into the Similarity Coefficient Vector (SCV) whose entries estimate the similarity between rhythm sequences in different k-levels.

3.2 Musical Knowledge

In this section some computer systems are introduced under the viewpoint of how the musical knowledge is acquired or stored. Generally speaking, knowledge can be explicitly placed by human experts in terms of declarations (or rules) and/or ordered structures (or grammars). On the other hand, systems can learn from sets of data or examples (machine learning).

3.2.1 Rule-Based Systems

Rule-based systems, also known as expert or knowledge-based systems, try to explicitly encapsulate the human expert knowledge in the musical domain. The amount of elements that must be addressed to efficiently describe a musical piece is nevertheless huge and frequently unmanageable, a fact that explains the many flaws of this approach. CHORAL is a typical example as it encodes around 350 rules targeting the harmonization of melodies in Bach's chorale style, which address aspects such as chord progressions, and melodic lines of the parts (Ebcioglu, 1988).

Another example is a system introduced by Pachet (1998) to explore harmonic variations in jazz chord sequences. One of the possibilities is the "tritone substitution rule" according to which a dominant chord (ch1) can be replaced by another dominant chord (ch2) where the root of ch2 is the augmented fourth (or tritone) of ch1. This substitution is possible because it is known that the third and the seventh of ch1 are the same as the seventh and the third of ch2. Figure 3 below shows a C major dominant chord and its corresponding tritone substitution.

Figure 3: Tritone substitution.

Another frequently used chord substitution is also applicable to dominant chords and consists in preparing these chords with a minor seventh chord based on the second degree of the local scale. Pachet (1994, p. 1) also introduced the MusES system to experiment with "various object-oriented knowledge representation techniques in the field of tonal harmony". Several layers deal with musical knowledge of several aspects of music generation (e.g., harmony, intervals, scales). The system can make analyses of jazz chord sequences as well as automatically generate harmonisations and real-time jazz improvisations.

3.2.2 Grammar-Based Systems

Information can also be stored in grammar-based systems. As well as language, music can be described in "grammatical terms" as they are both sequences of ordered structures - a viewpoint that can be especially helpful for the generation and analysis of music. Grammars are a "finite set of rules, which enable the symbolic description of the structure of a potentially infinite collection of structured symbols" (Wiggins, 1998, p. 3). In fact, knowledge-based and grammatical systems are very similar, since "both approaches focus on the form being produced, and both are stated in terms of rules as to what can be juxtaposed with what" (Wiggins, 1999, p. 4).

Schenker adopts grammatical principles in his method for music analysis, known as "Schenkerian analysis" (Forte, 1983; Marsden, 2007). Broadly speaking, this method consists in submitting a piece of music to a series of reductions (e.g., auxiliary and passing note progressions) until the elementary overall structure - the "ursatz" - is revealed.

Similar approaches lie behind other generative and analytical systems such as Lerdahl and Jackendoff's Generative Theory of Tonal Music (GTTM) (Cambouropoulos, 1998). In this case, the aim is to describe the cognitive processes involved in tonal music in terms of groupings (based on Gestalt principles), metrical, time-span and reductive structures. Steedman's rules are another grammar-based musical system that aims to capture the structure of jazz and pop 12-bar blues pieces (Wiggins, 1999); in this search, mental processes that lead to expectation in jazz progressions are considered.

3.2.3 Machine Learning Systems

Knowledge can be obtained by induction, i.e., by inferring general laws from particular instances (Anon, 2005b). Some of the computer musical systems that use this approach are presented in this section, in addition to the main data structures used by them.

In music education students are usually encouraged to listen to other musicians (performers and composers) as to derive rules concerning interpretation or to learn improvisation. This experience allows the identification of certain structures and regularities and people naturally start to induce the laws that govern them. If in the future the same or similar events happen, connections will spontaneously arise with the previously learned material.

In the artificial world, instead of programming the computer with rules and constraints, which is the case of knowledge or rule-based systems, the aim is to let the computer "learn" from a data set of examples. Stochastic processes are the paradigm used in machine learning: "... the main idea is to capture the local patterns found in the learnt corpus, using probabilistic schemes. New sequences are then generated using these probabilities. These sequences will contain, by construction, the patterns identified in the learnt corpus" (Pachet, 2002c, p. 2).

Named after mathematician Andrej Markov, Markov models are the reference in this field: "Music has a big deterministic component as present events strongly depend on many previous events (...) Markov chains seemed a reasonable way to deal with music memory and correlation" (Jorda, 1991, p. 1). Drawbacks of Markov models are the absence of long-term information (Pachet, 2002c) and, thereby, the difficulty to capture the overall structure of musical pieces. In addition to that, the length of the musical context plays an important role in the efficiency of algorithms. Low order Markov chains do not efficiently capture probabilistic rules while at higher orders, although some short-term structures could be captured (Walker, 1994), the computational cost grows exponentially (Assayag *et al.*, 1999) and the model tends to simply replicate the original material (Dubnov *et al.*, 2003).

Trivino-Rodriguez (2001) presented a survey of Markov based musical systems. Hiller and Isaacson (1959) used this approach to compose the *Illiac Suite* in the 50ties, and were followed by many other researchers (Pinkerton, 1956; Brooks Jr. *et al.*, 1993; Conklin *et al.*, 1995). Xenakis also used Markov chains in the composition *Analogique*. Recently a number of systems use this model for music style research (Cope, 2004) and interactive music improvisation (Vercoe *et al.*, 1985; Raphael, 1999; Thom, 2000b; Pachet, 2003; Assayag *et al.*, 2006). iMe (Gimenes *et al.*, 2007b) also uses stochastic techniques for memory modelling and music generation.

Trivino-Rodriguez and Morales-Bueno (2001) used multi-attribute prediction suffix graphs to generate new music. Continuator (Pachet, 2003), Band-Out-Of-The-Box (Thom, 2000a), and ImprovisationBuilder (Walker *et al.*, 1992) use machine-learning techniques for music style modelling. Manzolli uses probability vectors and boundary functions on generative processes where numerical structures are mapped to musical parameters (Manzolli *et al.*).

3.2.3.1 Methods and data structures

Some methods and data structures have been successfully used by machine learning systems for musical style modelling. Examples are incremental parsing, prefix trees, prediction suffix trees and factor oracles.

Pachet (2002a), for instance, uses prefix trees to store an ordered tree of all the sub-sequences (weighted by their number of occurrences) of a musical sequence. He proposes a simplification by storing reductions of the sequence instead of the whole input sequence.

Jakob Ziv and Abraham Lempel proposed the Incremental Parsing algorithm in the context of lossless compression (Dubnov *et al.*, 2003). A dictionary of motifs is built by traversing a sequence of symbols,

> "...sequentially adding to a dictionary every new phrase that differs by a single last character from the longest match that already exists in the dictionary. Using a tree representation for the dictionary, every

> node is associated with a string, whose characters appear as labels on the arcs that lead from the root to that node. Each time the parsing algorithm reaches a longest-match node it means that the node's string has already occurred in the sequence. Then IP grows a child node, with an arc labelled by the next character in the sequence. The new node denotes a new phrase that differs by one last character from its parent" (Assayag *et al.*, 2004, p. 1).

Another data structure is the Prediction Suffix Tree (PST) (Ron *et al.*, 1996) that stores Variable Length Markov chains; an improvement proposed by Rissanen (1983) to overcome drawbacks (e.g., growth of parameters) of the original Markov model and that has many applications in data compression:

> PSTs are dictionaries "of distinct motifs, much like the one generated by the IP algorithm. However, in contrast to the lossless coding scheme underlying the IP parsing, the PST algorithm builds a restricted dictionary of only those motifs that both appear a significant number of times throughout the complete source sequence, and are meaningful for predicting the immediate future. The framework underlying the approach is that of efficient lossy compression" (Assayag *et al.*, 2004, p. 1).

Finally, Factor Oracle (FO) is an automaton that captures all factors (sub-phrases) in a musical sequence and a linear number of transitional links (pointers) (Dubnov *et al.*, 2005). Forward pointers (or factor links) at generation time allow the reconstruction of the original phrases. With backward pointers (or suffix links) it is possible to follow other sub-phrases that share the same suffix and generate context-based recombination of the learned material (Dubnov *et al.*, 2005). The OMax system (Assayag *et al.*, 2006) uses FO for musical style learning and improvisation in real time.

3.3 Music Generation

The generation of music results from the interplay between representations of the musical knowledge and generative processes associated with it. An initial generative paradigm explored at the early stages of Artificial Intelligence (AI) was algorithmic composition (Hiller *et al.*, 1959). Other methods include evolutionary computation, intelligent agents and biologically inspired models (e.g. a-life, cellular automata and swarms). It is important to mention that many of these models mingle together in complex musical systems, reason why some of the systems presented in one of the sections below could also be used to illustrate other sections.

3.3.1 Algorithmic Composition

In fact, the use of algorithms in music is probably as old as music itself. Cope claims that it would be impossible to "compose without using at least some algorithms" and "those who rely extensively on intuition for their music actually use algorithms subconsciously" (Muscutt, 2007, p. 20). An algorithm is simply a "step-by-step recipe for achieving a specific goal" and therefore, algorithmic music could be regarded as "a step-by-step recipe for creating new compositions" (Muscutt, 2007, p. 10).

An algorithmic approach for music composition can be found in the chant generation method used by Guido d'Arezzo (Rowe, 1999). Another famous example are the Musikalisches Wurfelspiel (musical dice games) attributed to Mozart, the idea being to create musical segments and use them in compositions where the order of the sequences is determined by throwing the dice (Cope, 1991).

Even though computers are not a pre-requisite for algorithmic composition, they make the implementation of algorithms much easier (Muscutt, 2007). Taking advantage of this fact, pioneers Hiller and Isaacson (1959) working at the University of Illinois at Urbana-Champaign with the computer ILLIAC IV created in 1956 what is known as the first computer-generated algorithmic musical piece, the *Illiac Suite for String Quartet*.

There are algorithmic musical systems however that do not necessarily focus on music but simply map or make associations between outputs of more general algorithms to music. These systems must be distinguished from those that embody musical knowledge (Miranda, 2002b).

3.3.2 Evolutionary Computation

It was during the XIX century that the main theoretical propositions about the origins and evolution of species were introduced. Lamarck (Packard, 2007) initially suggested that individuals have the ability to adapt during their lifetime to the environment and the results of this adaptation would be transmitted from parents to offspring.

Darwin (1998) viewed this phenomenon from a different perspective: individuals with favourable traits in relation to their environment would be more likely to survive if compared to individuals with less favourable traits. For this reason, after a number of generations, the population of individuals with favourable traits would grow and be more adapted to the environment. Eventually, the differences from the original species would be so significant that new species would appear. These ideas (natural selection, heredity, and variation) are at the basis of Darwin's theory of evolution by natural selection, introduced in 1859 in his book "The Origin of Species". Natural selection can be understood as the opposite of artificial selection, a process through which breeders try to reinforce some desired characteristics in animals by favouring selective reproduction. With natural selection, the environment would be responsible for this process of constant change. During the same period, Mendel (1865) worked with pea plants in order to demonstrate that the inheritance of some traits followed general rules (e.g., dominance) today known as the Mendel's Laws, which represent the first steps to modern genetics.

The fundamental ideas behind evolutionary models are transmission, adaptation and survival of the fittest. The existence of genes entails the transmission of particular traits but evolution "occurs when a transformation process creates variants of some type of information. Normally there is a mechanism which favours the best transformations and discards those that are considered inferior, according to certain criteria" (Miranda, 1999, p. 8). Reproduction allows the transmission of genes but evolution results from "mutational exceptions to perfect transmission and the differential success of the products of this imperfect transmission" ((Cavalli-Sforza *et al.*, 1981) in (Jan, 2000)).

Not only have the above-mentioned ideas nurtured much of today's research in Biology but also the study of cultural evolution. A growing number of researchers are developing computer models to study this subject, including musical evolution (Blackmore, 1999). Cope, for instance, developed his work on musical styles around the idea of recombinancy (reordering of basic elements), mentioning that it "appears everywhere as a natural evolutionary and creative process" (Cope, 1999, p. 21).

Miranda studies the music origins and evolution "in the context of the cultural conventions that may emerge under a number of constraints (e.g., psychological, physiological and ecological)" (Miranda *et al.*, 2003a, p. 91). In his system, a community of agents evolve "a shared repertoire of melodies (or tunes) from scratch, after a period of spontaneous creation, adjustment and memory reinforcement" (Miranda *et al.*, 2003a, p. 94). In order to achieve this target, agents possess motor, auditory and cognitive skills and evolve vectors of motor control parameters by imitating each other's tunes.

The rules that govern cultural evolution are not as clearly defined as the rules that foster biological evolution, however. The concept of survival of the fittest is particularly difficult in this context; perhaps a reasonable parallel could be found in systems guided by social pressures. Todd and Werner (1999), for instance, modelled mating-selective pressure in the origins of musical taste where a society of "male" composers and "female" critics evolve "courting tunes".

3.3.2.1 Genetic Algorithms

Genetic algorithms (GAs) are a particular case in evolutionary computation (Holland, 1992) and are defined as a search technique inspired by some of the concepts (e.g., inheritance, mutation, selection) of Darwin's theory on evolution by natural selection. Roughly speaking, a GA involves the successive generation of populations of chromosomes representing the domain to be explored. At each generation, the previous population of chromosomes is transformed by a number of operators (crossover, mutation, etc.) and a fitness function evaluates the suitability of the new candidates for a given solution. From one generation to another, only the fittest candidates survive.

GAs have been used in many musical applications (Horowitz, 1994; McIntyre, 1994; Jacob, 1995; Brown, 1999; Moroni *et al.*, 2000; Tokui *et al.*, 2000; Weinberg *et al.*, 2007c) to address a number of different issues, in special to generate material for composition and improvisation. The definition of an appropriate fitness function is not an easy task, however. In GenJam (Biles, 1999), for instance, the fitness function is executed by a human expert who evaluates each newly generated candidate. This approach, known as Interactive Genetic Algorithm (IGA) presents a serious problem since the number of generated candidates is normally huge. Vox Populi (Moroni *et al.*, 2000), another example of an IGA, adopts another alternative where the end user controls the fitness function in real time via a graphical interface. In any case, the selection of the fittest candidates relies on the judgement (previous musical background, etc.) of the human controller.

GenJam, short for Genetic Jammer, is a well-known example of a musical interactive system that incorporates an IGA. Biles (1994) defines GenJam as a student learning to improvise jazz solos. This system incorporates a pitch-to-MIDI converter, which allows real-time improvisations and "trading fours" with a monophonic instrument. In these settings, GenJam listens to the human's previous four measures; it then "maps what it hears to its chromosome representation, mutates the chromosomes, and plays the result as its next four" segment (Biles, 1998, p. 1). The initial population is generated stochastically and, as an IGA, an expert controls the fitness function, deciding if each individual phrase is suitable to survive in the next population.

In fact, the suitability of GA-based systems for the study of musical creativity is very limited, as they by no means simulate human cognitive behaviour. In Wiggins words,

> "...They lack structure in their reasoning - composers have developed complex and subtle methods over a period of centuries involving different techniques for solving the problems addressed here. No one would seriously suggest that an author of hymn tunes works in the same way as the GA presented here, so while we may be able to produce (near) acceptable results, doing so sheds little or no light on the working of the compositional mind" (Wiggins *et al.*, 1999, p. 12).

3.3.3 Intelligent Agents

Intelligent agents, also known as autonomous or software agents (Russell *et al.*, 2002; Jones, 2008), are adaptive systems that reside in a dynamic and complex environment in which they sense and act autonomously executing a series of tasks in order to achieve certain targets for which they were devised (Maes, 1991; Russell *et al.*, 2002). Many times agent-based systems are designed as multi-agent systems in order to achieve the simulation of social interactions. As Miranda (1999, p. 40) explains, language and music must be seen as cultural phenomena that emerge from "sociological interactions, rather than as a ready-made feature of the infant's brain" (Miranda, 1999, p. 5). Intelligent agents are therefore extremely valuable for modelling simulations in this context.

Agents can be modelled as separate individuals where each one is specialized in specific abilities, such as in the Society of the Mind proposed by Minsky (1988). Another possibility, however, is to model agents as single individuals that have the same, similar or different abilities. In the musical field, Cypher (Rowe, 2004) adopts the first approach while Miranda's virtual musicians (Miranda, 2002b), the second.

Miranda and Todd (2003b) have identified three approaches for the construction of agent-based systems for composition: (i) the rendering of extra-musical behaviour into sound, (ii) genetic algorithm-inspired, and (iii) cultural systems. A perspective in which agents do not necessarily perform musical tasks exemplifies the first case; some aspect of their behaviour (such as moving around a defined space, etc.) is mapped into sound. Interaction affects the agents' behaviour and the music they produce but this music, on the other hand, does not necessarily affect their behaviour.

In the second (genetic algorithm-inspired) approach (Miranda *et al.*, 2003b), agents artificially reproduce the mechanisms of the theory of evolution by natural selection (Darwin, 1998) mentioned in section 3.3.2 above. The agents' survival and reproduction depend on the music they produce and the system as a whole will tend to produce more "successful music". The third and final approach uses virtual agents and self-organizing processes to model cultural systems where reinforcement mechanisms evolve the agent's abilities. The result of this evolution is kept in the agent's memories rather than in their genes. Only the last approach, according to Miranda and Todd (2003b, p. 1), would allow the "study of the circumstances and mechanisms whereby music might originate and evolve in virtual communities of musicians and listeners".

Many examples of agent-based systems can be mentioned. ImPact (Ramalho, 1997), for instance, uses agents to activate an ensemble of actions that instruct how music should be created. Agents initially perform a harmonic analysis of a musical database to define segments for the creation and performance of melodic phrases. Based on this analysis, best-fit segments are retrieved from a set of segments previously recorded from human performances (the musical memory).

OMax models a topology of interactive agents focusing on different abilities (listeners, slicers, learners, etc.) (Assayag *et al.*, 2006). A system called Frank (Casal *et al.*) uses MPEG7 and genetic co-evolution techniques along with artificial agents in live performances. Another system, Andante (Ueda *et al.*, 2004) uses a mobile agents infrastructure for music creation and performance within a distributed environment. Wulfhorst et al (2003) introduced a generic architecture of a multi agent system, the VMMAS (Virtual Musical MultiAgent

System), for artificial agents and human musicians interaction via music events using the MIDI protocol.

Impett (2001) uses an agent-based system to generate musical compositions where, through the interaction of embodied behaviours that co-exist and interact in the same world, agents are adaptive to the changing environment to which they belong. Pachet (2000) uses agents in an evolutionary context to emerge rhythm forms in real time simulation games. Agents have perceptive and actuator modules and synchronously play different rhythms (tempo and cycle length). Based on the rhythms they perceive, agents transform the rhythms that they play by applying transformation rules.

Miranda's (2002b) mimetic model uses intelligent agents to embed mechanisms of musical evolution. He remembers that "there is strong evidence that mimesis, or the ability to imitate the actions of other people, animals, and nature, is a defining feature of the first evolutionary steps to human intelligence ... Our hypothesis is that mimetic interaction is one of the keys to bootstrap music in our virtual society" (Miranda, 2002b, p. 79). In this model, agents have motor, auditory and cognitive skills: "we do not consider the existence of a specialized musical organ, but a more general substrate of motor, perceptual, and cognitive abilities that influences the very nature of musical development" (Miranda, 2002b, p. 79). Agents are all able to hear and produce sound through a vocal synthesiser in addition to storing associations between motor and perceptual parameters into their memory. There is no collective memory or global supervision and agents cannot see what is in each other's memory. As they are programmed to imitate each other, after some time, a shared repertoire of melodies is created.

3.3.4 Biologically inspired models

Other biologically inspired models such as A-Life, cellular automata and swarms have also been used to address musical creativity and are presented in the following sections.

3.3.4.1 A-Life Models

A-life models try to replicate biological phenomena via computer simulations (Miranda, 2003) and deal with concepts (e.g., origins of living organisms, emergent behaviour and self-organisation) that can shed light to the understanding of the genesis and evolution of music.

Miranda and Todd (2003b, p. 6) observe that perhaps the most interesting application of a-life techniques "is for the study of the circumstances and mechanisms whereby music might originate and evolve in artificially designed worlds inhabited by virtual communities of musicians and listeners". A number of scholars have addressed this issue throughout history (Thomas, 1995; Wallin *et al.*, 2000) even though computer modelling has not been often used for theoretical validation.

3.3.4.2 Cellular automata

Cellular automata consist of a multidimensional grid of cells, each one possessing one state, at any given time, from a number of possibilities. A function determines the evolution of these states in discrete time steps. In a recent work, Kirke evaluates different possibilities of mappings for cellular automata (Kirke *et al.*, 2007). In Conway's Game of Life (Kirke *et al.*, 2007), for instance, the state of the cells (alive or dead) is determined by the state of their neighbours. At each time cycle, all the cells are evaluated and their states are altered according to these rules. The initial configuration of cells affects the dynamics of the system and can allow the emergence of interesting behaviours, especially in the visual domain. Table 1 shows the rules defined by Conway's Game of Life:

Time t	Condition	Time t + 1
Dead	3 live neighbours	Alive
Alive	4 or + live neighbours	Dead
Alive	1 or 0 live neighbours	Dead
Alive	2 or 3 live neighbours	Alive

Table 1: Game of Life.

Musical applications have taken advantage of cellular automata. Chaosynth (Miranda, 2002a) uses this technique to control a granular synthesis audio synthesiser. Camus (Miranda, 2001) uses two simultaneous cellular automata - the Game of Life and Demon Cyclic Space (Griffeath *et al.*, 2003) - to generate musical structures (chords, melodies, etc.):

> "We devised a Cartesian model to represent a triple of notes; that is, an ordered set of three notes that may or may not sound simultaneously. These three notes are defined in terms of the intervals between them. Given a starting note, the horizontal co-ordinate of the model represents the first interval of the triple and the vertical co-ordinate represents its second interval" (Miranda, 2003, p. 33).

3.3.4.3 Swarms

Swarms, the last of the paradigms mentioned in this survey, model groups of individuals for which some rules of behaviour are defined. Individuals interrelate through a process known as stygmergy, "a method of indirect communication in a self-organizing emergent system where its individual parts communicate with one another by modifying their local environment" (Anon, 2005a). As the behaviour of the individuals is not sufficient to predict the organisation of the system as a whole, swarms are considered to perform emergent behaviour.

Swarm Granulator (Blackwell, 2006) is a swarm-based system example. A human plays a musical instrument, which produces attractors around which the artificial particles gravitate. The rules followed by these particles are simple and involve the concepts of cohesion ("if apart, move closer"), separation ("if too close, move apart") and alignment ("attempt to match velocities") (Blackwell, 2006, p. 4).

3.4 EMI

Finally, in order to end this Chapter, a special reference must be made to one of the best-known examples of the exploration of creativity and intelligence in music, the EMI (Experiments in Musical Intelligence). Cope (1991) started this project (also known as Emmy) around 25 years ago to explore topics such as musical style simulation, recombinant music, pattern matching, and form detection.

The idea behind "recombinant music" is the notion that composers tend to reuse certain musical structures throughout their work. Cope argues that recombinant music is "a serious attempt to understand how listeners recognize the style of a composer or period, one of the more elusive and difficult to describe musical phenomena" (Cope, 1991, p. 23). Emmy analyses a corpus of musical pieces, extracts "signatures" (musical patterns) and rules and reassembles the material derived from this analysis into new music that inherits the style of the original pieces.

Initially, Cope (1999) conceived the idea of creating a computer tool in which the way he dealt with musical ideas would be embedded. If, at any point he felt the need for help due to a mental block the system would automatically generate a number of new measures the same way he would have generated them himself.

Initial implementations adopted the approach of encoding rules for part writing. Cope reports that results were not very satisfying and the system only produced "after much trial and error ... vanilla music that basically adhered to these rules" (Cope, 1999, p. 21). Starting with these experiments though, Cope came across all sorts of important issues such as how to segment the original music, how to rearrange or reassemble these segments and to make musical sense.

At a subsequent stage Cope experimented with Bach chorales, separating them into beats that were then saved and categorised in lexicons. The system stored the pitch to which the voices moved in the following beat. Again, results were unsatisfactory as no phrase logic was being analysed. New music tended to wander with no large-scale structure.

To solve this problem, structure information had to be stored along with note-to-note rules. New analytical modules were added to allow the preservation of the location of each segment in relation to sections of the original pieces. The "character" of each beat, defined as a "rhythmic configuration" and the number of pitches also had to be preserved in order to assure "continuity" in newly generated music.

Recent releases of Emmy's music have shown that the output is convincing, especially when played by human performers (Cope, 2000). However, the whole process is extremely complex and far from being autonomous. To summarize, Cope starts the production of recombinant music with the preparation of a musical database, a manual, tedious and time-consuming task that relies entirely on his musical expertise. A number of similar pieces have to be chosen in order to assure that the output is consistent. Key, tempo and meter must be considered for this analysis. Cope uses MIDI files that are converted into music notation for proofing and correction. He once mentioned that to code his first Bach chorales database took him many months (Muscutt, 2007).

Once the musical database is ready, Emmy analyses the pieces and deduces musical signatures and rules for composition. Signatures are simply recurring note patterns or musical structures that characterize some features of a particular musical style. Cope discovered that these elements last "two to five beats (seven to ten melodic notes) and often combine melodic, harmonic, and rhythmic elements ... usually occur from four to ten times in a work" (Cope, 1999, p. 23). Variations (e.g., transposition, diatonic interval alteration) are also taken into account.

A comprehensive pattern-matching algorithm is applied on the input material and all possibilities (partial or full matches) are calculated statistically. All segments are checked for hierarchical and harmonic function. Connectiveness is checked for melody, accompaniment and harmony assuring, for instance, that a rising melody will be followed by a falling melody.

During recombination (generation of new material), signatures must survive, keeping their original form (intervallic relations) and context (location). The overall form of one of the original compositions is used as reference for the new music. Emmy fixes these signatures in their original places and then "fills the gaps" based on the rules found during the statistical analysis following an Augmented Transitional Network (ATN) (Woods, 1970). ATNs are used in the definition of natural languages and are "designed to produce logical sentences from sentence bits and pieces that have been stored according to sentence function" (Cope, 1991, p. 26).

Finally, Cope scrutinizes the output; after the generation of a new set of music, all outputs must be examined. Only the ones he considers the "most convincing" are kept. From the final output, on average just one piece is accepted for every four or five that are discarded (Muscutt, 2007).

Summary

This Chapter explored the state of the art of computer models that investigate musical creativity. In computer science, Artificial Intelligence is the field concerned with reproducing human intelligence and, in order to achieve this goal, a number of approaches (e.g., rule-based, and grammar-based systems) are used. In various ways, these systems attempt to model how human beings deal with music-related issues, such as perception, knowledge representation, learning, reasoning, and creativity. Rule-based systems encapsulate human expert knowledge via explicit rules while grammar-based systems define a finite set of rules that describe the structure of this knowledge. Machine-learning systems, on the other hand, attempt to reproduce human knowledge acquisition processes.

Evolutionary computation (e.g. genetic algorithms) and biologically inspired A-Life models try to replicate biological phenomena via computer simulations in order to shed light into areas such as the origins of living organisms, emergent behaviour and self-organisation.

Finally, a special reference was made to EMI, the Experiments in Musical Intelligence, in which David Cope explores the simulation of musical styles with machine-learning and grammar-based techniques.

The next Chapter will tackle the concept of interactivity, examining the approaches adopted by a number of different computer systems. This (Chapter 4 Interactivity) and the following (Chapter 5 Interactive Systems Design) chapters conclude the exploration of the literature and the presentation of the subjects that prepare the way for the ontomemetical model introduced in Part III: Ontomemetical Systems.

Part II: Interactive Musical Systems

Chapter 4 Interactivity

> "Interactive technology is one of the hot concepts of the 1990s. Advertisers and entrepreneurs are effectively exploiting its allure to entice consumers to buy products and investors to invest in speculative ventures. The media are hyping the concept to capture readers and viewers. And in the past few years, it has become a powerful magnet for a rapidly growing number of artists with backgrounds in a wide range of disciplines, including the visual arts, music, dance, and theater. Suddenly, interactive computer art is everywhere: theaters, museums, galleries, and exhibitions and performance series..." (Saltz, 1997, p.117).

The expression "interactivity" is used in many different fields and contexts, something that often leads to misunderstandings (see Figure 4, below). Sometimes this word is used to reinforce the idea that a given system or device is controllable based on some sort of user action; this concern brings to focus the question of interface design.

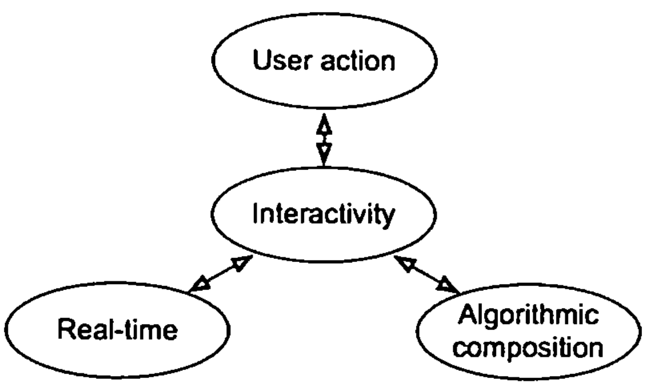


Figure 4: Interactivity in different contexts.

Interactivity can also refer to the fact that computer systems respond to incoming data. In this case the focus could be on algorithmic composition, among other subjects. In addition, the word interactivity is also used to specifically refer to "real-time interaction". In fact, it is possible, and many times desirable that all these elements (interface control, real-time intervention, etc.) coexist in the same system. Even so, for the sake of clarity, it is important to grasp the central meaning of interactivity.

However, before getting into more detail on how this idea fits into this Thesis, a number of examples will be introduced in order to understand how the development of technology has helped shape the definition of interactivity and consequently, of interactive musical systems (IMS).

4.1 A Plethora of Systems

Inquiring about the reasons why interactivity is such a fascinating subject, Saltz mentions that the experience of "hearing the system and the live performer adapt to each other's performances, in observing the development of a unique relationship between system and human" is much more captivating than simply listening to a pre-recorded performance: "... what is most interesting is precisely the feat itself, the action, the event" (Saltz, 1997, p. 124).

In fact, Jorda (2001, p. 1) remembers that the history of computer-based IMS goes "back to the late 1960s, initially involving computer-controlled analogue synthesizers in concerts or installations". In those early days live performance interaction was hardly feasible mainly because of reduced computational power. One would have to wait for the computer to calculate and generate a given musical material that would then have to be recorded and used along with some live setting.

As the processing power of computers increased, real-time algorithmic composition became more easily achievable. Many systems began to be designed and implemented having in mind that the generated material could be altered in real-time by directly controlling some of their parameters.

Walker remembers the pioneering work of American composer Salvatore Giovanni Martirano:

> "Professor Martirano was an early pioneer in the construction of and performance with interactive musical systems (...) His research with Sergio Franco culminated in the SalMar Construction, a sound synthesizer coupled to an algorithmic composition engine (...) While the system could operate autonomously, the user could intervene in the compositional and synthesis processes at many different levels of algorithmic abstraction" (Walker, 1997, p. 129).

The SalMar Construction system, by some called the world's first composing machine, as well as its successor, YahaSalmaMac, was used in many live presentations. The League of Automatic Music Composers is also often mentioned for its pioneering work with networked computers (Weinberg, 2005).

The first software to achieve a stricter kind of consistency according to Assayag would have been M, by Chadabe and Zicarelli (Assayag *et al.*, 2006). Consistency here refers to the idea of (re)producing a musical style adaptive to the performer. In order to accomplish this task, M used Markov chains to generate output based on what was played by a human musician (Walker, 1994). The advent of systems like Max and PureData two decades ago made the design and implementation of interactive systems more accessible by allowing the manipulation of objects in a graphical environment that models the flow of musical information.

Many IMS are conceived under the paradigm of meta-instruments. They capture the performance, interpret the incoming music data and respond according to a previously programmed series of algorithms and strategies.

There can be a human operator that controls the system by modifying some parameters during a performance in order to fine-tune some intended musical output.

By looking at all the above-mentioned systems, it is possible to observe that the nature of "interaction" is determined at the same time by the composition itself and the technology at hand. There are systems that can react to the gestures captured by a human player or to the sounds produced by an acoustic instrument. Rabisco (Manzolli *et al.*, 2002), for instance, captures drawings done with the mouse on a two-dimensional space (pad) and controls musical parameters such as notes ("x" axis) and velocity ("y" axis). Filters can be applied to produce notes of pre-determined scales. According to Manzolli (2002), the inspiration for this project was to employ a visual environment to enhance children's sound perception.

The QwertyCaster, a controller designed by Jorda, adopts a similar approach and is built using a computer keyboard, a trackball and a joystick pad, "all held together in a guitar-like piece of wood, connected to the computer" (Jorda, 2001). In a recent and more sophisticated project, the reacTable, Jorda (2003) uses, instead of simple and traditional devices (mouse, keyboard, etc.), a translucent round table.

Impett (2001) mentions the use of a pitch-to-MIDI converter connected to an adapted trumpet as a technical resource to allow interactivity. Lewis (2000) uses a similar device in his trombone and the Voyager system as well as Jorda in PITEL. The latter can generate up to four monophonic voices "under the control of a mouse-conductor" (Jorda, 2001, p. 3).

Hsu (2006) reported the development of an interactive system that creates a repository of gestures and timbre contours based on certain musical parameters (pitch, loudness and timbre) from a saxophone performance. Agents remap and transform elements from this repository in real time to generate responsive material. GRI (Morales-Mazanares *et al.*, 2005), is another system that captures "movement gestures" with accelerometers and gyros from a flutist improviser that are used to build styles of a "probabilistic transition automaton" that calculates responses in real-time.

Barry Vercoe's Synthetic Performer (Rowe, 1996) is an important reference in the area of score following and tempo tracking in a live performance. Within the same field, Dannenberg mentions the use of pitch-class real-time analysis to estimate musicians' current location (Dannenberg *et al.*, 1987).

Weinberg reports that the early works on musical robotics "primarily addressed mechanical keyboard instruments such as the Pianista (1863) by French inventor Jean Louis Nestor Fourneaux (...) In recent years, the field has received significant commercial, artistic, and academic interest, expanding to anthropomorphic designs (Rosheim 1994) as well as other robotic musical instruments, including chordophones, aerophones, membranophones, and idiophones" (Weinberg *et al.*, 2006, p. 29). Advocates of robotic systems propose that machines should sense the environment, manipulate it and "learn from the results of its manipulations through further sensing. Hence, the machine must be a robot that can learn" (Peters II *et al.*, 1999, p. 2).

Weinberg's Haile is one of these systems, a robotic anthropomorphic percussionist that is able to listen to human players in real-time, analyze perceptual aspects of their playing and play along collaborative and improvisatorially. The system's current version was extended to deal with melodic and harmonic information and play a one-octave xylophone (Weinberg *et al.*, 2007a).

Roboser is a robot integrated with an algorithmic-based generative system, CurvaSom. The system manipulates fragmented sequences of MIDI events based on internal states and sensory events captured by the robot (Manzolli *et al.*, 2000).

After these initial considerations, it is possible to realize that these systems address the issue of interactivity with rather different approaches. It is therefore of great interest to have a clearer definition of the field. However, before proposing a working definition of "interactivity", the next Sections will discuss in more detail some of the systems that have become a reference in this area.

4.1.1 Cypher

Conceived from the perspective of Minsky's (1988) Society of Mind, interaction is a keystone of Rowe's (2004) system, Cypher. Minsky views the human mind as a society of agents, which are individually responsible for the execution of simple tasks and that, together, contribute to the execution of more complex tasks. In Cypher, some of these agents are responsible for analyzing incoming musical events and others for producing new musical material based on the initial analysis.

Rowe explains that Cypher employs the algorithms initially developed by Bloch and Dannenberg (1985) for computer accompaniment in order to recognize "salient patterns from a MIDI stream (induction) and to flag subsequent appearances of those same sequences (matching)" (Rowe, 1996, p. 57). Cypher perceives a MIDI stream of data and classifies the behaviour of the incoming events as "regular" or "irregular" according to a number of features (e.g., loudness, speed and register). The system's user then establishes connections between this analysis and a compositional module whose algorithms control the generation of MIDI output, which is then sent to synthesizers, resulting in a new musical texture on top of the pre-existing one.

Walker suggests some examples of what sort of musical output could be produced using this type of approach:

> "... The human performer playing loud notes can cause Cypher to play more trills. While such individual connections are straightforward, several simultaneous connections produce complex and interesting behaviour. Other agents can reconfigure the connections, causing the system to behave differently during different sections of a performance" (Walker, 1994, p. 7).

Elaborating on Cypher aesthetics, Cambouropoulos remembers that even though this system is meant to be a general interactive compositional system, it is somehow oriented towards the Western tonal system:

> "For instance, vertical organisation of pitches is based on tonal harmonic relations of chords in a specific key and specification of grouping boundaries is biased strongly towards tonic and dominant cadential function of chords" (Cambouropoulos, 1998, p. 20)

4.1.2 Voyager

Voyager (Lewis, 2000) is another interactive system frequently mentioned in the literature that has been consistently used in improvisational performances for more than twenty years (Francois *et al.*, 2007a). Lewis regards his system from the same perspective of Robert Rowe's "player and instrument" paradigms.

As it happens with Cypher, Voyager analyses some aspects of the improviser's performance in real time, which then drives the generation of an automatic composer/improviser resulting in (up to 64 asynchronous) multiple voices or parallel streams of music production. Up to two musicians can be connected to the system either with a MIDI keyboard or a pitch-to-MIDI converter, as it is the case when Lewis plays his trombone.

Lewis explains that Voyager is responsive to the musician's playing and at the same time possesses an independent behaviour:

> "Several different (and to some, clashing) sonic behavior groupings, or ensembles, may be active simultaneously, moving in and out of metric synchronicity, with no necessary arithmetic correlation between the strongly discursive layers of multirhythm" (Lewis, 2000, p. 34).

Lewis suggests that Voyager embodies the "African-American aesthetics and musical practices" because this sort of system tends to reveal traits of thought and culture of the community that uses it. This claim is not particularly surprising if we consider that in systems such as this, the human performers generate the source material while the machine plays a cooperative role.

4.1.3 ImprovisationBuilder

ImprovisationBuilder (IB) is an object-oriented software framework developed by Walker (1997) in Smaltalk-80 to support the development of musical improvisation systems. Many issues belonging to this area are tackled, specially the structure of improvisation collaboration.

Despite the fact that IB does not implement tempo tracking, it adapts itself to the changing roles of group improvisation (soloing, accompanying, trading fours, etc.) based on "conversation analysis" techniques. Human performers can improvise freely as long as they adjust to the computer's tempo.

The main classes are Listeners, Transformers/Composers, Realizers and Improvisors. Roughly speaking, Listeners "sense" an incoming MIDI stream that is parsed into phrases by segmenting at the occurrence of silences longer than two seconds. Transformers and Composers generate new musical material by transforming (e.g., retrograde and transpose transformations) the incoming musical data or algorithmically creating new phrases (e.g., Markov chains and pattern instantiation by concatenating short melodic fragments or randomly walking on a given scale). Realizers control the execution of the generated material by assigning timbres. At last, Improvisors gather all the other classes in a linear chain from a Listener (in one end) to a timbre (in the other) in order to achieve a desired design.

One of the main interests of IB is however the implementation of the so called "structured improvisation" or the "conversation analysis", represented by a finite state machine, in which every state corresponds to a specific musical activity (e.g., soloing or accompanying) (Walker, 1997).

This is how Walker explains the conversation analysis mechanism:

> "Each state specifies a criterion for whether improvisational roles are about to change. An Accompany state interprets the appearance of bass notes in the input to mean that the other player is assuming the role of accompanist. Conversely, a Solo state interprets the appearance of treble notes to mean that the other player is beginning a solo ... When the current state satisfies its transition criterion, it passes control to the next state, which will give new instructions to the Harmony Generator and specify its own transition criterion" (Walker, 1997, p. 128).

4.1.4 Band-out-of-a-Box

Band-out-of-a-Box (BoB) was introduced by Thom (2001) as a system centred on the idea of an "improvisational musical companionship". The name of course is a reference to the well-known commercial application Band-in-a-Box (PGMusic, 2008) which, given a grid of chords, generates automatic accompaniments with a combination of different instruments, rhythms and musical genres. A typical Band-in-a-Box user is someone interested in improving improvisational skills by having a companion with which to practice.

However, interaction in BoB is implemented differently. In addition to having an artificial companion, the motivation lies on the possibility of having stylistically meaningful interactions. User and context are taken into account with the purpose of displaying stylistic consistency. The idea of a "playing mode" (in BoB's terminology) has some parallels with the concept of musical style presented in this Thesis (Music, p. 25): "... each user playing mode corresponds to a preference for certain pitch-class, intervallic, and directional trends" (Thom, 2003, p. 6).

The system models an artificial agent that trades solos (call/response) in real-time with a human musician, an approach called "trading fours" in jazz. Within this system, however, only one of two players (artificial or human) generates a monophonic line (melody) at a given time.

The overall performance is guided by a lead sheet and accompaniment prepared in advance. BoB generates responses to the human "calls" in real-time but according to the musical material learned during previous warm-up rehearsal sessions. During these sessions one bar length segments are collected, transcribed and converted into histograms of musical features, rhythm and melodic (pitch-class, intervallic and directional) content. A probabilistic improvisation model (the "history component") is then built based on the collected musical data (Thom, 2003). In generative mode, BoB generates one bar at a time and uses the history component to find and stochastically create responses to the human musician's most recent call.

The drawback of BoB's approach to interactivity is that influence is only apparent in the sense that the material played by the human performer in real time does not affect the memory of the agent. Thinking the other way round, a human improviser could (and probably would) collect the agent's musical material on the fly and adapt to it.

4.1.5 Continuator

The Continuator is a project run by Pachet at the Sony Computer Science Laboratories within the musical style domain. Pachet notices that interactive systems are "limited in their ability to generate stylistically consistent material" while composition systems are "fundamentally not interactive" (Pachet, 2002c, p. 1). The proposal would be "to fill in the gaps" with an "interactive reflexive musical system" that, for Pachet, is a branch of the more generic interactive musical system category. The "reflexive" element or behaviour would allow the human performer to engage in an experience of "self-discovery", producing an impression of similarity that incrementally conforms to the personality of the user (Pachet, 2006a).

Pachet mentions (2002b) that he was looking for a system that could not only learn and imitate musical styles but also be used as an actual musical instrument, allowing "real music interaction". Some aims were defined such as the system's ability to learn arbitrary musical styles automatically (without human intervention) and agnostically (without modelling any symbolic information). It would also seem integrated in the "playing mode of the musician", adapting to unexpected changes in rhythm, harmony or style in real-time (Pachet, 2002b). Additionally, this system would not possess a "standard graphical user interface", allowing "users to concentrate on playing music without thinking about the system design" (Pachet, 2006a, p. 14).

According to some accounts, it seems that these objectives have been successfully achieved. In recent case studies with children of 3 to 5 years of age, Addessi and Pachet (2005) reported that the Continuator was "able to develop interesting child/machine interactions and creative musical processes in young children".

The paradigm used for this analysis (Addessi *et al.*, 2006; Pachet, 2006b) is the theory of flow, introduced by Csikszentmihalyi's (1991) studies focusing on people's reports on activities that involve concentration (immersion) and deep enjoyment (or ecstasy, pleasure, timelessness, etc.). He observed that the achievement of satisfaction and happiness is a result of "flow" (or "optimal experience"), a state in which people are completely absorbed in their activities. The metaphor refers to the sensation of people being "carried by a current".

Some of the musicians that have had the opportunity to play with the system reported that it helped to trigger new ideas, to "create new forms of musical improvisation" and to understand the relation with one's own learning. (Pachet, 2002b, p. 80). In addition to that, the system seems to have succeeded during a Turing test (Pachet), where two music critics, Henkjan Honing and Koen Schouten, were unable to distinguish whether the music was being played by a human (jazz pianist Albert van Veenendaal) or by the machine. However, Collins does not seem convinced with this performance:

> "That the system passed a musical Turing test for two expert listeners run by a radio station is not so surprising since the output material maintains the same rates and densities as the input, as well as the same timbral base on a MIDI piano" (Collins, 2006, p. 24).

In practical terms, the system acts as a sequence continuator. Pachet explains that a continuation is not the same as an answer to a proposition:

> "When the system produces a continuation of a phrase played by the musician, the continuation is the second part of the same musical phrase. Conversely, a musical answer is a musical phrase in its own right, with a beginning and an end" (Pachet, 2002b, p. 81).

The note stream played by the human performer is segmented into phrases using a variable temporal threshold, which are subsequently sent to a phrase analyzer that builds up a statistical model of recurring patterns. Global parameters (e.g., note density, tempo, meter, dynamics) are also analyzed. An indexing scheme based on a prefix tree is adopted in order to represent all the sub-sequences of the musical input. (Pachet, 2002c).

A "phrase-end detector" detects when the human musician has finished playing and generates in real-time a continuation of the input sequence by exploring the Markov graph created by the analysis module. Pachet addresses the integration between man and machine not only by relying on the learned model of the analytical algorithms but also by taking the context into consideration. That is, the output is biased by elements such as the harmonic and melodic information that is currently being played by the human musician with the "adjunction of a fitness function which takes into account characteristics of the input phrase" (Pachet, 2002c, p. 6).

4.1.6 OMax

Another project focused on man/machine interaction flourished from a collaboration between several researchers, namely Assayag, Bloch, Chemillier, Lartillot, and Dubnov, among others (Durand, 2003). One of the main issues in this case is to address musical styles within the domain of improvisation. In 2003, the key outcome of this project was announced as an interactive system named OMax (Durand, 2003).

Objectives and pre-requisites have many similarities with the Continuator project. The initial design previews that, in order to achieve real-time interaction, learning should be incremental, fast and there should be an instant "switch" between learning and generation, something that OMax's researchers call "machine improvisation". In addition, the system should deal with multiple musical attributes simultaneously under metrical and harmonic alignment (Assayag *et al.*, 2004).

In effect, OMax is another successful interactive musical system, which is able to produce interaction with one or more human players with real-time stylistic learning and music generation, metrical and harmonic alignment. It is publicly available for download at the website of the project (Assayag, 2008). A series of concerts and experiments can be watched at the same website with the participation of clarinettist Jean-Brice Godet, saxophonist Philippe Leclerc and pianists Helene Schwartz, Dennis Thurmond and Bernard Lubat.

OMax combines resources from the well-known systems OpenMusic and Max/MSP and its execution requires that these systems be installed on the computer. Lisp-based OpenMusic is responsible for "deeper" style modelling (analysis and prediction) while Max/MSP for real-time processing (e.g., real-time control, MIDI and audio acquisition, and rendering). OMax current edition is capable of processing real-time audio and video as well as MIDI (Assayag *et al.*, 2006). A number of statistical non-supervised learning models (see p. 54) have been tested but the current implementation adopted Factor Oracle.

OMax can be used in "free" mode, where the computer captures the human performance and freely produces musical material in real time or in "beat" mode, where the computer generates musical material according to a chord grid. This project is influencing the appearance of other IMS, such as the improvisation engine adopted by Mimi (Francois *et al.*, 2007a).

4.2 Definition of Interactive Musical Systems

After the initial notes and examples introduced in the previous Sections, it is possible to build a working definition of interactive musical systems. A good initial reference can be found in the Oxford English Dictionary according to which to "interact (of two people or things)" is to "act so as to affect each other" (Anon, 2005a). "Act" and "affect" are visibly the two main expressions here. Actions are executed by people or things and produce an effect on other people or things.

Taking two initial examples from the area of interface design, broadly speaking the manipulation of controls such as pressing the key of a piano or clicking a mouse button affects the state of the key/piano or of the mouse/computer that generates the production of a sound on the piano or of an event on the computer. The perception of these effects entails more manipulations and reciprocal actions. These examples fulfil the initial definition of the previous paragraph.

One could argue however that the actions produced by the musician/user are of a different nature compared to those produced by the piano or the computer. The piano is a totally mechanical device, unable to produce any sound unless its keyboard is depressed. The response is also entirely deterministic in the sense that, given two identical actions, the same corresponding result is expected to happen. This type of interaction can be said to be purely responsive.

The computer, on the other hand, could be programmed to act not only responsively but also autonomously and non-deterministically. Given the advances in artificial intelligence, it is still a dubious question to evaluate how autonomous computers could really be. In this regard, Dobrian mentions that the

> "Computer can only be purported to be acting autonomously if it is programmed to make some decisions of its own that are not fully predicted by the algorithm. This implies inclusion of some elements of unpredictability: the use of pseudo randomness at some structural level" (Dobrian, 2004, p. 1).

Consciousness is another important element as even though computers can simulate a volitive behaviour, it is still an attribute that belongs to the "human realm". Interestingly, "interactive", the next entry in the Oxford English Dictionary, is defined as "influencing each other". In the context of computers or other electronic devices, interactivity allows a "two-way flow of information" with the user (Anon, 2005a). There seems to be no place for "volition" or "consciousness" in this definition. Instead, the main elements here are the existence of (at least) two ends between which a two-way flow of information is established.

Walkers mentions that many "composers have created musical human-computer interactions by running generative algorithms in real time and controlling their parameters with the human performer's input..." (Walker, 1994, p. 6). Nevertheless, should some systems be called "interactive" for the simple reason that they respond in real-time to some sort of user action? This does not seem to be an appropriate conclusion because very often these systems do not perform any "reciprocal influence". They could therefore be called "reactive" but not "interactive". IMS are much more difficult to design and implement (see Chapter 5, page 95) compared to simply reactive systems.

It seems desirable, therefore, that the expression "interactivity" is used within a stricter context. Rowe (1993) explains that an IMS is "a computer program that "listens" to the actions of a performer and generates responses in real-time"

(Cemgil *et al.*, 2001, p. 1). To attain interactivity, the processing chain must be conceptualized in three stages: initially, the data must be collected (sensed) from gestural or audio information. The computer then reads and interprets (process) the collected data and, finally, produces the output (responds) based on the previous steps (Rowe, 1996):

> "By transferring musical knowledge to a computer program and compositional responsibility to performers onstage ... the composer of interactive works explores the creative potentials of the new technology at the same time that he establishes an engaging and fruitful context for the collaboration of humans and computers" (Rowe, 1999, p. 87).

Numerous systems have been designed within this view such as Cypher (Rowe, 1996) and Voyager (Lewis, 2000). From a similar viewpoint, Impett explains that interactive music is "instantiated during performance on the basis of stored programs and materials, and performance and environmental information" (Impett, 2001, p. 1). According to Paine, "each interactive system also includes methods for constructing responses, to be generated when particular input constructs are found" (Paine, 2002, p. 3). The keywords here are "interpretation" and "response": the system "interprets" the user's action and returns a corresponding "response".

Garnet (2001) is more strict and understands that interactive computer music systems are a sub-genre of what "might be called performance-oriented systems", and that they would require at least one live performer, "joined with computer-generated or electronically produced or modified music" (Garnett, 2001, p. 21). In fact, this proposition does not seem to be accurate because, as mentioned before, important elements are the existence of a two-way information route that results from apprehending and analysing information, the ability to take actions and to change the inner states of the participants. This does not necessarily require real-time performance. iMe, the system introduced in this Thesis, implements both real-time and non real-time interactivity (iMe, p. 169).

Learning is another important issue. Rowe noticed twelve years ago that, although many systems possess certain methods for music generation, most of them do not really "learn":

> "Learning in real-time music programs has scarcely been explored (...) Many machine learning techniques demand large amounts of computation and (...) Adapting techniques for use in learning onstage may well have to wait for a faster generation of processors" (Rowe, 1996, p. 59).

The technical impediment mentioned by Rowe has to a certain extent been overcome. Nevertheless, it seems that today his statement is still true: even though there are a few systems that are able to learn from their environment, the majority of new systems that are still being produced are only concerned with the basic reactive approach mentioned above.

Pachet (2002c) observes that his system, Continuator, is an attempt to overcome the downside of IMS that are traditionally "limited in their ability to generate stylistically consistent material" and are unable to learn:

> Even though "a lot of work has been devoted to efficient controllers and interfaces for musical systems (...), these systems all share a common drawback: they are not able to learn, there is no memory of the past. Consequently the music generated is strongly correlated with musical input, but not or poorly with a consistent and realistic musical style" (Pachet, 2002a, p. 1).

Paine distinguishes interactive systems from systems that are simply reactive or responsive because they lack a minimum degree of cognition:

> "The terms interactive and interactivity are broadly applied in the new media arts, however, the diversity of application, has lead to a lack of focus. It will be argued that the term interactivity is therefore widely abused, and in line with Bongers (2000), that most systems are not interactive, but simply reactive or responsive, because they lack a level of cognition" (Paine, 2002, p. 1).

In the same trend, Blackwell (2006) introduces the idea of "live algorithms", whose requirements are (i) interact without human intervention, (ii) make creative contributions to the music being produced and (iii) avoid rule based approaches "or any simplistic mapping between input and output".

Dobrian also agrees with the idea that interactivity should involve cognition and mutual influence:

> The "prefix inter- in the word "interactivity" implies mutual influence between agents that are also in some way autonomous decision makers. Neither agent may be fully predetermined in its behavior; each must be able to modify its behavior - to improvise - based on unpredictable behavior by the other" (Dobrian, 2004, p. 1).

As a result of all the above-mentioned considerations, under the scope of this Thesis, I propose the following working definition for interactive musical systems:

> "Interactive musical systems are computer-based systems that, through musical information exchange, have the ability to perceive the environment, analyse and take actions as to alter the surrounding states as well as to change their own internal states".

Summary

We have seen in this Chapter that, in the last years, the concept of interactivity has become overused in many contexts with different meanings. In the musical field, its misunderstanding and misuse diffused the idea that even systems with simple reactive behaviour could be called interactive. Given these facts, in this Chapter I explored a number of so-called interactive systems in order to find the core elements that an interactive musical system must have in order to be called interactive.

Some of the most famous interactive musical systems were explored in more detail (Cypher, Voyager, ImprovisationBuilder, Band-out-of-a-Box, Continuator and OMax), especially in respect to their "interactive" elements. In light of these systems and to prepare the way for the introduction of the model proposed in this Thesis, I have proposed the following definition for interactive musical systems: "Interactive musical systems are computer-based systems that, through musical information exchange, have the ability to perceive the environment, analyse and take actions as to alter the surrounding states as well as to change their own internal states".

The next chapter, the last of the background chapters before the introduction of the ontomemetical model, will finally outline the elements that I have identified as important for the design and implementation of interactive music systems.

Chapter 5 Interactive Systems Design

> "Some instruments are more powerful, flexible or versatile than others, but there are so many dimensions by which a musical instrument can be evaluated, that comparisons between most instruments do not make sense, unless we clearly specified the parameters involved in the evaluation" (Jorda, 2003, p. 1).

The design and implementation of IMS involve numerous elements, many of which have already been addressed by researchers in different ways (Dannenberg, 1989; Borchers *et al.*, 1998; Borchers, 2000; Camurri *et al.*, 2002; Dannenberg, 2002). This Chapter will investigate some of the elements I consider the most significant to meet my working definition of IMS.

Among the many difficulties Dannenberg (1989) mentions, special attention should be paid to the occurrence of parallelism and of time-dependent errors and the need to use special purpose input/output devices. Other awkward issues include multitask programming (independent threads of control), sequencing, adjustment of many parameters, scheduling (long computation vs. real-time responsiveness), testing (requires real-time input by multiple performers) and the fact that most of the activity is asynchronous. All in all, systems should respond to many different types of controller-generated events, generate the appropriate responses with low latency rates in real-time, and lead to a desirable experience (Dannenberg, 2002).

A series of tools and techniques such as the use of specific programming languages have been proposed to deal with this sort of problems (p. 117). Almost two decades ago, Dannenberg proposed that certain techniques would help to achieve better results with IMS; these suggestions are still valid today:

> "(1) a program structure in which input events are translated into procedure calls, (2) the use of non-preemptive programs where possible, (3) event-based programming which allows interleaved program execution, automatic storage management, and a single run-time stack, (4) multiple processes communicating by messages where task preemption is necessary, and (5) interface construction tools to facilitate experimentation and program refinement" (Dannenberg, 1989, p. 1).

In fact, IMS are (really) difficult to design and implement. Many researchers report this impression, which I have experienced myself during the design and implementation of RGeme (p. 141) and iMe (p. 169). Given the experience obtained with these systems, the following Sections will focus separately on each of the questions I consider the most relevant to achieve outstanding interactive musical systems.

5.1 Improvisation, Composition, Musicology (What For)

Several systems have been developed with one eye (or both eyes) on improvisation. Just to mention a few examples, GenJam is an "interactive genetic algorithm that models a jazz improviser" (Biles, 1998, p. 1). In Blackwell's Swarm Music, "human improvisers interact with a music system that can listen, respond and generate new musical material" (Blackwell, 2006, p. 1). Jorda's PITEL is "a software environment for polyphonic real-time composition and improvisation" (Jorda, 2001, p. 3). Collins "describes an improvisation simulation for human guitarist and four artificial performers" (Hsu, 2007, p. 368). Francois et al (2007b, p. 277) introduce MIMI, "a multi-modal interactive musical improvisation system". Each one of these systems considers improvisation with a particular theoretical, technical and/or practical issue in view. The fact is that improvisation is an area of music making which IMS have intensely explored.

Some researchers raise the question of how the use of IMS could be applied to composition. Comparisons are often drawn between the traditional thoughtful structured engagement normally accredited to music composition and the acknowledged freer improvisation practice. Rowe, for instance, referring to his composition *Maritime* mentions that

> "both the human and computer players are left areas of freedom to add their own commentary to the ongoing elaboration, producing a composition stable in its overall development, but with different realizations at each performance" (Rowe, 1999, p. 87).

Composers often control the final musical outcome, forecasting and minimizing as much as possible the "liberties" taken by improvisers in ways that the actual performance is within a pre predefined aesthetic model. At the same time, they take advantage of the typical unpredictability of an improvisational setting in order to extend the limits of the composition. A dichotomy between composition and improvisation seems therefore to be widespread in this field. IMS could interconnect these two realms, according to Rowe:

> "Interactive music systems derive control parameters from an analysis of live performance, they can generate material based on analyses of improvisation as easily as they can on analyses of notated music. Such systems become a ligature connecting improvisation to notated composition, as the same processes used to govern the notated music can be employed to generate new improvisations in real time (Rowe, 1999, p. 85).

Even though musicology research has not yet been systematically explored by IMS, there are potentially many applications where IMS would be an appropriate strategy, especially for the exploration of cultural evolutionary models (Miranda *et al.*, 2003a; Coutinho *et al.*, 2005). Humdrum (Huron, 2002b) is a set of tools for musicological research but not an interactive system in the sense described in this Thesis (p. 93). iMe (Gimenes *et al.*, 2007b) focus on this area within the ontomemetical model proposed in this Thesis (p. 125) by allowing the development of music styles in societies of artificial agents.

5.2 Time and Space Awareness

If one really wishes to have a truly meaningful musical experience, especially within improvisation, time and space awareness are important issues that must be addressed. The literature often refers to these questions under the expressions "beat tracking" and "score following".

Starting with the former, in general terms, you must know "where you are" in a particular piece but also where the other musicians are in order to be able to "be together". The score is the road map on top of which musicians add their own particular musical intentions (interpretation), regardless of the fact whether the piece is being played by memory or not.

The score is always present even though it may not physically exist. If something unexpected happens (such as a lack of memory), certain techniques can help to retrieve the correct position in the score. One of these is to define and memorize places ("points of reference"), normally related to the music's overall structure, to which to return. If the piece is not played from memory, the score is physically in front of the musician to be followed. Again, if there is any problem during the performance, the musician can adapt to this new situation by going back and forth specific points of reference and continue thereafter.

Usually unconsciously, listeners do the same sort of thing. People tend to follow the "score" even though they never actually read it. We have a predisposition to group musical elements together and organize them in structures of different levels of organization. The result is that if we know a piece of music and hear it again, we will try to "find ourselves" along the new performance. If we do not know the piece beforehand, after a new performance we will have in our memory a representation to which we will refer in new performances.

Playing with other musicians presents some additional interesting challenges. Every musician has a mental representation of the piece and will perform his individual part, adding his own interpretations but, because the other musicians necessarily have their own individual representations, they all must be synchronised so that the result is an interconnected (and not chaotic) whole. Again, if anything "goes wrong" (e.g., lack of memory), a particular musician will be able to catch up with the others in the earliest occasion by figuring out where in the score the others are and restarting the performance from this place.

It is easy therefore to understand how important "score following" is for an IMS if one wishes to establish a truly meaningful collaboration between man and machine. Quite easily human beings can adapt to new situations such as when the machine plays a new and/or unexpected musical material. The same cannot be said if the human performer tries new directions experimenting with new musical structures. The machine will not follow if not programmed to do so. These situations can happen potentially numerous times, especially when dealing with improvisation settings. Both, man and machine must be listening to each other in real-time, assessing their positions in the overall structure of the piece and adapting to any unpredictable new situation.

"Beat tracking" is another very important issue and is intimately connected to "score following". Music involves the production of sounds and sounds, by definition, only exist in the time domain. Hence, in addition to the ability of knowing "where you are" during a performance, you must also know how to play with the others by adapting and synchronising your internal clock to the internal clock of the other participants.

For human beings synchronising is apparently very easy to do. We all possess an internal pulse that roughly corresponds to a constant given musical duration. This duration could correspond, for instance, to a crotchet or a quaver. All the other possible musical durations are obtained by comparison with this duration. If we want an even distribution, for instance, we simply apply division/multiplication factors (e.g., 1 crotchet = 2 quavers). If we want to reinforce some stylistic aspect, such as what happens with swing quavers in jazz, a different time grid can be applied (see Figure 5, below). Speeding up the reference pulse speeds up the execution of the piece as a whole and vice-versa, slowing down the pulse slows down the execution of the piece.

Figure 5: Jazz swing quavers.

It is usually hard for human beings to keep a constant and steady pulse; for numerous reasons, sometimes we tend to accelerate or to slow down during performance. Of course, *accelerandos* and *ritardandos* are stylistic resources that help to make the performance more attractive and human beings playing together are able to adapt to each other if the group (or group leader, etc.) starts to accelerate or to slow down. With machines it is exactly the opposite: they uninterestingly keep the pace and only accelerate or slow down if programmed to do so.

Pioneer researchers in this field are Vercoe (1984) and Puckette (1982). Schwarz (2003) prepared a commented bibliography, which was recently updated by Orio (2003).

5.3 Modes of Interaction (When To Play)

Generally speaking, during a performance performers adopt different approaches with regard to what and when they play. In a very basic scenario, for instance, monophonic instruments (e.g., flute, oboe) playing solo have a one-to-one relationship with the voice (or part) they play. These instruments cannot play multiple parts but, of course, a polyphonic instrument (e.g., piano, guitar) can.

Being a polyphonic instrument and having an extraordinary potential to reproduce a great number of sonorities, the acoustic piano is a particular case in which performers often tend to take specific approaches (dynamics, touch, etc.) in order to stress different voices and textures. Very often as an additional stylistic resource, pianists try to "reproduce" certain sonorities of other instruments, which are not necessarily particular to the piano. This is the case, for instance, of bass quavers or crotchets in Bach's keyboard music, where pianists use the metaphor of the cello sound while playing long bow movements.

Within group performances, instruments often change their roles. A melodic element, for instance, could initially be played by the oboe, and later be joined by the flute or by other instruments, the whole sequence being the exposition of a melodic theme. This interplay of voices and instruments helps to shape musical texture, density and orchestration.

Many times an instrument plays the role of an "individual", embodying personality and character, such as the duck, played by the oboe, or the bird, played by the flute in *Peter and the Wolf*, by Sergei Prokofiev. If we expand the concept of "individual" or "individuality" to musical parts, it could be possible to make some apparently strange assertions such as saying that the duck uses the oboe to represent itself or to express its voice. And the bird uses the flute to represent its voice. It could as well be possible to say that the main persistent theme in Ravel's *Bolero* could be regarded as some sort of entity that uses the orchestra to introduce itself and to slowly expand into different forms and shapes that ultimately lead to the climax at the end of the piece.

It may seem unusual to consider that musical voices have will or self-awareness. Of course, they do not, but let us consider another example. Going back to musical textures and orchestration, it is the composer who traditionally decides about the distribution of voices among the different instruments and it is the conductor that traditionally decides about the group performance. We know, however, that the equilibrium of the group does not depend on the composer and/or the conductor alone. The equation is much more complex, relying on an intricate interplay between score, conductor and performers, their ability to understand the musical material and to collectively "negotiate" means and resources to deliver the musical message. In other words, it is possible to imagine the existence of a metaphorical "creative mind" distributed among the entire group.

Looking at improvised music we observe that musicians can be left more or less freedom to develop their own ideas. There can be fully written arrangements or completely free schemes (free improvisation). Of course, as there is often no black or white but different shades of grey, it is more appropriate to say that different degrees of improvisation can be obtained by having more or less "fixed" references to follow during a given performance. The more common case, however, seems to be the one (p. 42) in which "each musician has a high-level, partial representation (schema) of the piece which is being played" (Murray-Rust *et al.*, 2005, p. 2).

From all of the musical activities mentioned above it seems that collective improvisation is the one where the "when" issue really represents a problem for IMS designers. There is not a sole "centre of control" as it is the case of the composer or arranger but several minds engaged by a creative activity that act at the same time and which must somehow negotiate their participation in the collective performance. These initial considerations contain the rationale behind what can be called "modes of interaction" and to which the literature also refers to as processes, modalities or protocols of musical interaction.

Miranda calls the interplay between interacting agents as "the enacting script", which "provides the agent with knowledge of how to behave during the interactions: the agent must know what to do when another agent produces a tune, how to assess an imitation, when to remain quiet, and so forth" (Miranda *et al.*, 2003a, p. 95). Paine mentions that each participant must constantly monitor the others and use "their interpretation of the other parties input to make alterations to their own response strategy, picking up points of personal interest, expanding points of common interest, and negating points of contention" (Paine, 2002, p. 4).

As mentioned in the previous Section (p. 101), human beings naturally adapt to a changing environment. However, the same type of behaviour is very difficult for an application. An IMS should decide when to intervene for which, of course, some sort of real-time analysis (listening and deciding) must exist (Rowe, 1996). Referring to this subject, Dubnov explains that:

> "One of the main challenges in producing a larger form or improvisation of significant span is in creating "handles" or means of control of musical generation, so that the result becomes more then accidental play of imitation and response" (Dubnov *et al.*, 2005, p. 1).

Distributed intelligent agents are often used as paradigms when dealing with this issue. Wulfhorst and colleagues , for instance, mention how some of the agent's characteristics could be translated into different behaviours during an improvisation. E.g., an agent could "lead" the others (by stimulating them to adapt to it), or be "flexible" (by allowing adaptations to meter changes). A "persuasive" agent would "gradually" try to return to its tempo or an "improvising" agent would propose harmonic changes. Murray-Rust mentions the modelling of goals between agents "to explore the work at hand and achieve a degree of commonality between the schema of different agents - this is "the direction of the exchange in which you are engaged" (Murray-Rust *et al.*, 2005, p. 2).

Pachet mentions several types or protocols of communication, "rules on which the system decides to play": question-answer, lectures, small talk, exams, baby talk, etc. (Pachet, 2006a, p. 6). Collins describes the use psychological states (e.g., shyness, sloppiness, keenness) as controls for agents in free jazz simulations (Collins, 2006).

Some of the possible modes of interaction are discussed in the following paragraphs.

Conversation Analysis

In order to address the "modes of interaction" issue, one of the theoretical models proposed by IMS designers is "conversation analysis" which studies how control is passed from one participant to another during conversation. In "music conversation", the participant should infer what to do (e.g., solo, accompany, duplicate another voice) at any given moment.

Winkler describes the human conversation as a "two way street" where two people share words and thoughts and are interchangeably affected ((Winkler, 2001) in (Paine, 2002, p. 4)). Some of the archetypes he mentions are the "conductor model", where the components are centrally controlled, the "chamber music model", where control is distributed sequentially from one component to another and the "improvisation model", where control is shared among the components in a pre-defined framework.

The system ImprovisationBuilder uses conversation analysis techniques (Walker, 1997): the improvisation task is functionally decomposed into listening to the ongoing musical stream, composing new utterances and realizing these new utterances by identifying any changes of roles of the other participants and adapting accordingly. In this system, the "listener" component determines what the human musician is doing (soloing, accompanying, etc.) by assessing the pitch range, density and harmonic content of the notes being played. This analysis enables the system to adapt to the ongoing state and react accordingly.

Theory of Musical Acts

The theory of "speech act", which belongs to the field of linguistic pragmatics, is another paradigm adopted by some systems. It views the natural language utterances as actions (e.g., statements, questions); to say something is to do something (Austin, 2006).

Murray-Rust adopts this approach in MAMA (Musical Acts - Musical Agents), where agent interactions are structured under acts such as agreements, calls for proposals, etc. The aim is to "create a society of agents which can improvise music with each other, and interact with humans (either musically or through a higher level interface)" (Murray-Rust, , p. 2).

Continuity

Pachet uses the concept of "continuity" in the Continuator, where interaction is expressed not by a "dialog" between man and machine but rather by the idea of man and machine being a single entity. The system is able to detect when the performer stops playing (phrase endings), continue whatever the performer had started to play and stop playing as soon as the performer wants to play a new phrase. In Pachet's words,

> "... the Continuator plays an infinite stream of music in some learned style (jazzy chord sequences, for example). The user plays a melody and the Continuator tries to adapt its generation to the user's input in real time" (Pachet, 2004, p. 3).

Interconnected Musical Networks

Weinberg (2005, p. 23) introduced a theoretical framework for musical interaction called "Interconnected Musical Networks" (p. 108), which "attempts to define and classify the aesthetic and technical principles of interconnected musical networks". In this model, taxonomy is based on combinations of sequential-synchronous operations and centralized-decentralized control schemes.

In sequential decentralized interactions, for example, "players create their musical materials with no outside influence and only then interact with the algorithmic response in a sequential manner" (Weinberg, 2005, p. 34). In a "synchronous centralized network topology, on the other hand, players modify and manipulate their peers' music in real-time, interacting through a computerized hub that performs analysis and generative functions" (Weinberg, 2005, p. 28).

Looking at the possibilities for interactivity on the Internet, Weinberg defines four generic approaches. The most basic one, the "server approach" only uses the network to send musical data to disconnected participants. The "bridge approach" views the network as a means (technical support) to connect the participants as if they were in the same space. Within the "shaper approach" the system generates musical material that would be transformed by the participants. At last, the "construction kit approach" would allow musicians to perform in multiple-user composition sessions, manipulating each other's musical material. Weinberg distinguishes between "small-scale local systems" (three to ten participants) and "large-scale local systems" (more than ten participants).

Based on the "Interconnected Musical Networks" paradigm, Weinberg developed a number of interaction modes for Haile, his robotic rhythmic system. From the perspective of "when to play" - the subject of this Section - the system can play sequentially (human and robot one after the other) or synchronously (human and robot together). With respect to the first possibility, the basic mode is "imitation", in which case the system simply imitates what it hears. In "stochastic transformation", it transforms what it hears by stochastically dividing, multiplying or skipping certain beats. In "perceptual transformation", Haile chooses and plays musical material that has similar levels of stability to what it hears.

Synchronous interaction can be done because Haile is able to detect the beat and tempo of the input and lock to it. In this mode, the system can play in "simple accompaniment" by performing previously recorded MIDI files. Finally, in the most sophisticated mode, "perceptual accompaniment", Haile is able to combine synchronous, sequential, centralized and decentralized operations:

> "Haile plays simultaneously with human players while listening to and analyzing their input. It then creates local call-and-response interactions with different players, based on its perceptual analysis. ... While Haile plays short looped sequences (captured during Imitation mode) it also listens to and analyzes the amplitude, density, and accuracy of human playing. At periodic intervals it then modifies its looped sequence, using the amplitude, density, and accuracy coefficients analyzed from human playing ... When the rhythmic input from the human players is dense, Haile plays sparsely, providing only the strong beats and allowing humans to perform denser solos. When humans play sparsely, on the other hand, Haile improvises using dense rhythms that are based on stochastic and perceptual transformations" (Weinberg *et al.*, 2007b, p. 101).

5.4 Musical Structure (What To Play)

Interactive musical systems must be able to decide when to play as well as what musical material to play. Some systems (p. 57) adopt an entirely algorithmic approach in regard to musical generation while others learn from musical input. Learning can be done from (non real-time) previously prepared material or from real-time input. Learning also raises the question of how the musical database for potential interactions should be controlled. In Francois' (2007b) system, for instance, a preparation phase precedes the performance itself. During preparation, the system learns the musical material that will be used for generating improvisations.

Pachet (2002b) introduces a number of possible modes that affects the "what to play" issue. In the simplest one, "autarcy", the system does not have any previous memory and learns progressively as it hears the music that is performed by the human musician. In "virtual duo" mode, the Continuator starts the performance with music loaded into the system's memory. "Contextual continuation" is more complex in which the system is connected to two musicians, learning from one of them but continuing phrases produced by the other one. "Playing twice with oneself mode" consists of two phases. During the first one, the system learns from a "harmonically rich" musical input. In the second phase, the system generates a continuous stream of music based on what it learned before and then the human musician improvises on top of what the system generates. At last, in "swapping mode", a different version the system is connected to several musicians. Memories are swapped and what results are continuations of one musician following improvisations of another musician.

In ImPact, Ramalho (1997) adopts the idea of "Potential ACTions" (PACTs) that correspond to instructions for the creation of melodic phrases based on parameters such as dissonance and density. The agents' memories are initially fed with previously prepared melodic segments from human performances. Agents analyse the musical input and establish "temporal segments" that will be used in generation mode to decide (based on a series of PACTs) which segments will be retrieved to compose the melody.

5.5 Interface Design

Looking back at the history of traditional music, it is not difficult to realize how the evolution of musical instruments contributed to the evolution of musical aesthetics. From the earliest recorders to the modern saxophones, from the rudimentary drums to the modern vibraphones, a continuous succession of additions and improvements helped to define music, as we understand it today.

In this equation, numerous elements come into play such as materials, shape, size and the way sound is produced, among many others. All these features together belong to the field of instrument design and construction and inevitably suggest another element that is intrinsic to them, the interface. In simple terms, interface is where the instrument "touches" the player (or vice-versa), or the elements in the instrument that afford interaction with or control by the player.

Interfaces must be understood and mastered if one wishes to achieve "optimal musical results", an appropriate musical output, given compositional and aesthetic contexts. The collection of techniques used to master the instrument is what we call "instrumental technique".

Traditionally, musical instruments, technique and aesthetic values evolve together and it is not uncommon that the development of one of these elements pushes another one towards completely new boundaries. That happened, for instance, during the Romantic period when the improvement of technology applied to the construction of instruments, allied to the appearance of virtuosos (e.g., Paganini, Chopin, and Liszt) allowed the arrival of new sonorities.

The same must be said in regard to the music transformation (not to say revolution) during the 20th century, especially after the advent of the so-called computer age. The traditional cause and effect paradigm typical of acoustic instruments gave its place to a new notion where the user action on the interface produces all sorts of different and even unexpected results (meta-instruments).

Of course, the use of interfaces as simple as a mouse or a computer keyboard could be regarded as an invitation to a broader audience to come and to practice music. People not interested in spending thousands of hours trying to control a specific instrumental technique (such as it is often the case with traditional instruments) could use day-to-day gestures (mouse clicks, key presses, etc.) to make music. Without disregard to the merits of assembling larger audiences, the drawback of this approach is that some simplistic solutions adopted by a number of systems do not necessarily improve the sense of fulfilment typical of music making activities.

Instrument design and interface necessarily deal with this type of compromises. One can at the same time obtain an uncomplicated interface but miss a desired sonority. The way technology today overcomes this complex dilemma is by providing means by which simple gestures (again, the click of a mouse button, etc.) can be interpreted and expanded through computer algorithms. In contrast, computer algorithms can re-interpret and expand music under the scope of traditional instrument and music-making contexts by assigning new meanings to musical gestures that would not be normally captured by these instruments.

In any event, the advent of all these new technologies and musical interfaces raises new and crucial questions. Jorda inquires: "what is a good music instrument? Are there instruments better than others?" (Jorda, 2003, p. 1). I would like to add a few other questions, myself: what should be the most appropriate instrument technique to apply in each specific case? To which extent traditional instrumental techniques could be applied to new interfaces?

Some questions concerning interface design are addressed by the issues discussed in the next paragraphs.

Independence of Operation

The degree of autonomy or independence regarding external control is another interesting issue that affects the design if IMS. One must consider whether there should be a human operator in control of some of the systems parameters during performance. In OMax, for instance,

> "...the improvising musician interacts with the system based on purely aural feedback, while another human operator controls the machine improvisation (the improvisation region in the oracle, instrumentation, etc.) through a visual interface, during performance" (Francois *et al.*, 2007b, p. 278).

Visual Cues and Feedback

Should the system incorporate some sort of visual cues during performance? Pachet understands that users "engaged in creative music-making cannot afford (to) have their attention distracted from the instrument to the computer" and that "all the interactions with the system should be performed only by playing" (Pachet, 2006a, p. 6). Jorda even mentions that "visual feedback is not very important for playing traditional instruments, as the list of first rank blind musicians and instrumentalists may suggest" (Jorda, 2003, p. 3).

However, I am not convinced this is the best approach. Contrary to Pachet's assertions, body movements, visual cues, among other forms of manifestations, are an integral part of human performance. They help communication with the audience as well as among interacting musicians. IMS could therefore take advantage of this type of expression to enhance interaction between humans and machines.

In a typical scenario, communication among musicians happens all the time. Just to mention one example, in jazz settings,

> "...verbal and visual cues are used to coordinate accompaniment and solo. Each solo usually takes several choruses, allowing the soloist time to explore several musical ideas or elaborate on the melody. At some point during the solos, one of the players may raise four fingers to propose "trading fours," a standard procedure for exchanging shorter improvisations amongst the musicians" (Walker, 1997, p. 125).

Weinberg also advocates that interactive systems should provide some visual cues or feedback to help players coordinate their performance: He asserts that IMS are

> "...hampered by their inanimate nature, which does not provide players and audiences with physical and visual cues that are essential for creating expressive musical interactions. For example, motion size often corresponds to loudness and gesture location often relates to pitch. These cues provide visual feedback and help players anticipate and coordinate their playing". (Weinberg *et al.*, 2007b, p. 97).

Francois claims that "visual feedback of future and past musical material, and of high level structural information, provides timely cues for planning the improvisation during performance" (Francois *et al.*, 2007b, p. 278). Mimi, an interactive musical improvisation system was developed with this concern in mind: Francois divided the screen in to parts that exhibit "the musical material to come, and recently passed, in real time" (Francois *et al.*, 2007b, p. 278).

5.6 Music Representation

Music comprises a multitude of sound events distributed in time and, in order to be handled computationally, it must be efficiently represented. In fact, music representation is a critical issue for IMS. Addressing this issue, Cambouropoulos remarks that "finding a balance between structural hierarchic and linear dynamic aspects of musical understanding seems to be a particularly difficult task" (Cambouropoulos, 1998, p. 25).

Wiggins (1993) carried out an extensive study on this subject. According to him, an optimal solution should be a

> "logical specification of an abstract representation. The representation should be abstract in that it should not commit the user to any particular (cultural and stylistic contexts) representation of, say, pitch in terms of a frequency in Hertz or some particular note name. On the other hand, the representation of pitch should allow the determination of intervals from pitches, and, in general, should permit the user to determine and manipulate all the musically meaningful aspects of pitch" (Wiggins *et al.*, 1989, p. 1).

Even though numerous works have been dedicated to this subject, it seems that a universal solution is far from being generally accepted, all sorts of computer systems adopting their own way out. In general, each system tends to adopt a tailored specific solution for the task at hand. CHIME (Franklin, 2001) for instance, represents melodies (pitches and rests) "with sixteenth-note durations (slurs are needed to create longer notes)" (Thom, 2003, p. 37). The Continuator uses a more comprehensive scheme representing pitch (integers between 0 and 127), velocity/amplitude (idem) and timing information (long integers) (Pachet, 2002c).

Huron's system Humdrum comprises a set of (numerous) tools for musicological research (Jan, 2004). In the "Frequently Asked Questions" (FAQ) file that accompanies the system's documentation, Huron mentions an astonishing number of representations that are predefined in Humdrum (e.g., pitch, frequency, MIDI, cents, semitones, pitch-class) (Huron, 1994). Huron suggests that users should add their own representation schemes, acknowledging that there are potentially endless aspects in music that can be computationally manipulated.

The Musical Instrument Digital Interface (MIDI) plays an important role in this field. The MIDI protocol was created in 1983 to address communication issues between electronic musical devices (Huber, 2000) - especially keyboards - and even though not being particularly suitable for music representation, many systems use its features to represent and store musical information.

Its main focus and philosophy were largely based on the idea that music communication could be achieved with relatively few performance elements or "event messages" (time information, note number, velocity, etc.). More than 20 years after its inception, MIDI is still successful in commercial terms due to its simple and inexpensive interface but has lost its *allure*, in particular among IMS researchers mainly because of its inability to address features that do not belong to the keyboard family of instruments.

MIDI would not be appropriate to tackle controls such as the ones belonging to the bow instruments. A cellist, for instance,

> "...has available roughly four degrees of freedom: bow force, bow velocity, bow distance from the bridge, and finger position on the bridge. Each one of these control dimensions affords the player some degree of control over various parameters of the sound: bow force controls dynamic, for example, while the finger position on the bridge is primarily responsible for pitch" (Rowe, 1996, p. 53).

Open Sound Control (OSC) (Wright *et al.*, 1997; Wright, 2005), another protocol for communication among multimedia devices and based on networking technology, is a more flexible, powerful and up to date alternative compared to MIDI.

In a parallel area, MusicXML (Good, 2006) is a standard for interchange of symbolic music among applications that is rapidly achieving widespread acceptance. Because of its flexibility, it is also being regarded as a possible *lingua franca* for music representation and storage purposes, among many other applications.

The representation of polyphony is a special case. Among the propositions so far presented we find the ones introduced by Assayag's team of researchers (1999) that process a polyphonic stream by slicing the music information at every possible boundary. "Slicers" and "unslicers" functions are responsible for exchanging between the model representation and MIDI. More details of this representation can be found in (Dubnov *et al.*, 2003). Pachet (2002c) also adopts a similar scheme.

Within human performance, music representation becomes more complex. Human performances are "expressive". Often, the kind of precision that human players are used to or able to perform with regard, for instance, to the moment of attack of each individual note in a chord is not (luckily) the same as the type of precision that computers can produce. Because of its reductionist quality, even though musical scores define that chordal notes should be played at the same time, for stylistic reasons, the music's dramaturgy or simply because of the lack of accuracy in human performance, they are often not played simultaneously.

To address (and avoid) these idiosyncrasies, reductionist functions ("pre-analytic simplification") are sometimes suggested which, of course, result in loss of musical data. Pachet aggregates clusters of notes that sound approximately together. On the other hand, notes slightly overlapping are not considered to belong to the same "cluster" but indicate a legato style (Pachet, 2002c).

Lartillot (2001) and Dubnov (2003) reported the development of a toolkit containing five simplification filters that address specific issues. They propose that notes that are attacked or released nearly at the same time should be vertically aligned (arpeggio and release filter), overlapping or silence between successive notes be removed (legato and staccato filters) and durations be statistically quantized (duration filter). Fixed thresholds could be applied to control the operation of these filters.

Conklin and Witten (1995, p. 51) propose "a multiple view-point system, a collection of independent views of the musical surface each of which models a specific type of musical phenomena". Pachet (2002c) uses a function (PitchRegion) that simplifies pitch information by considering the region of its occurrence.

5.7 Languages and Environments

So far, I have demonstrated that implementing interactive musical systems is not an easy task. In addition to all the elements mentioned in the previous Sections, IMS are not well supported by general-purpose languages such as Lisp, C++ or Java even with support of music libraries (Dannenberg, 2002). Hence, it is not surprising that a number of specific tools, languages and environments have been designed.

Max Mathews, one of the pioneers in the world of computer music systems, introduced a series of developments starting with Music-I, created in 1957 at Bell Labs, often mentioned as the very first computer application for the generation of digital audio waveforms through direct synthesis. Others followed until Music-V appeared in 1968, written in Fortran, which meant that it could run on any computer with an appropriate compiler.

With these systems, Mathews addressed some important issues, namely, the specification of functions to describe sounds ("unit generators") and to represent a sequence of sounds ('notes'). A unit generator (oscillators or envelope generators) can receive input control and produces sound. Sound objects can be combined to create instruments or patches. Amatriain explains that a

> "...traditional Music-N orchestra file is very similar to a program source code. The score initializes the system (with information usually contained in the header) and then contains a list of time-stamped notes that control the different instruments" (Amatriain, 2004, p. 127).

Other systems followed the experience started by Mathews and are considered to be "heirs" in the Music-N family, such as Common Lisp Music, CSound, JSyn, Max, PureData and SuperCollider (Amatriain, 2004). Written in Common Lisp and created by Bill Schottstaedt, Common Lisp Music is an object-oriented music synthesis and signal framework that supports sound synthesis (Schottstaedt, 1994). Components are generators, instruments, list of notes, tools for sound file input/output and graphical interface.

In 1985 Vercoe created Csound, an application used for sound design, music synthesis and signal processing that implements features for composition and performance (Vercoe, 1986; Boulanger, 2000). The philosophy is very similar compared to the previous systems: one can program a text file (orchestra file) with the instructions on how to build musical instruments. This can be done by combining from 1300 "operational codes", the basic building blocks that, together, define the patches that are then translated into a machine-resident data-structure, the instrument. Sound is then generated from instructions (list of note events and other data parameters) contained in a second text file (score file), a MIDI file or real-time controllers (MIDI, computer keyboard, mouse, etc.).

Originally developed at IRCAM by Miller Puckette in the mid 1980s and named after Max Mathews, Max has been used for over twenty years and can be said to be one of the most popular environments for the development of IMS to date. A first commercial version was released in 1990 by Opcode Systems, Inc. and the current version, Max 5, which includes Max, MSP and Jitter objects, is maintained by David Zicarelli's company, Cycling '74 (Cycling'74, 2008). MSP adds signal processing functionalities and Jitter implements video, matrix and 3D graphics objects for Max. *Pluton*, composed by Philippe Manoury in 1988 is known to be the first piece to use Max patches and countless others have appeared ever since.

Puckette describes the "Max paradigm" as

> "...a way of combining pre-designed building blocks into configurations useful for real-time computer music performance. This includes a protocol for scheduling control and audio sample computations, an approach to modularization and component intercommunication, and a graphical representation and editor for patches" (Puckette, 2002, p. 1).

Programmes in Max are called "patches" and can be implemented within its graphical user interface by connecting (input and output of) objects that represent all sorts of functionalities, creating a flow of control. It is very modular and opened to the development of "external objects" by third parties. For these reasons, a legion of users (composers, performers, researchers, etc.) and developers has adopted Max as their main platform for the implementation of IMS.

Puckette (1996) was also responsible for the development of Pure Data (Pd) in 1996, an open source programming environment for real-time audio analysis, synthesis and processing. It has many commonalities with Max/MSP, especially the manipulation of graphical objects under a data flow paradigm. Main differences are the possibility to be used for simultaneous computer animation and audio (using Mark Dank's external GEM - Graphics Environment for Multimedia) and facilities for defining and accessing data structures. As well as Max, Pd is opened for the creation of modular code in the form of "externals" by third party developers that can be distributed as standalone applications. It can also be used to create network (e.g., Internet) systems for live collaboration

An environment and programming language at the same time, SuperCollider was originally created by McCartney (1996; 2002) for real time audio synthesis and algorithmic composition and is considered one of the best-known systems for interactive audio. The series of environments and programming languages for the development of real-time audio systems introduced by Dannenberg must also be mentioned: Arctic (1986), the CMU Midi Toolkit (1996), Aura (1996), Nyquist (1997), and Serpent (2002).

Summary

This Chapter introduced the main elements that have emerged in my research as important for the design and implementation of IMS and explained how they have been tackled by many researchers.

The "what for" issue concerns the general goals of IMS, which directly affects some of the decisions about the other issues. "Time and space awareness" is associated with the spatial localization within music and the synchronization with other musicians in real-time. Modes of interaction deal with the roles of players in real-time interaction. The "what to play" issue regards the musical material manipulated (e.g., perceived, generated) by IMS. Interface design affects instrument technique associated with IMS, the independence of their operation, visual cues and feedback. Music representation is related with how computers manipulate musical variables, something that, given the lack of a universal formula, must be addressed by each system individually. Finally, in "Languages and Environments" the main languages and environment that facilitate the design and implementation of IMS were introduced.

This chapter closes the background for this Thesis; each and every subject studied in the previous chapters is intimately connected with the next chapters. Part I: The Field of Research and Part II: Interactive Musical Systems were structured with the concern of gradually building a critical knowledge on which Part III: Ontomemetical Systems, the core and main outcome of my research, is founded.

The next chapters will formally introduce the ontomemetical model and two interactive musical systems (RGeme and iMe) that were designed and implemented based on this model.

Part III: Ontomemetical Systems

Chapter 6 The Ontomemetical Model

> "To argue about whether nature or nurture is more important for the development of adult intelligence is like asking which contributes more to the area of a rectangle: its length or its width." (Hebb *in* (Scott, 1995, p. 153))

6.1 Ontogeny

The ontomemetical model proposed in this Thesis is greatly inspired by the notions of *ontogeny* and *memetics*. The word ontogenesis comes from the Greek "onto", which means being and "genesis", which means origin. In the sciences, ontogenesis refers to the "development of an individual organism or anatomical or behavioural feature from the earliest stage to maturity" (Anon, 2005b). The term development could either refer to the "embryonic development" of organisms, as in this definition, or, in other contexts, to the development of organisms during lengthier time spans.

In the context of embryonic development, Parncutt (2008) studies the origins of musical behaviour during the prenatal phase, which involves the acquisition of perceptual, cognitive, motor and emotional abilities. This was also the sense adopted by the Theory of Recapitulation, introduced by Haeckel in 1866. Haeckel claimed that the embryonic development of organisms reproduces their phylogeny, hence the statement "ontogeny recapitulates phylogeny". This theory remained discredited during the 20th century but has been recently revisited (Gould, 1977). "Phylon" is the Greek word for tribe, race, and

Phylogeny, studied by evolutionary developmental biology, is the evolutionary path of species as, from the evolutionary perspective, some species would evolve into different species, hybridize or terminate (extinguish).

Minelli explains that ontogeny "is the unfolding of coupled developmental mechanisms whose parameters are largely specified by the genome" (Minelli, 2003, p. 1). Genetic studies started with Mendel's work (1865) on the transmission of physical characteristics of plants. The unit of biological information, which defines a single trait, is called "gene". The ensemble of genes is defined as the genotype while the observed properties of an individual are called the phenotype. Genetic inheritance plays an important role not only in the transmission of physical attributes but also on intelligence and behaviour.

The study of genetic abnormalities, such as the Williams Syndrome, sheds light into the study of the development of individuals. This rare condition is caused by a chromosome alteration that affects the neuronal development of individuals. It is the cause of, among other symptoms, mild retardation and high prevalence of musical abilities. As music communicates affection, this last symptom has been attributed as a protective factor for individuals (Blomberg *et al.*, 2006).

Welch (2000) approaches the ontogenesis of musical behaviour from a sociological perspective and observes that music results from an interplay between "the individual's neuropsychobiological potential and their socio-cultural environment. This interaction constantly shapes musical development and facilitates both cultural transmission and cultural transformation of dominant musical genres" (Welch, 2000, p. 1). The functioning of the human's cognitive system for instance is shaped by socio-cultural factors (e.g., home, school) and associated values (e.g., norms, roles, identities).

There were heated debates about the importance of the roles on the development of an individual's behaviour played by, on one side, inner qualities (e.g., genotype), and on the other side, personal experiences. Today this question is considered outdated, as both factors would be equally important (Scott, 1995, p. 153).

Arshavsky studied the importance of genetic factors and environmental exposure to music to the development of cognitive abilities and suggested that:

> "Mozart became a Mozart at the moment of the formation of the zygote, due to a 'lucky' combination in the variation of genes determining musical abilities" even though one cannot be sure if "Mozart's gift would be fully manifested if his father were not a musician and Mozart had not listened to music from his infancy" (Arshavsky, 2003, p. 329).

Indeed, we can see from these examples how the ontogenesis of individuals contributes to the formation of their musical abilities. In fact, as well as the development of technology and social interaction, among many other factors, human ontogeny also plays a major role in the development of music history or of music ontogeny. If we consider music as a "living organism", something that surpasses the lifetime of individuals, we could imagine an "ontogeny of music", which would be concerned with the study of music development, from its "early stages to maturity".

Expanding these ideas a bit further, musical styles could be studied from the perspective of their emergence, development and termination. They could be compared according to different contexts (e.g., historical, geographical) and grouped into categories and families of styles as to draw a sort of evolutionary path. In this case, we would be dealing with a "phylogeny of music".

In any case, in order to study the ontogeny or the phylogeny of music, it is important to define which elements are being taken into account. Science has already been able to "materialize" Mendel's ideas and objectively describe the inheritance of physical (and behavioural) traits via genetic transmission, making associations between different genotypes and phenotypes. Culture, on the other hand, does not benefit of the "materiality" of the gene model.

6.2 Memetics

Some scholars (Dawkins, 1989) introduced the idea that the biological paradigm could be transposed to the cultural realm; that the evolution of culture would follow principles very similar to the ones that rule the evolution of species. There should be units of cultural information the same way that, in biology, there are units of genetic information. These units of cultural information were called "memes".

Memetic theorists believe that, the same way information patterns evolve through biological processes, memes would evolve through the adaptive exploration and transformation of an informational space through variation, selection and transmission (Gabora, 1997): "just as genes propagate themselves ... memes propagate in the meme pool by leaping from brain to brain via a process which, in the broad sense, can be called imitation" (Dawkins, 1989, p. 206).

According to Dawkins (1989), memes are elements such as fashion and tunes. They could be regarded as "living biological structures" as they exist in the form of particular neuronal configurations (a pattern of "synaptic hotspots") in the brain. Memes would have the potential to move from one brain to another via a variety of external substrates "such as scores, sound waves, and recordings" (Jan, 2004, p. 70).

Gabora (1997) explains that our minds perform tasks on its replication through an aptitude landscape that reflects internal movements and a worldview that is continuously being updated through the renovation of memes. The "memetic hypothesis" is based on the concept that the understanding of sounds takes into consideration the sounds that someone has already produced (Cox, 2001). The process of comparison involves tacit imitation, or memetic participation that is based on the previous personal experience on the production of the sound.

Musical memes would be characterised by their "indivisible particulateness": small, discrete, and self-contained unities that would survive for a large number of generations as to represent units of cultural transmission. "The more concise a musical meme the greater the likelihood that it will survive the travails of replication by imitation intact" (Jan, 2000). Another characteristic is "coequality": to be regarded as a meme, "one must, by definition, isolate the copy/copies - what one might term the coequal/s - of the particle, from which the particle is imitated, or which is/are derived by imitation of the particle" (Jan, 2000). Figure 6 and Figure 7 show examples of musical memes taken from Beethoven's Symphony no. 5 in C minor and Bach's Invention no. 1 for Two Voices, respectively:

Figure 6: Meme from Beethoven's Symphony no. 5.

Figure 7: Meme from Bach's Invention no. 1.

In order to explore a memetics of music, Jan (2004) understands that some conceptual issues must be addressed. In the first place, the "ontological basis of the musical meme" must be defined in analogy with the genotype-phenotype distinction in biology. Secondly, the nature of a "musical meme" should consider the segmentation of the musical flow as to allow their replication in other contexts. Jan understands that this segmentation should mainly be regarded under the "presence of analogous segments of the same or another symbolic stream" rather than "by means of potentially deceptive surface articulations in the symbolic stream" (Jan, 2000). Finally, the "evolutionary dynamic of musical memes" should reveal the "continuous change of musical style over time as a consequence of the differential transmission and survival of mutant memes" (Jan, 2004, p. 68).

According to Jan (2000) music evolves "because of the differential selection and replication of mutant memes within idioms and dialects. Slowly and incrementally, these mutations alter the memetic configuration of the dialect they constitute. Whilst gradualistic, this process eventually leads to fundamental changes in the profile of the dialect and, ultimately, to seismic shifts in the overarching principles of musical organization, the rules, propagated within several dialects" (Jan, 2000).

Even though it is possible to derive many insights from the parallels between genes and memes, the fact is that it is still not possible to open someone's mind and objectively check which neuronal configurations correspond to which cultural concepts. I wonder if science will ever be able to read neurons the same way it is possible to read genes and associate them with specific ideas.

Furthermore, despite what some theorists advocate, it is not clear why cultural evolution should not be strictly viewed from a Darwinian perspective. Indeed, from the cultural perspective, the concepts of transmission and inheritance are understandable but the "survival of the fittest" seems to be a rather forced adaptation of the biological paradigm. It is true that styles emerge, exist during a period of time and then disappear (or merge into other styles, etc.). Nonetheless, this does not necessarily mean that certain elements in music are more "fitted to survive" than others. What survives? Survives to what? Defining these questions from a cultural perspective is something much more complex to achieve.

Another difficult question is the "ontological basis of the musical meme" that, according to Jan (2000), must be defined in analogy with the genotype-phenotype distinction. In biology, genes encode information that is associated to observable properties (phenotype). On the other hand, memes would be "neuronal configurations" (memory) associated to ideas. It seems, however that a musical meme could be at the same time a neuronal configuration and the musical message or the musical structure. The distinction genotype-phenotype is therefore difficult to establish.

The second conceptual issue mentioned by Jan is that the nature of a musical meme should consider the segmentation of the musical flow and that segmentation should mainly consider analogous segments rather than "deceptive surface articulations". It is not clear however why, from Jan's view, surface articulations are "deceptive". In fact, the musical surface could be described as a complex set of parameters, each one contributing to define segmentation, not necessarily taking into account the existence of "analogous segments". We can track the replication of musical memes by searching for similar or analogous repetitions and these repetitions have the effect of reinforcement on human's memories. Nevertheless, human attention and perception is guided not only by the reoccurrence of similar or specific segments but also by the behaviour of the musical flow itself.

In fact, the multidimensionality of musical elements is subject to a continuous process of development that results from an immense number of factors (e.g., individual genetic configuration, personal experiences, social pressures, technical developments). Changes can occur in such ways that music can shift from one distinct style to another. Even though some of the insights (e.g., transmission, inheritance) produced by the memetic model are easier to understand, sometimes, depending on the context, it could be more appropriate to talk about the "development" instead of the "evolution" of music. With these considerations in mind, the words evolution and development are used interchangeably in this Thesis.

6.3 The Ontomemetical Model

Given the issues mentioned in the previous sections, in fact there aren't currently any computational models specifically concerned with the study of musical ontogenesis taking into account the insights provided by the memetic theory. For this reason, this Thesis proposes the design of such a model, called the Ontomemetical Model of Music Evolution (OMME). The word "ontomemetical" borrows the prefix "onto" from the word ontogenesis and combines it with the suffix "memetical", derived from memetics. The general proposition is to have a paradigm under which computer systems could be designed to artificially explore the evolution of music centred on the transmission of musical memes and, therefore, on the human cognitive system. The aims of the OMME are to:

1) Contribute to the understanding of natural phenomena such as human perception and cognition by computational modelling,

2) Contribute to the achievement of machine musicianship, and the interactivity between humans and machines.

3) Provide computational tools for musicology mainly focused on theoretical models that explore cultural evolution,

Considering these aims, in the following Section three essential principles are proposed for the definition of OMME systems.

6.3.1 Interactive system

Condition 1: "OMME-based systems are interactive systems".

The concept of interactivity in musical systems was explored in Part II of this Thesis (p. 73), at the end of which I proposed a definition for interactive musical systems: "Interactive musical systems are computer-based systems that, through musical information exchange, have the ability to perceive the environment, analyse and take actions as to alter the surrounding states as well as to change their own internal states". Deriving from this definition, OMME-based systems must include mechanisms for:

a) Music information exchange: music information exchange is at the basis of music interactivity.

b) Perception, analysis and action: music information exchange entails perception, analysis and action.

c) Mutual alteration of states: music information exchange must also allow the alteration of states in the system as well as in the environment it interacts with.

6.3.2 Human faculties

Condition 2: "OMME-based systems regard music as an expression of human faculties".

OMME-based systems must explore theoretical and/or empirical models of the perceptive, memory and creative human faculties because:

a) Music is as an expression of these faculties (Part I of this Thesis), and

b) The foundation of the memetic transmission resides on a memory model (Sections 1 and 2 of this Chapter).

6.3.3 Evolution Tracking

Condition 3: "OMME-based systems must implement mechanisms to evaluate music evolution".

A major aim of OMME-based systems is to provide a contribution to musicology, helping to pave the path to an ontogeny and a phylogeny of music. OMME-based systems must therefore implement mechanisms to evaluate aspects of music evolution, such as the occurrence of ontogenetic changes in artificial and/or in real worlds.

Given these conditions, OMME is briefly defined in the following statement:

"The Ontomemetical Model of Music Evolution is a computer model for the design of interactive systems that regard music as an expression of human faculties and provide creative models for the exploration and understanding of music evolution".

6.4 Current Models and the OMME

This Thesis reviews a substantial number of computer systems that deal with different aspects of music making, especially with music interaction. To the best of my knowledge, however, no computer systems have ever been designed to comply with the OMME's principles.

6.4.1 Condition 1: Interactive systems

Indeed, the vast majority of interactive systems do not comply with the definition of interactivity adopted in OMME's Condition 1. Cypher and Voyager, for instance, implement the exchange of musical information with the environment and have sensory and analytical modules that guide actions and/or generative modules but do not have any reported memory modules. On the other hand, systems such as Continuator, Band-Out-Of-The-Box, ImprovisationBuilder and OMax, implement memory models that are transformed by interacting with the environment. EMI is not an interactive system.

6.4.2 Condition 2: Human faculties

The second Condition of the OMME is rarely adopted by any musical system. The General Computational Theory of Musical Structure - GCTMS (Cambouropoulos, 1998), for instance, takes into consideration the principles espoused by Gestalt psychologists. GCTMS is however not aimed at describing music from a developmental point of view and it is not clear how it could comply with the other OMME's conditions.

Cypher and Voyager implement interactivity with intelligent agents under Minsk's Society of Mind where each agent is responsible for the execution of separate tasks such as the analysis of incoming musical events and the production of new musical material. Agents are at the same time collaborative and complementary. It seems however that this approach does not comply with OMME's objectives for a number of reasons. For instance, in these systems the result of perceptive-like algorithms must be hand-mapped to generative algorithms. This approach enables the exploration of a universe of sonorities but does not correspond to the ways human beings cognitively process music. In fact, there is no indication in the current literature that the human mind works as in Minsk's Society of the Mind, i.e., as a society of agents, each individually responsible for the execution of simple tasks (p. 63).

Continuator and OMax have memory modules but, despite the fact that these systems achieve impressive results in reproducing musical styles and in interactivity, these modules do not focus on modelling the human cognitive system

6.4.3 Condition 3: Evolution Tracking

Finally, according to OMME's Condition 3, systems should provide some method to evaluate music evolution, the interest being, among other things, to track how interaction affects their knowledge. None of the systems mentioned above implement mechanisms that evaluate music evolution, however.

OMax and Continuator have a memory model that could be used as a parameter to track music evolution but their authors have not explored this possibility. In Cypher and Voyager this possibility does not even exist; in these systems the agents' outcomes could not be taken as references for comparison because they perform different things. Intelligent agents (p. 63) are an interesting paradigm for the implementation of interactive systems but they are interesting precisely for the fact that they could model interactivity in a society of similar entities and, therefore, whose behaviour could be compared with each other. Consequently, from an OMME perspective, an appropriate approach would be to model agents as single entities with multiple abilities. In this case, one could objectively study which aspects of the environment influence the agents' behaviour by comparing different agents' outcomes.

In order to make an objective analysis of how the above-mentioned musical systems compare to the OMME, six essential features have been identified which summarize the aims and conditions of this model: (i) the existence of musical information exchange within the system, (ii) the concern with the achievement of machine musicianship, (iii) the implementation of perceptive, analytical, and active modules, (iv) the ability to alter the internal states of the system (memory), (v) the adoption of human (perceptive and cognitive) faculties modelling, and (vi) the implementation of musicological tools to evaluate musical evolution. Table 2 shows a comparative table of seven computer musical systems in relation to these features:

#	Feature	EMI	Cypher	Voyager	ImprovisationBuilder	Band-out-of-a-Box	Continuator	OMax
1	Musical information exchange	Y	Y	Y	Y	Y	Y	Y
2	Machine musicianship	Y	Y	Y	Y	Y	Y	Y
3	Perception, analysis, action	N	Y	Y	Y	Y	Y	Y
4	Alteration of states (memory)	N	N	N	Y	Y	Y	Y
5	Human faculties (perception, cognition)	N	N	N	N	N	N	N
6	Musicology (musical evolution)	N	N	N	N	N	N	N

Table 2: Comparative Table of Musical Systems
(Y = Yes, N = No)

Summary

In this Chapter, a computer model for the design of interactive systems called the Ontomemetical Model of Music Evolution (OMME) is introduced, inspired by the fields of musical ontogenesis and memetics.

Ontogenesis refers to the development of an organism from the earliest stage to maturity. Among many other factors, the development of technology and social interaction play a major role in the development of music ontogeny. Memetics is a cultural analogy to genetics. In biology, the unit of information is called "gene" while in memetics, it is called "meme". Memes would evolve through the adaptive exploration and transformation of an informational space by means of variation, selection and transmission.

There is no computational model specifically concerned with the study of musical ontogenesis taking into account the insights provided by the memetic theory. The OMME is a new paradigm under which computer systems could be designed to artificially explore the evolution of music centred on the transmission of musical memes and, therefore, on the human cognitive system.

OMME-based systems must comply with the following three general principles:

1. OMME-based systems are interactive systems,
2. OMME-based systems regard music as an expression of human faculties, and
3. OMME-based systems must implement mechanisms to evaluate music evolution.

To the best of my knowledge, no computer systems have ever been designed to comply with the above-mentioned principles. The next Chapters introduce two systems of my own design - RGeme and iMe - that fulfil all OMME's criteria.

Chapter 7 RGeme

7.1 Overview

RGeme (Rhythmic Meme Generator) is a multi-agent interactive musical system developed in C++ and was designed as an initial test bed for the OMME. Having in mind that this system could be expanded in subsequent implementations, some of its components (e.g., musical material, segmentation algorithm, and generative model) were intentionally kept simple.

The concepts of the OMME were all implemented, however. Interactivity is achieved via (one or more) agents, who execute human-inspired activities and communicate with the outside world and/or between themselves via MIDI files. RGeme logs all the agents' activities and memories and this material is used to evaluate their respective musical evolution.

In the following Section, RGeme's main components are described.

7.2 The System

7.2.1 Music Representation

RGeme only deals with monophonic rhythms; therefore, in this Chapter, the word music can be used to refer to a piece of music in its entirety or only to its rhythmic component. A minimal unit of rhythmic duration is chosen (e.g., 8th note, 16th note) and rhythms are encoded as vectors of 0s and 1s (Figure 8), where 1 means the triggering of sounds and 0s are used to represent rests and as time placeholders. This scheme allows the representation of only note onsets and, although simple, can be efficiently employed to any music.

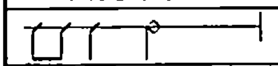
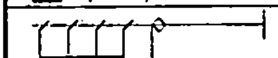
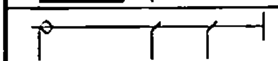
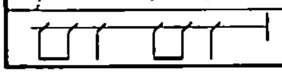
Music staff	Representation
	11101000
	11111000
	10001010
	11101110

Figure 8: Musical examples and corresponding representation.

All musical pieces are kept in the same space in the system called "musical store", where the agents can look for music for interaction and deliver their own compositions.

7.2.2 Interactivity

All agents have the same internal structure and the ability to perform three types of musical activities (tasks): listen to, practice and compose music. A number of tasks (Goal Matrix) must be specified for each agent, prior to the start of a simulation. The sequence in which these tasks are executed directly affects the development of the agent's memories (Style Matrices).

7.2.2.1 Musical Tasks

Agents execute musical tasks according to the stage they are in within their "musical career", which is progressive in terms of complexity, starting from "listeners" and moving towards "students" and then "composers". The existence of different stages of development does not change the fact that all agents have the same internal structure but only serves to structure their career. Table 3 shows which tasks (listen, practice, and compose) agents are allowed to perform in each stage (listener, student, and composer) of their career:

	Listen	Practice	Compose
Listener	Y	N	N
Student	Y	Y	N
Composer	Y	Y	Y

Table 3: Agents' musical career.
(Y = yes and N = no)

Listen: the listening task was designed to simulate the general human ability of listening to music. Agents actively seek music to listen to at any time in their musical career. Once a piece of music is chosen, agents will listen to it and progressively transform their memories according to a number of perceptive and cognitive algorithms. Agents can listen to and practice music composed by other agents as well as music composed by the outside world.

Practice: practice tasks are executed by student and composer agents and are similar to listening tasks except for the fact that once a piece of music is chosen, agents apply the transformation to their memories several times. This design attempts to replicate the fact that, because of a higher level of attentiveness and exercising, music students have their memories transformed more permanently when compared to normal listeners.

Compose: only composers can compose music. This task entails the possibility of interaction between agents as well as between agents and the external world.

7.2.2.2 Goal and Evaluation Matrices

Goal Matrices define the number of tasks the agents will execute during their lifetime and can be individually set for each agent prior to the start of a simulation. Therefore, depending on the simulation design, agents can have identical or different goal matrices. Table 4 contains an example of a Goal Matrix and shows the tasks a hypothetical agent is programmed to perform during a specific simulation.

	Listen	Practice	Compose
Listener	33	n/a	n/a
Student	33	34	n/a
Composer	0	0	0

Table 4: A Goal Matrix.

An Evaluation Matrix is also set for each individual agent; depending on the simulation design, agents can have identical or different evaluation matrices. It consists of a set of rules (composer's name and/or year of composition) that agents use to choose from the music available in the system at any given time. Evaluation matrices can determine the same rules for an agent's entire lifetime or different rules according to the stage in which agents are in their career. For instance, Table 5 shows that a given agent will choose from the pieces composed by Bach during the listener phase, by Bach or Mozart during the student phase and by the other agents, during the composer phase.

	Composer	Year begin	Year end
Listener	Bach	-	-
Student	Bach or Mozart	-	-
Composer	Agents	n/a	n/a

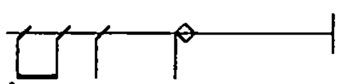
Table 5: Evaluation Matrix.

Goal and evaluation matrices forecast the series of activities that agents will perform during their lifetime, something that is analogous to the idea of a musical ontogeny. In the real world, one could not predict how many times or what type of music someone would listen to in the future. However, these are precisely some of the major elements that contribute to human's artistic values and to how music evolves.

In artificially synthesised worlds it is often convenient to restrain the agents' freedom to decide about certain aspects of their interactivity and have a controlled environment through which some objective analyses can be done. Different scenarios that combine goal and evaluation matrices can be drawn in order to allow these analyses, some of which will be described below.

7.2.3 Perception and Memory

Once a piece of music is chosen from the available pieces in the system, agents perceive the rhythmic stream and then segment it at every music bar line. Each segment corresponds to one rhythmic meme (Figure 9).



11101000

Figure 9 A rhythmic meme.

Memes are stored in Style Matrices (memory), along with the following information:

a) The dates (cycle) in which they were listened for the first and the last time,

b) The number of times they were listened to,

c) Their weight (importance), and

d) The music they were listened from.

RGeme sequentially generates integers that are sent to the agents and correspond to an internal "date", i.e., an abstraction of a time step, also called a cycle. When an agent receives the integer representing the new cycle, it executes all the tasks assigned to that cycle. The system will only generate a new integer once all the agents execute all the tasks assigned to the previous cycle. In summary, a cycle is a time slot that is allocated for the execution of a certain number of tasks.

In addition to controlling the sequence of execution of the tasks, the date (or cycle) information is stored in the style matrix. For each individual rhythmic meme, the date of the first (dFL - date of first listening) and the last (dLL - date of last listening) listening is kept.

Table 6 shows an extract of a style matrix.

#	Meme	dFL	dLL	nL	W
1	01010110	1	1	2	1.026
2	01011000	1	1	2	1.017
3	11010000	1	1	2	1.021
4	00100010	1	1	2	1.013
5	01110111	1	1	4	1.025
6	11011101	1	1	6	1.022
7	10010111	1	1	6	1.023
8	10010101	1	1	4	1.019
9	11110111	1	1	15	1.014
10	10001000	1	1	1	1.000

Table 6: Extract of a Style Matrix.
(Meme = rhythmic representation, dFL = date of first listening, dLL = date of last listening, nL = number of listening, W = weight)

The style matrix is in fact an abstract representation of the agent's memory, and ultimately corresponds to the agent's musical knowledge. In addition to this, the weight assigned to the memes must be understood in relation to each other, i.e., as the interactions occur, some of them will be strengthened while others will be weakened. Roughly speaking, making a parallel with the human memory, we could say that the "neuronal configurations" associated with each meme in the agent's memory is constantly being adapted to the environment.

Learning

Learning in RGeme is achieved by the application of a "transformation algorithm" that consists in reinforcing the weights of each individual meme in the memory according to the measure of distance between these memes and the meme that is being heard or generated at a given moment. The same rules apply whether the agent is listening to, practicing or composing a new piece. The transformation algorithm follows these steps:

In the beginning of a simulation, the style matrix is empty and receives the first meme (Table 7). Its weight is set to 1 and the dates of first and last listening are set according to the general time (cycle) controlled by the system.

#	Meme	dFL	dLL	nL	W
1	01011101	1	1	1	1.000

Table 7: Style matrix after 1st meme.

The second incoming meme (Table 8) is then compared with the meme in the style matrix. If they are different the new meme is copied and its weight is set to 1. The weight of the previous meme is updated according to a measure of distance between the new and the existing meme.

#	Meme	dFL	dLL	nL	W
1	01011101	1	1	1	1.125
2	11011101	1	1	1	1.000

Table 8: Style matrix after 2nd meme.

The same process is repeated with all the next incoming memes until the end of the musical piece. If the incoming meme is different, it is copied to the style matrix, its weight is set to 1 and the all the previous memes in the style matrix have their weights updated. If the incoming meme is not different, it is not copied but the weights of all the previous memes in the style matrix have their weights updated.

Supposing that during the same simulation the agents interact with more than one piece of music, from the second piece onwards, at the end of the interaction with each new piece, a "forgetting effect" is applied to the memes of the style matrix that are not present in the new music. In this case, the weight of the memes is reduced by a constant factor (e.g., f = -0.001).

Measure of distance

The distance between any two given memes a = [a1, a2, ... an] and b = [b1, b2, ... bn], is defined as:

$$d(a,b) = \frac{1}{n}\sum_{i=1}^{n} | a_i - b_i |$$

Equation 1: Measure of distance between two memes.

For example, the distance between the memes a = 01011101 and b = 11011101 is d(a, b) = 0.125 while the distance between the memes a and c = 11111111 is d(a, c) = 0.375. This type of measurement is often referred to as city block distance (Verth, 2008).

7.2.4 Creativity

The generative model adopted by RGeme is very straightforward: compositions are simply a sequence of 'n' memes ('n' defined a priori by the user). An agent generates each one of these memes sequentially and by randomly choosing from the existing memes in its style matrix according to the distribution of probability based on their individual weights. Therefore, in order to execute a composition task, an agent must already possess some rhythmic memes in its memory learned from the execution of previous tasks. Every time a new meme is generated, the transformation algorithm is also applied to the style matrix according to the algorithm described in the previous Section. Figure 10 shows one of RGeme's rhythmic compositions.

Figure 10: An example of a rhythmic composition generated by RGeme.

As mentioned above, RGeme was meant to be a test bed for the OMME and this generative algorithm, although unsophisticated, achieved the purpose of contextually transmitting the memes to the other agents. "Contextually" here means that the transmission of memes is representative of the importance they have in the agent's memory.

7.3 Case Study

A series of experiments were carried out with RGeme, some of which have already been published in Conference papers (Gimenes *et al.*, 2005a; Gimenes *et al.*, 2005c; Gimenes *et al.*, 2005b). In this and in the next sections some of these experiments are presented in order to demonstrate the potentialities that have been accomplished with RGeme in a number of typical cases and scenarios.

In the case study described in this Section[3], the aim was to experiment with compositions belonging to the same genre and composers from the same period as to draw connections within a specific musical school. A corresponding scenario in the real world would be to observe how the musical worldview of a single individual would evolve if this individual interacted with a controlled set of musical pieces from a single historical period and geographical location. Therefore, in this case, a simulation will be designed with only one agent and a number of controlled compositions and musical tasks.

7.3.1 The Musical Data Set

A musical data set of 29 pieces by three Brazilian composers was chosen:

> Chiquinha Gonzaga: *Annita, Atraente, Gaucho, Lua Branca, Nao Insistas Rapariga*
>
> Ernesto Nazareth: *A Fonte do Lambari, Amèno Reseda, Apanhei-te Cavaquinho, Arrojado, Bambino, Bombom, Brejeiro, Catrapuz, Comigo na Madeira, Crises em Penca, Fado Brasileiro, Matuto, Samba Carnavalesco, Tenebroso* and
>
> Jacob do Bandolim: *A Ginga do Mane, Ao Som dos Violoes, Caricia, Diabinho Maluco, Feitico, Magoas, Noites Cariocas, Orgulhoso, Quebrando o Galho, Receita de Samba*

These pieces belong to a genre called "chorinho" which was very popular during the end of the 19th century and beginning of the 20th century in Brazil. Chorinho originates from the minstrel tradition and could be described as very lyrical and predominantly instrumental. Played by small ensembles, the pieces are generally consistent in terms of structure (symmetry) and rhythm.

---

[3] The MIDI files of the musical pieces as well as the data generated by this experiment and analysed in this Section (file "RGeme Case Study 1.xls") can be found in the CD that accompanies this Thesis.

All the 29 pieces were transcribed into the software editor Finale from which MIDI files were generated and put into RGeme's musical store.

7.3.2 Goal and Evaluation Matrices

RGeme was configured to create only one agent to which a series of 100 listening tasks was given (Table 9). No composition task was assigned, as the aim of this simulation was to observe the development of a single agents' musical worldview and not the transmission of memes between agents.

	Listen	Practice	Compose
Listener	33	n/a	n/a
Student	33	0	n/a
Composer	34	0	0

Table 9: Agent's Goal Matrix.

The evaluation matrix divided the simulation into three specific phases. During the first 33 tasks the agent should choose only from Gonzaga's pool of compositions, in the next 33 tasks only Nazareth's compositions and during the remaining 34 tasks, only Bandolim's compositions (Table 10). As no time constraint (year begin, year end) was specified, the agent was allowed to choose from the entire set of compositions by each composer.

	Composer	Year begin	Year end
Listener	CGonzaga	-	-
Student	ENazareth	-	-
Composer	JBandolim	n/a	n/a

Table 10: Agent's Evaluation Matrix.

7.3.3 The Simulation

In each time cycle, the agent performed a listening task for which a piece of music was chosen according to the rules of the evaluation matrix. After that, the agent retrieved the rhythmic information from the chosen music and then parsed this information into rhythmic memes that were sequentially sent to the style matrix (memory) for transformation. The segmentation was done at every bar line, as explained in Section 7.2.3 above.

RGeme took a "snapshot" of the agent's style matrix at the end of each task in order to register the transformations and observe the behaviour of each individual rhythmic meme during the simulation. In the following paragraphs two extracts from the style matrix are shown.

Cycle 1

The first composition chosen by the agent in this simulation was *Lua Branca*, by Chiquinha Gonzaga. Table 11 shows the state of the style matrix for the first seven memes after the execution of the listening task:

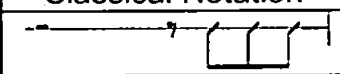
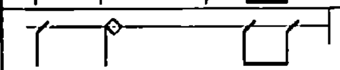
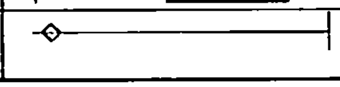
#	Meme	Classical Notation	dFL	dLL	nL	W
1	00000111		1	1	1	1.026
2	11111111		1	1	18	1.036
3	10100111		1	1	15	1.035
4	10100011		1	1	3	1.030
5	11111010		1	1	6	1.025
6	10000111		1	1	6	1.023
7	10000000		1	1	1	1.000

Table 11: Extract from Style Matrix after 1st music. (Meme = rhythmic representation, dFL = date of first listening, dLL = date of last listening, nL = number of listening, W = weight)

Cycle 2

In cycle 2, the agent chose the composition *Gaucho*, by Chiquinha Gonzaga. Table 12 shows an extract of the agent's style matrix after the execution of this task:

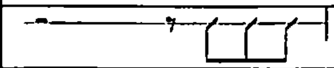
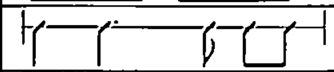
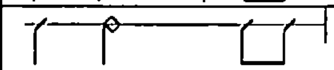
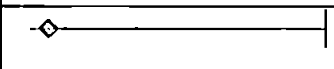
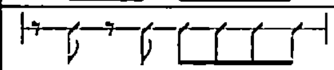
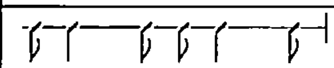
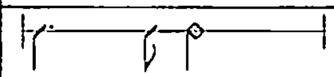
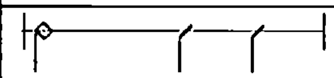
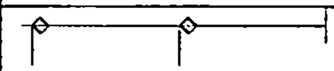
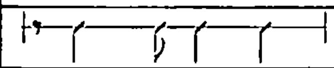
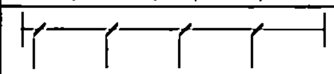
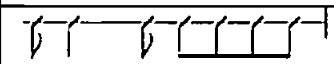
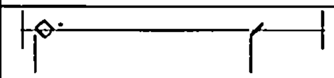
#	Meme	Classical Notation	dFL	dLL	nL	W
1	00000111		1	1	1	1.024
2	11111111		1	2	28	1.062
3	10100111		1	1	15	1.034
4	10100011		1	1	3	1.030
5	11111010		1	2	10	1.053
6	10000111		1	1	6	1.024
7	10000000		1	2	4	1.020
8	01111111		2	2	2	1.023
9	01011111		2	2	5	1.021
10	11011101		2	2	1	1.021
11	10011000		2	2	1	1.022
12	10001010		2	2	1	1.016
13	10001000		2	2	9	1.015
14	01011010		2	2	5	1.011
15	10101010		2	2	1	1.008
16	11011111		2	2	1	1.005
17	10000010		2	2	1	1.004

Table 12: Extract from Style Matrix after 2nd music.
(Meme = rhythmic representation, dFL = date of first listening, LL = date of last listening, nL = number of listening, W = weight)

It is important to know how to read these memory snapshots in order to track the differences between them. For instance, after the first cycle of interaction (Table 11), meme 11111111 (second in the list) had been listened (nL) 18 times and its weight (W) was 1.036. After the second cycle of interactions (Table 12) the number of listening for the same meme was 28 and the weight had been raised to 1.062.

In the first cycle, meme 10100111 (3rd in the list) was listened 15 times and its weight had been set to 1.035. In the second cycle, the number of listening had been kept the same (it was not present in the second composition) and the weight dropped to 1.034 meaning that a "forgetting" effect was applied because it was not listened in two consecutive interaction cycles.

Meme 10000010 (last in Table 12) only appeared in the second cycle and its weight was set to 1.004. This means that, after its first appearance (in which the weight had been set to 1), the weight raised due to the comparisons made with the other memes that were listened to afterwards.

7.3.4 Results

These readings allow us to address a series of interesting questions and observe the development of the agent's musical knowledge in comparison with the compositions it interacted with.

7.3.4.1 Total of Learned Memes

Starting with basic questions, it is possible to track the number of (and which) memes were learned during each phase of the simulation. For instance, during the first 33 tasks, while the agent was listening to the music by Gonzaga, 31 memes were learned. In the second part of the simulation, 32 new memes were learned from the music by Nazareth. Finally, in the last part of the simulation, the agent learned 24 new memes from Bandolim's music. This information is represented in Figure 11.

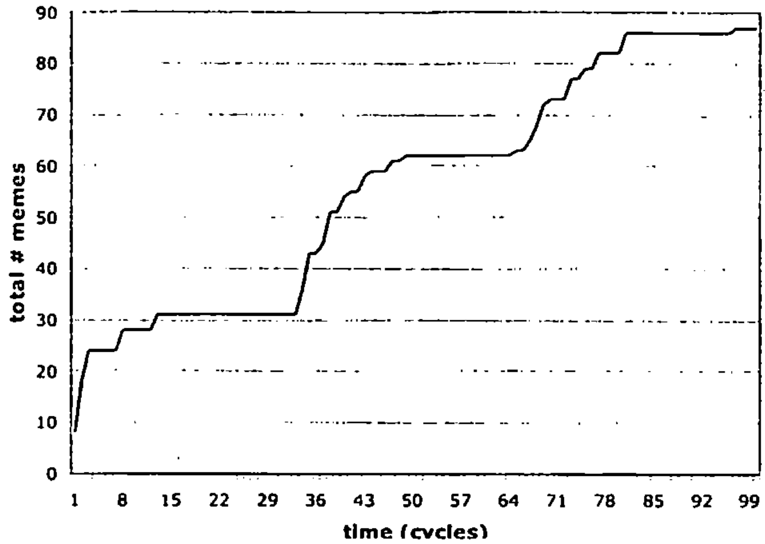


Figure 11: Number of memes learned in time.

We could hardly imagine that, in the real world, someone would listen sequentially to a series of composers the way the agent did in this simulation. In an artificially controlled environment, however, the interest of the graph shown in Figure 11 would be to demonstrate the "degree of novelty" that each composer contributes to the agent's memory in comparison to the previously heard composer.

7.3.4.2 Number of Times Each Meme Is Heard

Another measurement obtained from this simulation was the number of times each one of the memes was heard by the agent. As it would be difficult to view all of them in the same graph, Figure 12 shows this information for only the first learned 20 memes.

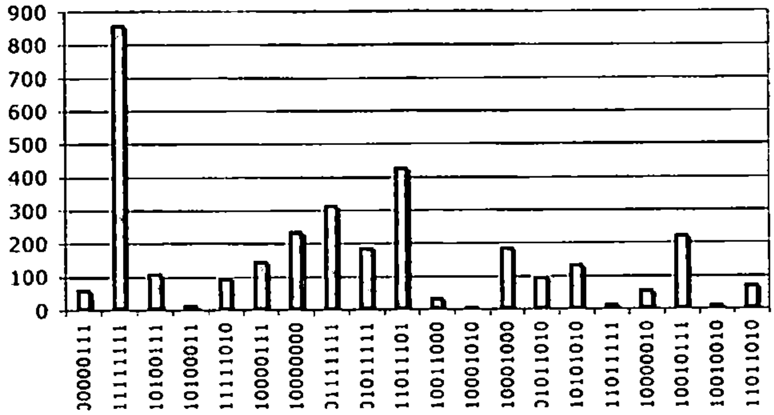


Figure 12: First 20 learned memes: number of listening.
(Axis x = meme representation, axis y = number of times memes were listened)

By itself, this type of information would only allow a rough indication of the importance of each meme. In fact, we could get much better results if, instead of looking at the number of times each meme was heard, we followed their relative importance (weight) at the end of each cycle. This measurement is demonstrated in the following paragraphs.

7.3.4.3 Relevance of Memes In Time

As mentioned above, the increase or decrease of the weight of the memes is the direct result of the number of times and the order (date) they were received by the memory. Figure 13 shows precisely this behaviour for all the memes during the whole simulation. Each line corresponds to an individual meme. Every time a new meme was learned a new line appears. If a meme is not heard, its curve starts to fall, meaning that the agent starts to forget it.

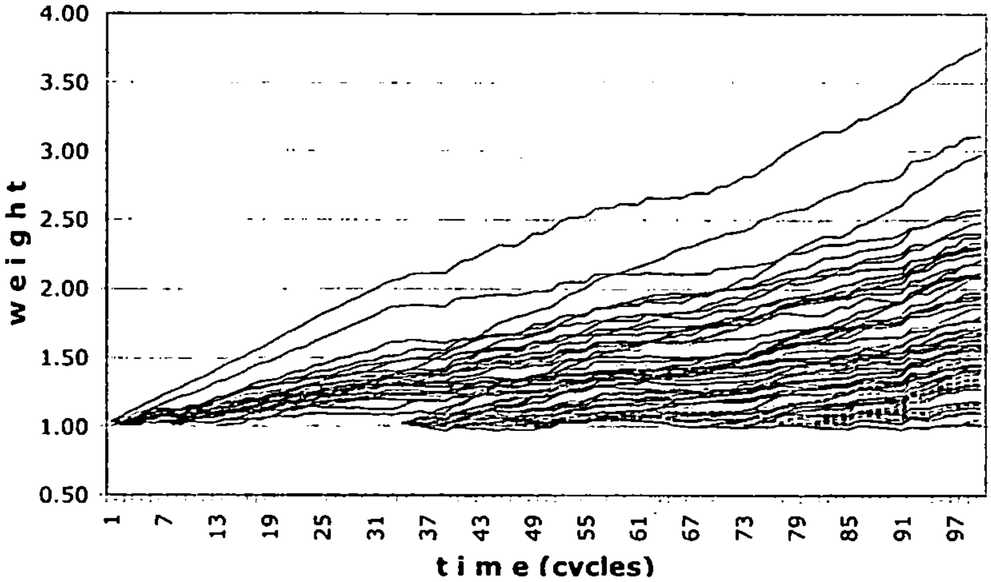


Figure 13: Memes curve of importance in time.

Of course, it is very difficult to view the evolution of all the 87 memes at the same time in the same graph. Figure 14 shows a few of them, which represent some of the typical behaviours that emerged from the interactions. Table 13 shows which memes correspond to the curves in Figure 14.

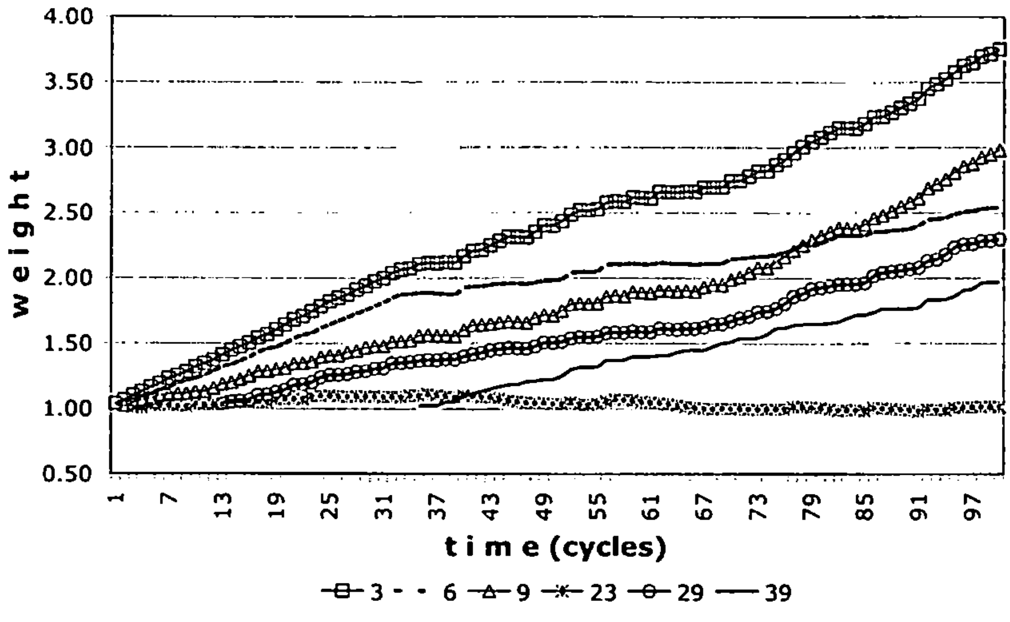


Figure 14: Memes curve of importance in time (selection).

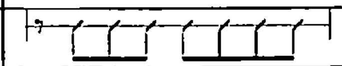
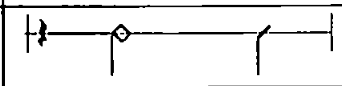
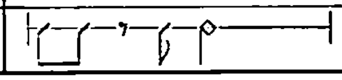
#	Meme	Classical Notation	dFL	dLL	nL	W
3	11111111		1	100	862	3.753
6	11111010		1	91	100	2.543
9	01111111		2	100	318	2.982
23	00100010		3	57	14	1.013
29	10111111		13	95	51	2.297
39	11011000		35	98	69	1.970

Table 13: Description of memes (Figure 14).
(Meme = rhythmic representation, dFL = date of first listening, dLL = date of last listening, nL = number of listening, W = weight)

In order to demonstrate the relevance of this analysis, it is important to read both Table 13 and Figure 14 together. For instance, the agent listened to memes 3 and 6 in the first interaction (dFL = 1) from music *Lua Branca*, by Gonzaga. Meme 9 appeared in the second interaction (music *Gaucho*, same composer), meme 23 in the third (music *Annita*, same composer), meme 29 in the 13th (music *Atraente*, same composer) and meme 39 in the 35th interaction (music *Tenebroso*, by Nazareth).

Although meme 23 was heard the first time in cycle 3, its relative importance comparing to the other memes was never very high. On the other hand, the agent only listened to meme 39 in cycle 35 and, at the end of simulation was more important than meme 23.

Meme 6 was relatively important in the music by Gonzaga but its importance was less significant after the agent began to listen to the music by Nazareth. For this reason, at the end of the simulation, meme 9 was more important than meme 6. Memes 3 and 9 had a steady and comparable performance during the whole simulation. At the end, meme 3 was the "winner" over all the other memes. Table 14 shows the 10 most relevant memes at the end of the simulation.

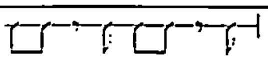
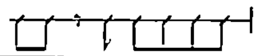
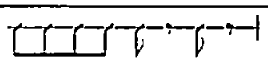
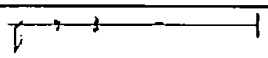
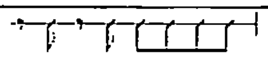
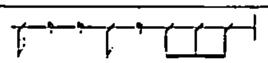
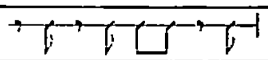
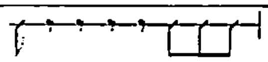
#	Meme	dFL	dLL	nL	W
3		1	100	862	3.753
11		2	98	431	3.111
9		2	100	318	2.982
17		2	92	18	2.577
6		1	91	100	2.543
8		1	100	240	2.482
10		2	98	191	2.422
19		3	94	225	2.402
38		35	98	161	2.381
7		1	99	149	2.329

Table 14: Winning memes.
(Meme = rhythmic representation, dFL = date of first listening, dLL = date of last listening, nL = number of listening, W = weight)

7.4 Additional Case Studies

Another series of four experiments (simulations A to D) was prepared with a different musical data set. In this case, 12 compositions were chosen, the 6 Bach's Inventions for Two Voices (only the right hand melodic line) and 6 Tom Jobim's tunes shown in Table 15 below. The Appendix CD that accompanies this Thesis includes all the data related to these simulations.

#	Composer	Music
B1	Bach	Invention 1
B2	Bach	Invention 2
B3	Bach	Invention 3
B4	Bach	Invention 4
B5	Bach	Invention 9
B6	Bach	Invention 10
J1	Jobim	Corcovado
J2	Jobim	Eu sei que vou te amar
J3	Jobim	Felicidade

#	Composer	Music
J4	Jobim	Garota de Ipanema
J5	Jobim	Samba de uma nota so
J6	Jobim	Wave

Table 15: Musical Data Set

7.4.1 Simulations A and B

In the first two simulations, the objective was to observe how the curve of the learned memes would be affected using the same set of pieces but in a different order. Therefore, simulations A and B used the same agent, musical data set, and number of tasks. Only the order of the pieces is different: in the first case, a series of compositions by Bach and Jobim (B1 - B2 - B3 - B4 - B5 - B6 - J1 - J2 - J3 - J4 - J5 - J6[4], in this order) and, in the second case, Jobim and then Bach (J1 - J2 - J3 - J4 - J5 - J6 - B1 - B2 - B3 - B4 - B5 - B6 [5], in this order). Figure 15 and Figure 16 show the evolution of the number of learned memes.

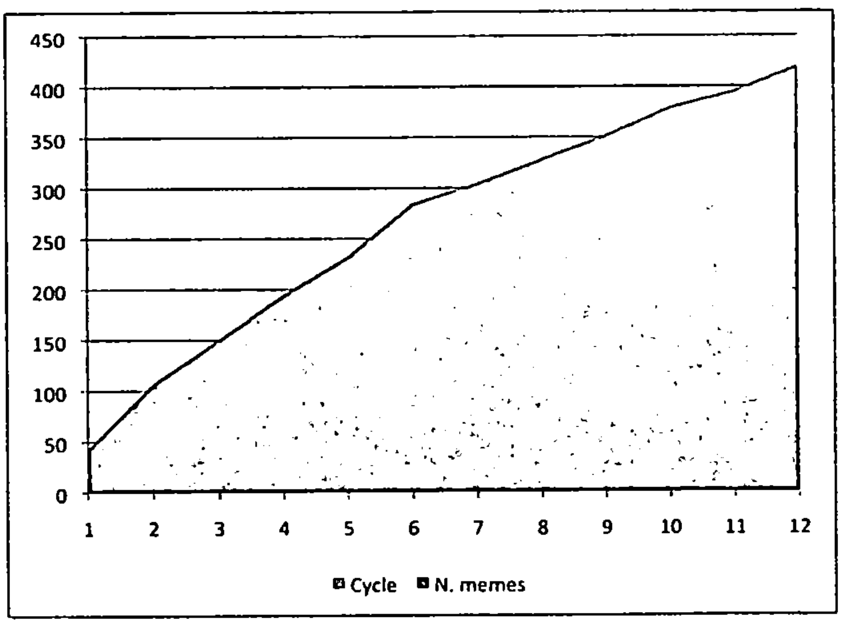


Figure 15: Bach then Jobim (simulation A)
(Axis x = time cycles, axis y = number of memes)

[4] Codes refer to left hand column of Table 15.

[5] idem.

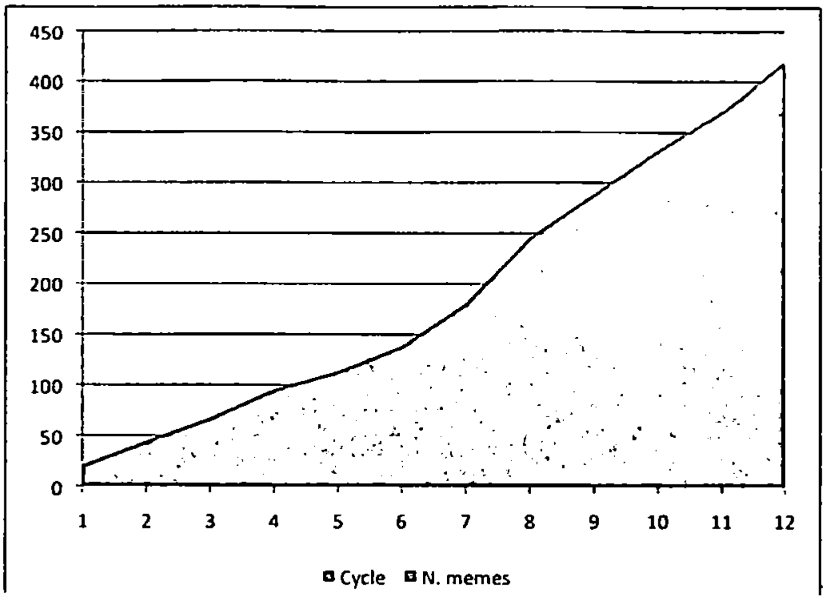


Figure 16: Jobim then Bach (Simulation B)

(Axis x = time cycles, axis y = number of memes)

It is easy to observe in these cases that, even though the number of memes (421) is the same at the end of the two simulations, the knowledge at corresponding moments in the two graphs is different. If the agents had been given composition tasks at different moments in their career, the new pieces would have reflected this fact.

7.4.2 Simulation C

Simulation C was conceived with a different objective and strategy in mind. Agent X should listen to a number of works by the same composer (B6 - B6 - B3 - B5 - B3 - B5 - B2 - B6 - B2 - B3) and then to a different composer (J1). At the end of the simulation Agent X should create a new composition. The idea was to follow the transmission of these different musical sources into Agent X's music.

After the simulation was run, the log file[6] showed that the memes Agent X used in his composition came from a number of different sources: B5 - B2 - J1 - B2 - B2 - B5 - B2 - B6 - B2 - J1 - B2 - B2 - B6 - B3 - J1 - J1 - B2 - B6 - B6 - B6 (in this order). Figure 17 shows this piece:

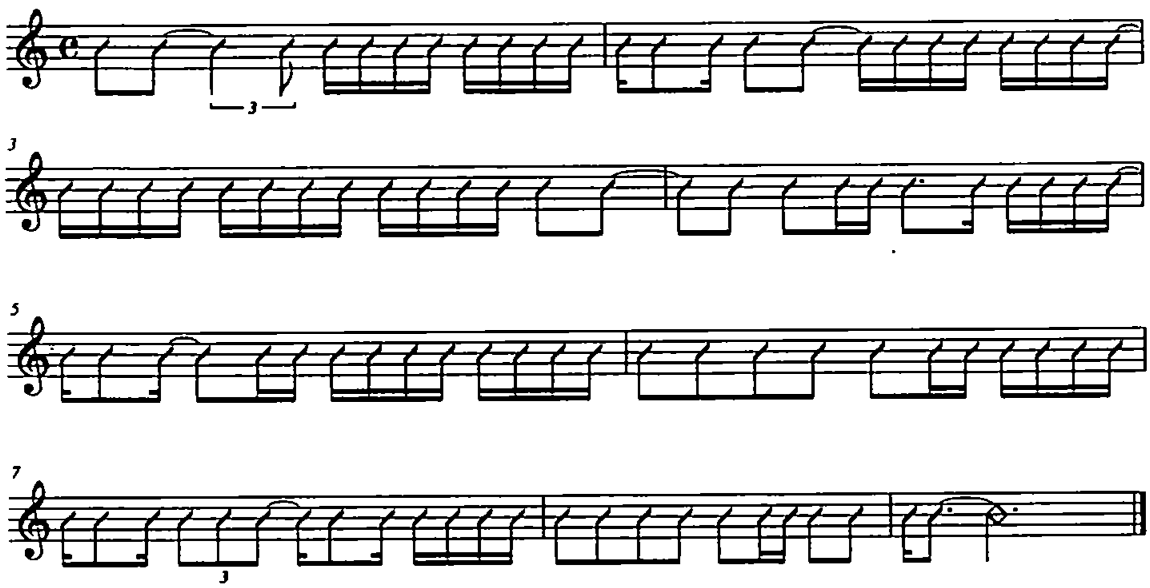


Figure 17: Agent rhythmic composition

The same piece is shown in Figure 18 but with added pitch information. RGeme was at this moment at the end of its cycle of development and already possessed the ability to deal with pitch information. As the previous version only dealt with rhythmic information, it could be fair to say that the compositions produced so far were "acceptable" at times. The added pitch, however, helped to show the weakness of the creative model behind RGeme. The melodies often seem to wander without any special sense of direction or of structure. This was due, in the first place, to the fact that RGeme was not keeping track of how one meme was enchained to the following one. In addition, the agents' compositions were simply a sequence of randomly generated memes taken

[6] Log file "runlog.txt" can be found in the Appendix CD

from their Stile Matrices. These facts can be easily seen in the melody of Figure 18:

Figure 18: Agent composition

7.4.3 Simulation D

Finally, simulation D was prepared with two agents. Agent X should initially listen to 20 pieces by Bach, listen to one piece by Jobim and then compose one new piece. After Agent X accomplished its tasks, Agent Y should listen to one piece by Jobim, one piece by Agent X and compose its own piece. This simple simulation demonstrates how, in a multi-agent environment, memes propagate from one "brain" to another.

The pieces Agent X chose to hear were: B4 - B2 - B2 - B4 - B3 - B4 - B6 - B2 - B4 - B1 - B2 - B1 - B3 - B4 - B6 - B3 - B2 - B6 - B3 - B1 - J3. After that, Agent X created a new composition (X_Music) by generating 20 memes from its Style Matrix. The sequence of memes Agent X used in its composition originated from these pieces: B3 - B1 - B1 - B1 - J3 - B3 - B4 - B1 - B2 - B1 - B6 - B6 - B3 - B6 - B4 - B2 - B2 - B1 - B4 - B6. Figure 19 shows the rhythmic composition and Figure 20 shows the same compositions with the added melodic information.

Figure 19: Agent X rhythmic composition

Figure 20: Agent X composition with melody

After Agent X finished its tasks, Agent Y listened to two pieces, J1 and X_Music. Finally, Agent Y created a new composition (Y_Music). The 20 memes used in Agent Y's composition originated from these pieces: J1 - J1 - J1 - X_Music - X_Music - X_Music - J1 - X_Music - J1 - X_Music - J1 - J1 - J1 - J1 -. X_Music - J1 - J1 - X_Music - J1 - X_Music. Figure 21 shows Agent Y's rhythmic composition and Figure 22 shows the same compositions with the added pitch information.

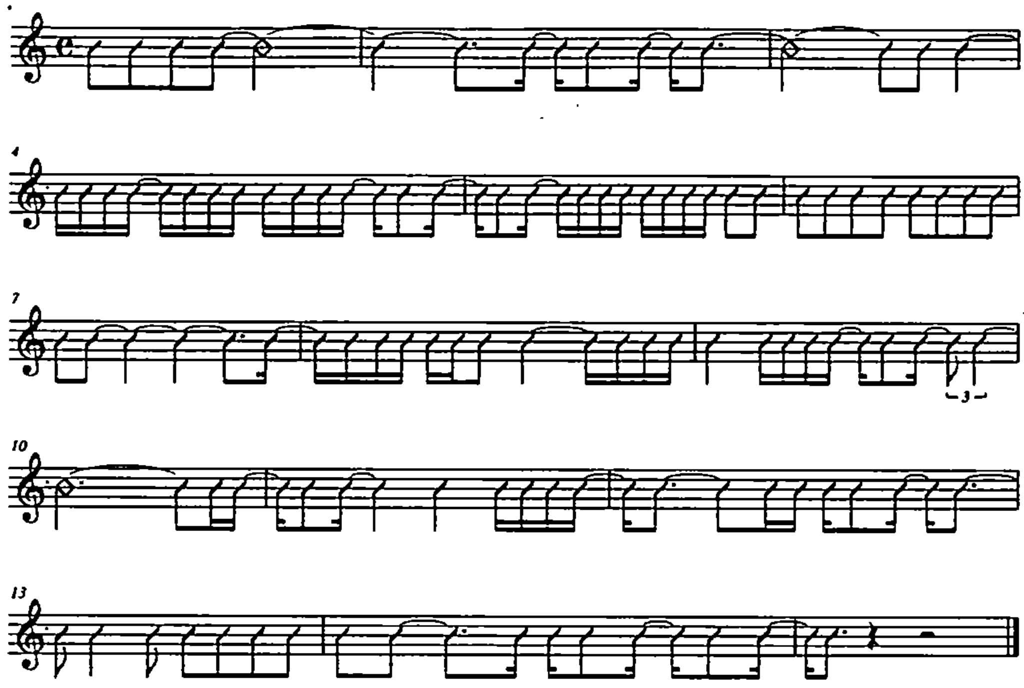


Figure 21: Agent Y rhythmic composition

Figure 22: Agent Y composition with melody

7.5 Conclusion

This Chapter introduced the multi-agent musical interactive system RGeme as a test bed for the OMME. In fact, RGeme complies with all the three conditions that were defined for this model. In the first place, RGeme is a fully interactive system in the sense proposed in this Thesis (p. 93). Interactivity is based on the intelligent agent's paradigm: agents exchange musical information with the environment, perceiving, analysing and taking actions. They have their own representation of the world (memory), which is transformed via human-like interactive musical activities that convey the transmission of musical units of information (rhythmic memes). Finally, RGeme provides tools to evaluate the development of the agent's musical knowledge.

A case study has also been presented in which it is demonstrated how RGeme could help to achieve the OMME aims (p. 133), contributing to the understanding of the human cognitive system via computer modelling. In addition to this, the analyses described above demonstrated the contributions that an OMME-based system can give to musicology, particularly in the field of music ontogenesis.

As explained above, given the fact that RGeme was a first attempt to meet the OMME's conditions, some of its features were intentionally kept simple. This is the case, for instance, of the musical material that is only monophonic rhythm and the segmentation of the rhythmic stream, based on a fixed length criterion, which could be considered unrealistic. Additionally, the implemented creative model in RGeme could not efficiently generate new compositions that would convey the sense of continuity from one rhythmic meme to another.

Considering the limitations discussed above and the lessons learned from RGeme, a new and expanded OMME-based system called Interactive Musical Environments (iMe) was implemented, which is the subject of next Chapter.

Chapter 8 iMe

8.1 Overview

The Interactive Musical Environments - iMe (Gimenes *et al.*, 2007a; Gimenes *et al.*, 2008) was initially intended to be an expansion to RGeme and provide solutions to some of the issues that had been postponed during the preliminary implementations of this system (e.g., musical representation, segmentation). iMe's large number of improvements justified, however, the new name.

The backbones of RGeme and iMe are very similar. They are both multi-agent interactive musical systems based on the OMME's requirements. Agents perform human-inspired activities and communicate between them and with the outside world. However, there are a considerable number of differences between these two systems, which will become clearer as iMe is introduced below:

iMe was especially designed to support interactivity from an improvisational point of view, a subject addressed in Section 2.2.1 (p. 42). Another important aspect is that interactivity is achieved through real-time as well as non real-time tasks. In addition, the choice for improvisation, and particularly for piano improvisation, among other features, directly affected the way music is represented in this system.

8.2 The System

Following the same structure adopted in the previous Chapter, this Section will address iMe's main elements.

8.2.1 Music Representation

Music representation in iMe is influenced by two major factors. Firstly, it is intimately associated with how sounds are captured by the ears, processed by and stored in the brain. Section 2.1 (p. 32) explained that the hearing system features a number of mechanisms for analysing different sound characteristics, a process called feature extraction. Following this model, iMe implements a series of filters (feature filters) that would correspond to different ear sensors and that extract particular sound features such as the increase and/or decrease of sound frequency (pitch direction) or the musical density (number of simultaneous notes).

iMe uses the MIDI protocol for communication among the agents and between the agents and the external world, from which the agents extract symbolic representation. It would appear that the process of feature extraction would not make much sense in this case, because the agents directly manipulate musical symbolic representation. In fact, it does; initially, because agents do not have a particular knowledge about musical note events themselves. They perceive changes in the "sound flow" between two consecutive moments in time and extract "sensory information" from them. Secondly, because even though agents currently extract information from the symbolic representation of the MIDI protocol, they could as well capture information directly from sound. This ability is not currently implemented but future versions of the system in this regard would not significantly affect the agents' memory model (explained below).

The second factor that affects music representation in iMe is that, for the reasons explained in Section 2.2.1 (p. 42), I decided to focus on piano improvisation. In view of this, some of the filters mentioned in the previous paragraphs may be better suited for a genre of performance, or musical texture, in which pianists use the left hand to play a series of chords while the right hand plays the melodic line. In fact, any music that could be considered a subset of this type of performance would be also efficiently addressed (e.g., a series of chords, a single melody or any combination of the two).

Currently there are 5 filters that deal with the notes from the melody and another 5 filters that deal with non-melodic notes. These filters are shown in Table 16 below:

Filter	Type	Represents	Values
Melodic direction	Integer	Direction of the melody	-1 (downward), 0 (same pitch), 1 (upward), -2 (no data)
Melodic leap	Integer	Interval (measured in half steps) between two consecutive pitches in the melody	Any positive integer
Melodic inter-onset interval	Integer	Time interval between two consecutive pitches in the melody	Any positive integer
Melodic duration	Integer	Duration of the melody note	Any positive integer
Melodic intensity	Integer	Amplitude of the melody note	Integers between 1 and 127
Non-melodic number of notes	Integer	Number of notes that sound at the same time (vertical structures)	Any positive integer
Non-melodic note intervals from melody	Array of integers	Intervals (for non-melodic notes) from the melody note	Any positive integer, any number of elements in the array
Non-melodic inter-onset interval	Integer	Time interval between two consecutive vertical structures for non-melodic notes	Any positive integer

Filter	Type	Represents	Values
Non-melodic duration	Integer	Shortest duration of the vertical structures for non-melodic notes	Any positive integer
Non-melodic intensity	Integer	Strongest amplitude of non-melodic notes	Integers between 1 and 127

Table 16: Filters - range of values.

In order to avoid some idiosyncrasies of performed music, two reductionist functions were implemented. Notes that are attacked or released nearly at the same are vertically aligned, according to a threshold defined by the user. In addition to this, intensity, whose original range is defined by the MIDI protocol (0-127), is reduced to multiples of 10.

All musical pieces are kept in a memory space called the "Music Store" (Figure 48), where the agents can look for music for interaction and deliver their own compositions. Examples of iMe's music representation will be given and the above-mentioned concepts will be better understood in the Section "Perception and Memory" below (p. 175).

8.2.2 Interactivity

Interactivity in iMe has the same overall structure as in RGeme. Agents have the ability to perform musical tasks through which they communicate between each other and with the outside world. The outcome of this communication is that the agents' memory is constantly being changed and, as a result, their musical style constantly evolves.

In the case of iMe, however, the number and complexity of the musical tasks was increased. In addition to non real-time tasks (read and compose music), some real-time tasks (listen, perform, solo-improvise and collective-improvise) were also implemented.

8.2.2.1 Tasks

Agents are able to:

Read: Agents read music from MIDI files provided by the system's user and/or generated by the agents. As happens in the real world, the time spent to read a piece of music does not necessarily correspond to the duration of the music. In fact, agents can read and cognitively process music much faster than human beings.

Practice: Agents repeat the reading task a number of times (defined by the user of the system), attempting to replicate the activities of music students.

Listen: Agents listen in real-time to the music played by human performers on a MIDI instrument attached to the system. Because listening is a time-dependent task, during its execution (as well as during the execution of other real-time tasks) the rest of the simulation waits until this task is finished.

Perform: Agents play in real-time (send MIDI information to a MIDI port) music available in the Music Store.

Compose: Agents generate a new composition and deliver it to the Music Store

Solo-Improvise: Agents improvise in real-time. At the end of the task, the improvisation is delivered to the Music Store.

Collective-improvise: Agents improvise with a human performer in real-time. At the end of the task, both (agent's and human's) improvisations are delivered to the Music Store in separate files.

Similarly to what happens in RGeme, once a simulation is started, the system iMe sequentially generates integers that are sent to the agents and correspond to an internal "date", i.e., an abstraction of a time step, also called a cycle.

When an agent receives the integer representing the new cycle, it executes all the tasks assigned to that cycle. iMe will only generate a new integer once all the agents execute all the tasks assigned to the previous cycle.

8.2.2.2 Goals

In the literature (Maes, 1991; Russell *et al.*, 2002) agents are normally described as entities that act autonomously (p. 63) in order to achieve some goals. In the case of an agent-based musical interactive system this would mean that agents should decide, for instance, when they would interact or what type of task they would perform. However, in order to comply with the OMME's requirements and to use iMe as a platform for live music interaction as well a musicological tool, I decided to initially constrain the agent's autonomy: I have allowed them to choose which music they interact with (based on some parameters) but not how many times or when they interact.

Therefore, when designing a new simulation, the user of the system must assign a number of agents (at least one) and a number of tasks (at least one) for each one of these agents. If there are reading, practicing, and/or listening tasks, agents must also be instructed with the criteria (composer's name, genre, year of composition) they will use to choose from the pieces available in the Music Store. The more criteria given, the more constrained the choices will be, and vice-versa, the less criteria are given, the less constrained the agents' choices will be.

This design follows the experience obtained with RGeme's goal and evaluation matrices (p. 144) and facilitates the sketch of "developmental maps". Developmental maps are at the same time the agents' goals and the ontogenesis of their musical knowledge. As a hypothetical example, Figure 23 shows one of these maps in a simulation where two agents (agent "A" and agent "B") were assigned a number of listening, practicing, and composing tasks.

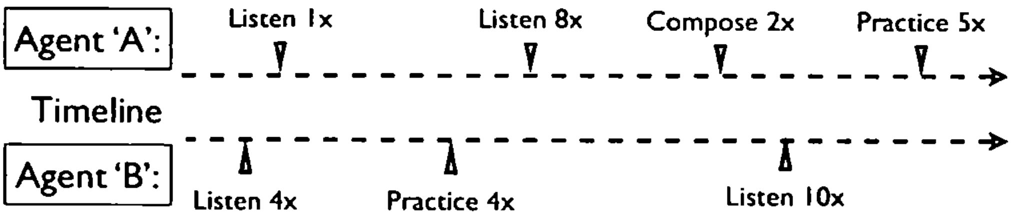


Figure 23: Agents' developmental paths.

I mentioned earlier that agents could interact with each other. This happens every time they are assigned listening, practicing or performing tasks in which the evaluation criterion is defined to choose from one of the other agents' compositions. In the developmental map shown in Figure 23 this would happen, for instance, after agent "A" delivers his first two compositions (third item in agent's "A" timeline) and agent "B" listens to a new series of compositions (third item in agent's "B" timeline).

8.2.3 Perception and Memory

iMe's agents initially perceive the music data flow and subsequently decompose it into a number of independent but interconnected streams that correspond to basic human sensory information (e.g., melodic direction or melodic inter-onset intervals). This mechanism results in a "feature stream" of data that is used for segmentation, storage (memory) and style definition purposes (Figure 24).

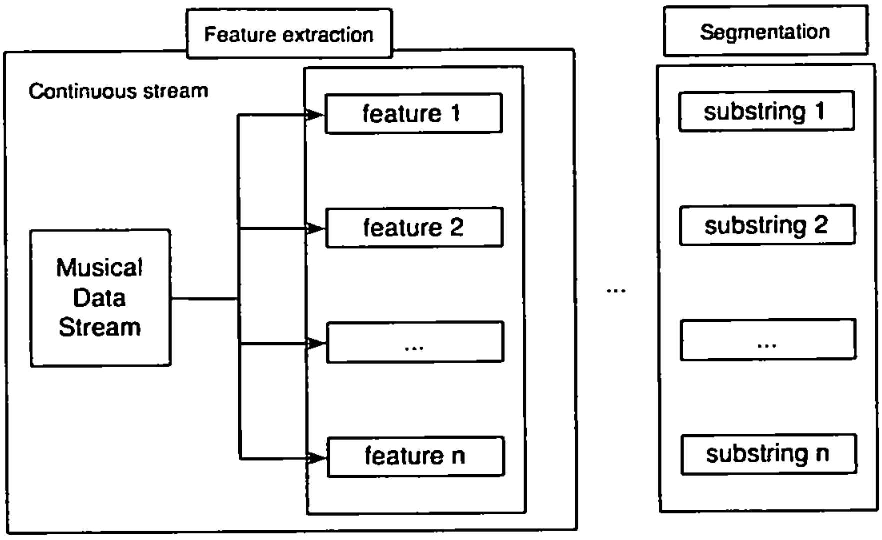
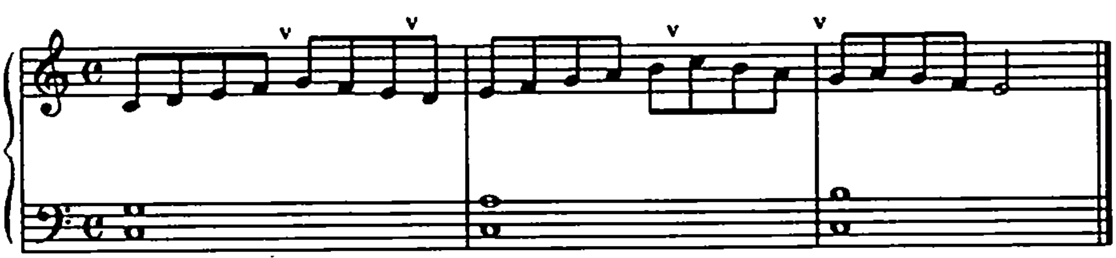


Figure 24: Feature extraction and segmentation.

As mentioned above, I have implemented 10 filters that focus on melodic (direction, leap, inter-onset interval, duration and intensity) and non-melodic notes (vertical number of notes, note intervals from the melody, inter-onset interval, duration and intensity). As might be expected, the more filters that are used, the more accurate is the representation of the musical flow. To help clarifying these concepts, Figure 25 shows a basic musical example in traditional musical notation and the corresponding feature streams as perceived by an agent:



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
a)	0	1	1	1	1	-1	-1	-1	1	1	1	1	1	1	-1	-1	-1	1	-1	-1	-1
b)	0	2	2	1	2	2	1	2	2	1	2	2	2	1	1	2	2	2	2	2	1
c)	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120
d)	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	480
e)	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6
f)	2	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	2	0	0	0	0
g)	5, 7	-2	-2	-2	-2	-2	-2	-2	7, 9	-2	-2	-2	-2	-2	-2	-2	8, 11	-2	-2	-2	-2

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
h)	120	-2	-2	-2	-2	-2	-2	-2	120	-2	-2	-2	-2	-2	-2	-2	120	-2	-2	-2	-2
i)	960	-2	-2	-2	-2	-2	-2	-2	960	-2	-2	-2	-2	-2	-2	-2	960	-2	-2	-2	-2
j)	6	-2	-2	-2	-2	-2	-2	-2	6	-2	-2	-2	-2	-2	-2	-2	6	-2	-2	-2	-2

Figure 25: Feature stream music representation.
(a = melodic direction, b = melodic leap, c = melodic inter-onset interval, d = melodic duration, e = melodic intensity, f = non-melodic number of notes, g = non-melodic note intervals from melody, h = non-melodic inter-onset interval, i = non-melodic duration, j = non-melodic intensity).

In the feature stream music representation of Figure 25, each row was generated by the agent's feature filter. The numbers in the first row of the table (from 1 to 21) correspond to each vertical structure in the traditional notation above. The number -2 represents the absence of data in a particular stream. Melodic direction can assume -1, 0 and 1 values, meaning descending, lack of and ascending movement. Melody leaps and intervals (non-melodic note intervals from melody) are shown in half steps. In streams that have time information (inter-onset intervals and duration) the value 240 (time resolution) is being assigned to quarter notes. Intensity is represented by the numbers in the MIDI range (0 to 127) divided by 10.

Segmentation

The next step in the agent's perceptive flow is the segmentation of the feature stream. As seen in Section 3.1 (p. 48), even though a fair amount of research has been done on segmentation, it remains an unsolved issue, due to its complexity. iMe adopted the principles of the Gestalt Psychology (p. 37) viewed under the perspective of habituation. Section 2.1 (p. 32) mentioned that the human attention is affected by this phenomenon, that results from "the fact that when many types of neurons are stimulated repeatedly with an identical stimulus, their output of impulses does not remain constant, but instead decreases over time" (Snyder, 2000, p. 24).

Therefore, in iMe while the agents perceive the sound flow, the repetition of the same signal in the feature stream would convey a lack of interest to the agents' cognitive system whereas a change of behaviour in the signal, after a number of repetitions, would draw their attention. Based on this notion, whenever there is a break in the continuity of the behaviour of each one of the feature streams, this would be an indication for a possible segmentation.

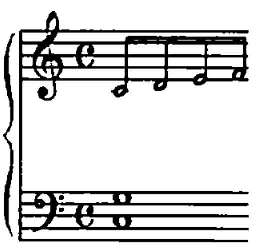
It is not obvious, however, which of the feature streams would be more appropriate to be used by the segmentation algorithm. Melodies, for instance, would be good candidates, as they appear to be more prominent in the musical flow; but which of its parameters: direction, leap, inter-onset intervals, duration or intensity? The texture of the whole music flow (in conjunction with intensity or another parameter) is also a good candidate as it conveys the sense of rupture and boundaries. Perhaps an optimal solution would be achieved by dynamically considering the behaviour (stability vs. rupture) of a combination of all the feature streams.

A further aspect of the feature streams is the fact that they provide an interesting approach to define larger structures in a musical piece. Phrase ends, for instance, usually convey a higher sense of closeness than local boundaries. Considering the behaviour of the feature streams, wherever the stability is broken at a higher number of streams, this would also be an indication of the existence of a higher sense of closeness and, therefore, of larger structures.

In iMe, I tried to segment the musical flow with separate feature streams and I found that a good compromise was to consider the behaviour of the melodic direction stream. This was the criterion I used to segment the musical example shown in Figure 25. Hence, every time the direction of the melody is about to change (row "a" of Figure 25), a new grouping starts. In the musical score of Figure 25 these boundaries are indicated with the symbol "v".

Each segment is called a "musical meme" and a meme is a group of "n" "feature substrings" (FS) originated from the feature streams. "n" is the number of musical features perceived by the agents; it can be variable as in a iMe a window (Figure 46) controls which feature filters will be active during a simulation.

Figure 26 shows an example of musical meme where the agent perceived only four feature substrings.



Feature substrings:
mdi: 0, 1, 1, 1
mle: 0, 2, 2, 1
mii: 120, 120, 120, 120
mdu: 120, 120, 120, 120

Figure 26: Example of musical meme.
(mdi = melodic direction, mle = melodic leap, mii melodic inter-onset interval, mdu = melodic duration)

8.2.3.1 Memory

The execution of any of the musical tasks requires the perception and segmentation of the musical flow and the adaptation of the memory. In general, musical tasks imply the production of sound, which is perceived by the ear and transmitted to the brain. A reading task does not produce sound, of course, but it can produce changes in the neuronal connections in the brain as if sounds had been produced (p. 32). Therefore, the next step in the flow of musical information consists of storing the musical memes in the agent's memory by comparing it with the elements that are previously stored.

In iMe, agents' memories have a short-term memory (STM) and a long-term memory (LTM). The STM is simply a queue of the newest memes sequentially sent to the memory for adaptation every time an agent executes any of the musical tasks. This queue has a limited number of elements that is defined *a priori* by the user of the system. Memes newly perceived occupy a new position in the STM queue; older ones "vanish", i.e., they disappear from the agent's "focus of conscious awareness".

A much more complex structure, the LTM is a group of "Feature Tables" (FTs). Each FT is responsible for storing a specific musical feature and each row in a FT, called a "Feature Line" (FL), keeps a record of:

FS: the feature substring,

Dates of interaction (date of first contact - DFC, date of last contact - DLC): time cycles in which the FS was sent to the memory,

Number of contacts (NOC): number of times the FS was sent to the memory,

Weight (W): relative weight of the FS

Connection pointers (CP): pointers that connect the FSs that were originally part of the same meme

Figure 27 shows an excerpt of a hypothetical FT (for melodic leaps) with 10 FLs; for the sake of simplicity, only the FSs and connection pointers are shown.

Feature n.2 (melodic leaps)		
FL #	FS	CP
FL 0	[0, 0]	0, 0, 0, 0, 0, 0, 0, 0
FL 1	[2, 2, 0, 1, 0, 1, 2]	5, 3
FL 2	[1, 0, 0, 3, 2, 2, 0]	1
FL 3	[1, 0, 0, 0, 1, 2, 2, 4]	2, 20, 10, 10
FL 4	[2, 0, 2, 0, 4, 1, 3, 0]	3
FL 5	[0, 3, 2, 7, 0, 2, 0, 4]	4
FL 6	[3, 0, 2, 0, 3, 2, 4]	5, 8, 10

Feature n.2 (melodic leaps)		
FL #	FS	CP
FL 7	[1, 0, 1, 2, 2, 0, 3]	6, 5, 3
FL 8	[2, 0, 2, 0, 2, 0, 0]	7, 3
FL 9	[2, 0]	8, 31, 8

Figure 27: A Feature Table excerpt.
(FL = feature line, FS = feature substring, CP = connection pointers)

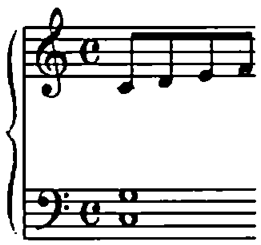
Notice that, when the musical memes are stored in feature tables, they are "dismantled" as the FSs that originally formed them are stored into separate FTs. CPs are the links between FSs from the original memes. However, it would not be possible to assure that, in generative mode, the same group of FSs are recombined and the original meme is replicated. In generative tasks (p. 187), the combination of FSs into new memes follow stochastic rules extracted from the material that is sent to the LTM.

These concepts will become clearer in the next Section through an example that will explain the adaptation algorithms step-by-step. To keep the example consistent with the rest of the Chapter, the same musical fragment of Figure 25 will be used.

8.2.3.2 LTM Transformation (Adaptation)

iMe specifies that every time agents interact with a piece of music their memories are changed according to the similarities and/or differences that exist between the current piece and their previous musical knowledge. The adaptation mechanism is fairly simple: the "strength" of an FS (weight and the number of CPs) is increased as much as an agent perceives it and the more agents perceive a FS, more CPs are registered and the FS's weight is increased. Conversely, if a FS is not perceived, its weight is decreased, which represents the fact that the agent begins to "forget" the FS. This mechanism is an innovation if compared to previous interactive musical systems and is responsible for much of the ever-changing dynamics of the FSs' weights.

Let us start with a hypothetical simulation in which the LTM of an agent is completely empty. As the agent starts perceiving the musical flow, the agent's "sensory organs" (feature filters) generate a parallel stream of musical features, which is segmented according to the mechanism described above (p. 175). The first meme (Figure 28) is then sent to the LTM, which, as a result, is adapted to it (Table 17).



Feature substrings:
mdi: 0, 1, 1, 1
mle: 0, 2, 2, 1
mii: 120, 120, 120, 120
mdu: 120, 120, 120, 120

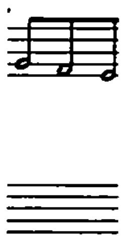
Figure 28: Meme 1.
(mdi = melodic direction, mle = melodic leap, mii melodic inter-onset interval, mdu = melodic duration)

In order to keep the example intelligible, I am only showing the representation of four feature tables: melodic direction (FT1), leap (FT2), inter-onset interval (FT3) and duration (FT4). Table 17 shows the four feature tables together. Notice that the connection pointers (CP) of FTs 2 to 4 point to the index "i" of FT1. This indicates that these FSs were originally part of the same meme. The CP information of FT1 is still empty because it stores a different type of information, as it will be explained in the following paragraphs. Initially the weight (W) was set to 1.0 for all of the FSs and the "date" information (DFC, DLC) is set to the cycle in which this task is performed during the simulation, in this case, the first task.

i	FS	DFC	DLC	NOC	W	CP
FT1: Melodic direction						
1	0, 1, 1, 1	1	1	1	1.0	
FT2: Melodic leap						
1	0, 2, 2, 1	1	1	1	1.0	1
FT3: Melodic inter-onset interval						
1	120, 120, 120, 120	1	1	1	1.0	1
FT4: Melodic duration						
1	120, 120, 120, 120	1	1	1	1.0	1

Table 17: LTM after adaptation to meme 1 (Figure 28).
(i = feature line index, FS = feature substring, DFC = date of first contact, DLC = date of last contact, NOC = number of contacts, W = weight, CP = connection pointers)

Then "comes" the next meme (Figure 29):



Feature substrings:
mdi: 1, -1, -1
mle: 2, 2, 1
mii: 120, 120, 120
mdu: 120, 120, 120

Figure 29: Meme 2.
(mdi = melodic direction, mle = melodic leap, mii melodic inter-onset interval, mdu = melodic duration)

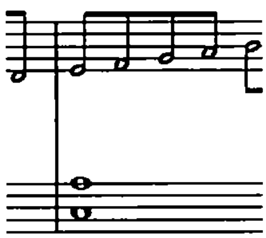
And the LTM is adapted accordingly (Table 18):

i	FS	DFC	DLC	NOC	W	CP
FT1: Melodic direction						
1	0, 1, 1, 1	1	1	1	1.0	2
2	1, -1, -1	1	1	1	1.0	
FT2: Melodic leap						
1	0, 2, 2, 1	1	1	1	1.0	1
2	2, 2, 1	1	1	1	1.0	2
FT3: Melodic inter-onset interval						
1	120, 120, 120, 120	1	1	1	1.0	1
2	120, 120, 120	1	1	1	1.0	2
FT4: Melodic duration						
1	120, 120, 120, 120	1	1	1	1.0	1
2	120, 120, 120	1	1	1	1.0	2

Table 18: LTM after adaptation to meme 2 (Figure 29).
(i = feature line index, FS = feature substring, DFC = date of first contact, DLC = date of last contact, NOC = number of contacts, W = weight, CP = connection pointers)

Here all the new FSs are different from the previous ones and are stored in new FLs in the corresponding FTs. Now the FS of index 1 in FT1 points (CP) to the index 2. Differently from the other FTs, this information represents the fact that FS of index 2 comes after the FS of index 1. This is how iMe roughly keeps track of the sequence of memes to which the agents are exposed. The CP of the other FTs still point to the index in FT1 that connect the elements of the meme to which the LTM is being adapted. The weights of the new memes are set to 1.0 as previously.

The same process is repeated with Meme 3 (Figure 30 - Table 19)



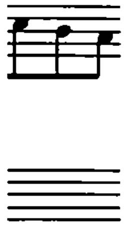
Feature substrings:
mdi: -1, 1, 1, 1, 1, 1
mle: 2, 2, 1, 2, 2, 2
mii: 120, 120, 120, 120, 120, 120
mdu: 120, 120, 120, 120, 120, 120

Figure 30: Meme 3.
(mdi = melodic direction, mle = melodic leap, mii melodic inter-onset interval, mdu = melodic duration)

i	FS	DFC	DLC	NOC	W	CP
FT1: Melodic direction						
1	0, 1, 1, 1	1	1	1	1.0	2
2	1, -1, -1	1	1	1	1.0	3
3	-1, 1, 1, 1, 1, 1	1	1	1	1.0	
FT2: Melodic leap						
1	0, 2, 2, 1	1	1	1	1.0	1
2	2, 2, 1	1	1	1	1.0	2
3	2, 2, 1, 2, 2, 2	1	1	1	1.0	3
FT3: Melodic inter-onset interval						
1	120, 120, 120, 120	1	1	1	1.0	1
2	120, 120, 120	1	1	1	1.0	2
3	120, 120, 120, 120, 120, 120	1	1	1	1.0	3
FT4: Melodic duration						
1	120, 120, 120, 120	1	1	1	1.0	1
2	120, 120, 120	1	1	1	1.0	2
3	120, 120, 120, 120, 120, 120	1	1	1	1.0	3

Table 19: LTM after adaptation to meme 3 (Figure 30).
(i = feature line index, FS = feature substring, DFC = date of first contact, DLC = date of last contact, NOC = number of contacts, W = weight, CP = connection pointers)

... and with meme 4 (Figure 31).



Feature substrings:
mdi: 1, -1, -1
mle: 1, 1, 2
mii: 120, 120, 120
mdu: 120, 120, 120

Figure 31: Meme 4.
(mdi = melodic direction, mle = melodic leap, mii melodic inter-onset interval, mdu = melodic duration)

The novelty here is that the FSs for melodic direction, inter-onset interval and duration had already been stored in the LTM. Only melodic leap has new information and, as a result a new FL was added to FT2 and not to the other FTs. The weights of the repeated FSs were increased by 0.1, which means that the relative weight of this information increased if compared to the other FSs.

Every time a FS is sent to the agent's memory, the weight of this FS is increased by a constant factor (e.g., f = 0.1). On the other hand, given a simulation with multiple cycles, if at the end of each cycle a FS is not sent to the agent's memory, the weight of the FS is decreased by another constant factor, which represents the fact that the agent forgets this FS. The latter case does not happen in this example because we are considering that the simulation is being executed in one single cycle.

i	FS	DFC	DLC	NOC	W	CP
FT1: Melodic direction						
1	0, 1, 1, 1	1	1	1	1.0	2
2	1, -1, -1	1	1	2	1.1	3
3	-1, 1, 1, 1, 1, 1	1	1	1	1.0	2
FT2: Melodic leap						
1	0, 2, 2, 1	1	1	1	1.0	1
2	2, 2, 1	1	1	1	1.0	2
3	2, 2, 1, 2, 2, 2	1	1	1	1.0	3
4	1, 1, 2	1	1	1	1.0	2
FT3: Melodic inter-onset interval:						
1	120, 120, 120, 120	1	1	1	1.0	1
2	120, 120, 120	1	1	2	1.1	2, 2
3	120, 120, 120, 120, 120, 120	1	1	1	1.0	3

i	FS	DFC	DLC	NOC	W	CP
FT4: Melodic duration						
1	120, 120, 120, 120	1	1	1	1.0	1
2	120, 120, 120	1	1	2	1.1	2, 2
3	120, 120, 120, 120, 120, 120	1	1	1	1.0	3

Table 20: LTM after adaptation to meme 4 (Figure 31).
(i = feature line index, FS = feature substring, DFC = date of first contact, DLC = date of last contact, NOC = number of contacts, W = weight, CP = connection pointers)

Finally, the LTM receives the last meme (Figure 32) and is adapted to it (Table 21).

Feature substrings:
mdi: -1, 1, -1, -1, -1
mle: 2, 2, 2, 2, 1
mii: 120, 120, 120, 120, 120
mdu: 120, 120, 120, 120, 480

Figure 32: Meme 5.
(mdi = melodic direction, mle = melodic leap, mii melodic inter-onset interval, mdu = melodic duration)

i	FS	DFC	DLC	NOC	W	CP
FT1: Melodic direction						
1	0, 1, 1, 1	1	1	1	1.0	2
2	1, -1, -1	1	1	2	1.1	3, 4
3	-1, 1, 1, 1, 1, 1	1	1	1	1.0	2
4	-1, 1, -1, -1, -1	1	1	1	1.0	
FT2: Melodic leap						
1	0, 2, 2, 1	1	1	1	1.0	1
2	2, 2, 1	1	1	1	1.0	2
3	2, 2, 1, 2, 2, 2	1	1	1	1.0	3
4	1, 1, 2	1	1	1	1.0	2
5	2, 2, 2, 2, 1	1	1	1	1.0	4
FT3: Melodic inter-onset interval						
1	120, 120, 120, 120	1	1	1	1.0	1
2	120, 120, 120	1	1	2	1.1	2, 2
3	120, 120, 120, 120, 120, 120	1	1	1	1.0	3
4	120, 120, 120, 120, 120	1	1	1	1.0	4
FT4: Melodic duration						
1	120, 120, 120, 120	1	1	1	1.0	1

i	FS	DFC	DLC	NOC	W	CP
2	120, 120, 120	1	1	2	1.1	2, 2
3	120, 120, 120, 120, 120, 120	1	1	1	1.0	3
4	120, 120, 120, 120, 480	1	1	1	1.0	4

Table 21: LTM after adaptation to meme 5 (Figure 32).
(i = feature line index, FS = feature substring, DFC = date of first contact, DLC = date of last contact, NOC = number of contacts, W = weight, CP = connection pointers)

8.2.4 Creativity

In iMe agents are also able to create new music through composing, solo-improvising and collective-improvising tasks. The creative model behind these tasks is in fact very similar and should be considered more "improvisational" than "compositional". Actually, the only substantial difference between a composition and the two forms of improvisation ("solo-improvise" and "collective-improvise") in the current implementation of iMe is the fact that compositions are not played in real-time; the agents generate a sequence of memes and the composition is delivered straight to the Music Store. Conversely, during improvisations, agents play newly generated memes before generating the next one.

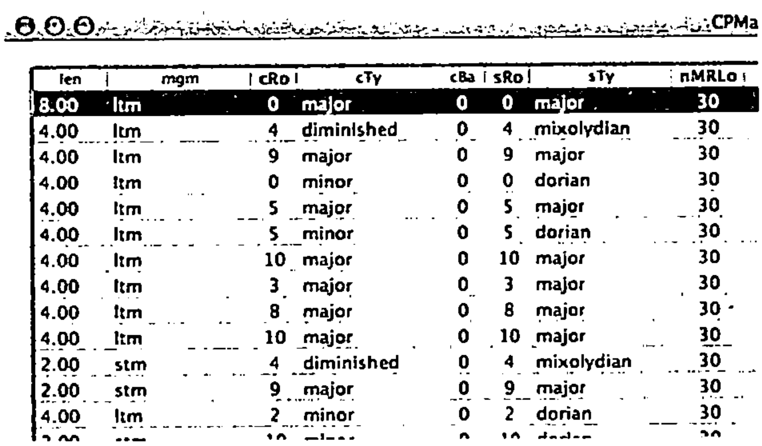
In addition, the generation of the memes is based on the agents' musical knowledge, i.e., on the framework defined in their memories through the many interconnected feature substrings and their recorded data (weights, connection pointers, etc.). However, once a meme is generated, it cannot be revisited; the agent cannot change its mind. Given these considerations, the words "composition" and "improvisation" are used interchangeably in this Section.

8.2.4.1 Compositional and Performance Maps

I designed and implemented a generative model called the "Compositional and Performance Map" (CPM) that must be defined by the system's user prior to the execution of generative tasks. Ideally, some of the parameters in the CPM (e.g., performance instructions) should have their data collected by the agents during the learning tasks and not be predetermined by the user. Yet, the interest of having a fixed CPM for a whole simulation is that it supports the use of the agents' compositions for further analysis.

In regard to some aspects, a CPM could be compared to a "lead sheet" in jazz music. In jazz, the musical score (lead sheets) generally only contains the tune (melody) and a sequence of chords that, together, define the global structure of the piece. Improvisers use these scores at the same time as a source for musical ideas and as a guide on top of which they adapt the musical structures that they generate "on the fly". When collectively improvising, the same structure is used by all musicians and serves as a common ground for their improvisation.

Figure 33 shows a fragment of iMe's window that controls the CPM:



len	mgm	cRo	cTy	cBa	sRo	sTy	nMRLo
8.00	ltm	0	major	0	0	major	30
4.00	ltm	4	diminished	0	4	mixolydian	30
4.00	ltm	9	major	0	9	major	30
4.00	ltm	0	minor	0	0	dorian	30
4.00	ltm	5	major	0	5	major	30
4.00	ltm	5	minor	0	5	dorian	30
4.00	ltm	10	major	0	10	major	30
4.00	ltm	3	major	0	3	major	30
4.00	ltm	8	major	0	8	major	30
4.00	ltm	10	major	0	10	major	30
2.00	stm	4	diminished	0	4	mixolydian	30
2.00	stm	9	major	0	9	major	30
4.00	ltm	2	minor	0	2	dorian	30

Figure 33: A CPM excerpt.
(len = length, mgm = meme generation mode, cRo = chord root, cTy = chord type, cBa = chord bass, sRo = scale root, sTy = scale type, nMRLo = non-melody range location)

Each column in Figure 33 corresponds to a parameter ("constraint") that addresses a particular aspect in improvisation: the generation of new musical ideas (memes) and the performance of these ideas. In fact, one could implement in a CPM any musical aspect belonging to these two areas. Examples of generative parameters include the mode of meme generation (p. 191), meme transformations (e.g., inversions, mirroring, transpositions), local chords and scales, etc. Performance parameters would be, for instance, the ratio between the intensity of the left hand (non-melodic notes) and the right hand (melodic notes), shifts for note onsets, and the speed of execution. Instructions regarding performance only affect the MIDI information before it is sent to the MIDI port and is not stored with the composition.

The constraints currently implemented in iMe are:

Length: double "q" that controls the duration of the constraint, where q = 1.00 represents a half-note

Meme generation mode: string that controls the mode of meme generation. Valid values are LTM, STM and MemeArray

Scale root: integer (values between 0 and 11) that controls the root of the scale played by the right hand (melodic notes)

Scale type: string that controls the "type" of the scale played by the right hand (melodic notes). Values can be chromatic, whole note, major, ionian, dorian, phrygian, lydian, mixolydian, natural minor, aeolian, and locrian.

Chord root: integer (values between 0 and 11) that controls the root of the chord played by the left hand (non-melodic notes)

Chord type: string that controls the "type" of the chord played by the left hand (non-melodic notes). Values can be major, minor, augmented, and diminished.

Chord bass: integer (values between 0 and 11) that controls the bass of the chord played by the left hand (non-melodic notes).

Chord added notes: array of integers (values between 0 and 11) that controls the added notes to the chord played by the left hand (non-melodic notes).

Melody range location and melody range length: integers that control the range of the notes played by the right hand (melodic notes). The resulting note ranges must be valid MIDI note values (0-127)

Non-melody range location and non-melody range length: integers that control the range of the notes played by the left hand (non-melodic notes). The resulting note ranges must be valid MIDI note values (0-127)

Melody non-melody velocity ratio: float that controls the ratio of the velocity (intensity) between the left and the right hands

Melody velocity shift: float that controls velocity (intensity) shifts for notes played by the right hand

Melody duration shift: float that controls duration shifts for the notes played by the right hand

Non-melody velocity shift: float that controls velocity (intensity) shifts for notes played by the left hand.

Non-melody duration shift: float that controls duration shifts for the notes played by the left hand

Each constraint has a duration variable ("length"). Therefore, when the agent generates a new meme, it checks its position (compositional pointer) in the CPM and applies the rules of the current constraint.

8.2.4.2 Compositional Tasks

As mentioned above, agents can perform three types of generative tasks: composing, solo-improvising and collective-improvising. Solo-improvisations will be initially explained, as the other two tasks are variants of it.

a) Solo-Improvisation

The execution of solo-improvisations requires a CPM and that the agent's memory (STM and LTM) is not empty. Being a task performed in real-time, the rest of the simulation is paused until the solo-improvisation is finished. The algorithm of this task follows these steps:

1. Generate a new meme

Memes are generated according to the "meme generation mode" defined in the current constraint. There are three types of generative modes:

A) LTM: This mode is the more complex of the three meme generation modes and uses the information stored in the agent's long-term memory, hence the name LTM. In this case, the agent must recombine the FSs stored in the several FTs in order to generate a new meme.

(i) The process starts with the agent choosing from the first Feature Table (FT1):

a) If the meme being generated is the very first meme of the composition, the agent chooses one FS from the first Feature Table (FT1) based on the distribution of probability of the **weights** of the FSs in that table,

b) If the meme being generated is not the first meme of the composition, the agent takes the FS that was chosen from FT1 in the previous meme and, based on the distribution of probability of the **connection pointers** of this FS, it chooses a FS from FT1 for the new meme. Remember that a CP in FT1 points to (an array of) FSs in the same FT1 that come after that particular FS. Once all CPs are stored sequentially for every FS, we are able to extract stochastic rules that guide the generation of the next FS. It is also possible to know which CPs are newer (and supposedly more relevant) and which are older (supposedly less relevant) in any FL, looking at the order they appear in the CP array. This information can be used to modulate the generation of the FSs by giving a higher weight to more recent CPs in the CP array.

(ii) Once the first FS from FT1 is chosen, the agent

a) Prepares an array of FSs at the other FTs to which the FS chosen at step "ii" above points at, and

b) Chooses a FS for all the other FTs LTM according to the distribution of probability of the **weights** of the array of FSs prepared in step "ii-a" above.

At the end of this algorithm, the "skeleton" of the first meme is ready and is represented as a series of "n" FSs, where n = number of FTs in the LTM.

B) STM: In this meme generation mode, agents use the information stored in the agent's short-term memory. As mentioned above, the STM is a queue of the last "n" (n = integer defined by the system's user) memes brought to the agent's memory. Therefore, the STM can potentially store memes that:

- Were previously generated by the agent,

- The agent's memory received in any of the other tasks, or

- Were played by a human performer and perceived by the agent during a collective-improvising task.

In STM meme generation mode, agents randomly choose from the memes stored in their short-term memory. The generation algorithm can modulate this choice by giving a higher weight to more recent memes in the STM queue.

C) Meme-array: Every time a meme is generated, it is stored in a meme array of a particular composition. Within the meme-array mode of meme generation, agents randomly choose from the memes stored in this array. Again, the generation algorithm can modulate this choice by giving a higher weight to the memes that were generated more times or that have been generated more recently.

2. Adapt the memory with the newly generated meme.

Once the new meme is generated, it is immediately sent to the memory where

- It will occupy a new position in the STM meme queue, and

- It will transform the LTM according to the algorithm described above (p. 181).

3. Generate notes and adapt the meme to the rest of the CPM

Up to this moment a meme is an array of FSs. The "actual notes", given the previously generated notes, are then calculated and the other variables of the current CPM's constraint are applied.

4. Play the meme

The block of notes computed in the previous steps is sent to a playing buffer that sends MIDI information to the MIDI port.

5. Repeat previous steps until the end of the CPM.

Wherever the number of notes in the playing buffer, is equal to or less than "x" (where x is an integer defined by the user), a new meme is generated (go to step 1) until all the constraints of the CPM are satisfied.

6. Deliver the improvisation to the Music Store

At the end of the improvisation the new music is stored in the Music Store.

b) Collective-Improvisations

Collective-improvisations are very similar to solo-improvisations. The main difference is that in collective-improvisations an agent plays along with a human being. This task was implemented as two different tasks (a listening task and a solo-improvisation task) running in separate threads. Memes are generated as in solo-improvisations and both agent and human improviser follow the same CPM. Because the two threads run in parallel, the agents' memory is equally affected by the memes they generate as well as by the memes they perceive from the musical data originated by the human improviser. At the end of the collective-improvisation, both the agent's newly generated music and the music played by the human performer are delivered to the Music Store.

A graphical interface (Figure 34) shows the current position (constraint) in the CPM, which the agent is performing during a collective-improving task. This feature acts as visual cues for the human performer.

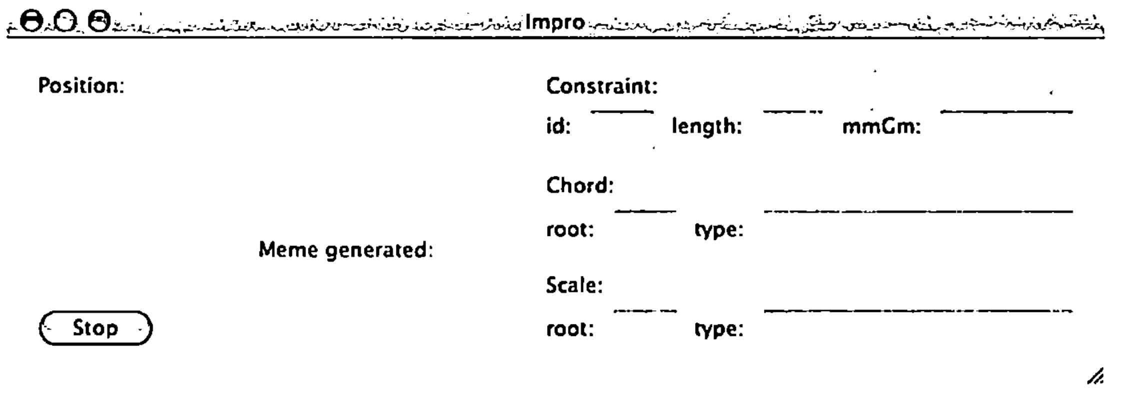


Figure 34: Improvisation window.
(id = constraint number, mmGm: meme generation mode)

c) Compositions

Finally, the last type of generative task is the composition task. As mentioned above (p. 187), during composition tasks agents follow the same algorithm described for improvisation tasks. The difference is that compositions are not played in real-time. Once all the constraints of the CPM are satisfied, the new composition is simply sent to the Music Store.

An example of composition

In order to illustrate the compositional process and how the generation of new memes occur, especially within the LTM mode of generation, the following paragraphs give an example of a short composition, which, for the sake of clarity, will include only a melody. As explained in the previous paragraphs, compositions follow the constraints specified in CPMs. The agent "checks" the current constraint and generates a new meme according to the meme generation mode. The new meme is then adapted according to the other parameters of the same constraint. This process, illustrated in Figure 35, continues until all the constraints of the CPM are applied.

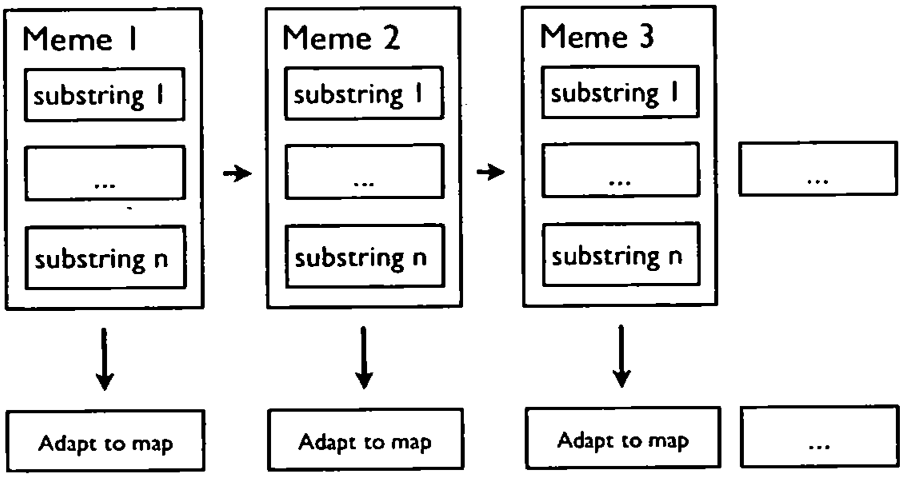


Figure 35: The process of meme generation and adaptation.

Figure 36 shows the CPM of this example, but only the parameters (length, meme generation mode, scale root and scale type) that are relevant for generating the notes of the melody.

#	len	mgm	sRo	sTy
1	4	ltm	0	major
2	4	ltm	0	major
3	4	meme-array	0	major

Figure 36: A CPM example.
(len = length, mgm = meme generation mode,
sRo = scale root, sTy = scale type)

The constraints are almost identical in this CPM: each one has 4 crotchets of duration, the meme generation mode is the "LTM" or the "meme-array" mode, the scale root is the pitch "C", and the scale type is "major". Let us assume that the agent's long-term memory is populated with the FTs show in Table 22. Once more, only the parameters (feature substring, weight, and connection pointers) that are essential for this example are displayed.

i	FS	W	CP
FT1: Melodic direction			
1	0, 1, 1, 1	2.0	2, 3

i	FS	W	CP
FT1: Melodic direction			
2	0, 1, 0, 1,	1.5	2
3	1, 0, 1, 0	1.2	
FT2: Melodic leap			
1	0, 2, 2, 1	1.8	1
2	0, 2, 0, 1	1.0	2, 1
3	2, 2, 1, 2	1.0	3
FT3: Melodic inter-onset interval			
1	240, 240, 240, 240	1.0	1
2	240, 240, 240, 240	1.1	2, 2
3	240, 240, 240, 240	1.0	3
FT4: Melodic duration			
1	240, 240, 240, 240	1.0	1
2	240, 240, 240, 240	1.1	2, 2
3	240, 240, 240, 240	1.0	3

Table 22: Agent's LTM.
(i = feature line index, FS = feature substring,
W = weight, CP = connection pointers)

Notice that the number 240 in the FTs that contain durational information (melodic inter-onset interval and melodic duration) corresponds to the duration of a crotchet and all substrings have the same length. This suggests that the memes that had been previously perceived by the agent had the same length. Normally, this would not be the case as the perceptive algorithm (Section 8.2.3, p. 175) allows the extraction of memes of different lengths.

a) Generating meme 1:

The agent generates the first meme. Constraint 1 of the CPM determines that the generation mode is "LTM". The algorithm chooses one FS from FT1 based on the distribution of probability of the weights of this table. Let us assume that the algorithm chooses the first substring (i = 1, FS = "1, 1, 1, 1"). The algorithm then chooses the other substrings from the other FTs. In FT2, substrings 1 and 2 have pointers to substring 1 in FT1. The algorithm chooses between these two substrings (1 and 2 from FT2) according to the distribution of probability of their weights. Let us assume that the algorithm chose i = 1 for FT2. The same process is repeated for FT3 and FT4. In FT3, substrings 1 and 2 have pointers to substring 1 in FT1 and, in FT4, only substring 1 has a pointer to substring 1 in FT1. At the end of this step, the agent would have chosen, the meme shown in Table 23.

FT	substring
1 - Melodic direction	0, 1, 1, 1
2 - Melodic leap	0, 2, 2, 1
3 - Melodic inter-onset interval	240, 240, 240, 240
4 - Melodic duration	240, 240, 240, 240

Table 23: Meme 1
(FT = Feature Table, substring = feature substring)

Next, the LTM should be reinforced by the newly chosen meme. The algorithm of the LTM transformation was described in Section 8.2.3.2 (p. 181) and this step will be skipped in this example, as we will consider that the weights of the FSs will remain constant. Meme 1 shown in Table 23 must then be adapted to the other parameters of Constraint 1, particularly to the scale root and scale type. iMe has a reference pointer ("rp") that keeps two variables, a pitch ("rp.pitch") and a position ("rp.position"), and helps to translate the meme into "actual notes". At the beginning of the composition, rp.pitch = 60 and rp.position = 0. By adding the numbers of meme 1 (FT1 and FT2), four notes are calculated: 60, 62, 64, and 65. These numbers are MIDI note numbers and correspond to pitches C, D, E and F. At this moment, rp.pitch = 65 and rp.position = 4. Melodic inter-onset intervals and durations are all the same (240). As these pitches belong to the scale of C major, no further adaptation is necessary and the new melodic segment is completed, as shown in Figure 37.

Figure 37: Melodic segment (Meme 1)

b) Generating meme 2:

Now, a new meme must be generated, i.e., four new substrings must be chosen from each FT. Given that rp.position = 4, the corresponding constraint in the CPM is the second one, which also defines that the meme generation mode is the "LTM". As explained in "Solo-Improvisation" above (p. 191), from the second meme onwards, the substring from FT1 is chosen from its connection pointers. Notice that FS 1 in FT1 points to FSs 2 and 3. These pointers (2 and 3) only appear once, which means that they are evenly distributed (50% of probability each). Let us assume that the algorithm chose FS 2 (substring = "0, 1, 0, 1"). Next, the other three substrings in the remaining FTs must be chosen following the same algorithm described for meme 1. Let us assume that the algorithm chose the second substring for each one of these tables. At the end of this step, the agent would have chosen, the meme shown in Table 24.

FT	substring
1 - Melodic direction	0, 1, 0, 1
2 - Melodic leap	0, 2, 0, 1
3 - Melodic inter-onset interval	240, 240, 240, 240
4 - Melodic duration	240, 240, 240, 240

Table 24: Meme 2
(FT = Feature Table, substring = feature substring)

Given that rp.pitch = 65 and rp.position = 4, by adding the numbers of the meme in FT1 and FT2, four notes are calculated: 65, 67, 67, 68, or pitches F, G, G, G#. Constraint 2 in the CPM defines that the scale is C major and G# does not belong to this scale. The algorithm forces an adaptation to the nearest pitch belonging to this scale. Let us assume that the pitch chosen is A. At this moment, rp.pitch = 69 and rp.position = 8. The new melodic segment is F, G, G, A, shown in Figure 38.

Figure 38: Melodic segment (Meme 2)

b) Generating meme 3:

The agent looks at the CPM again to check if there is still any constraint left and finds Constraint 3. In this case, the meme generation mode is "meme-array", which means that the algorithm will randomly choose from the previous generated memes. Let us assume that in this case, the algorithm chooses meme 2. Now, meme 3 has the same FSs as meme 2, shown in Table 25:

FT	substring
1 - Melodic direction	0, 1, 0, 1
2 - Melodic leap	0, 2, 0, 1
3 - Melodic inter-onset interval	240, 240, 240, 240
4 - Melodic duration	240, 240, 240, 240

Table 25: Meme 3
(FT = Feature Table, substring = feature substring)

Given that rp.pitch = 69 and rp.position = 8, the new notes are calculated: 69, 71, 71, 72, or A, B, B, C. These pitches belong to the scale of C major, defined in Constraint 3 and, therefore, no other adaptation is necessary. At this moment, rp.pitch = 72 and rp.position = 12. The new melodic segment is shown in Figure 39.

Figure 39: Melodic segment (Meme 3)

The agent checks the CPM again and finds that there is no other constraint left at rp.position = 12. The composition is complete and shown in Figure 40:

Figure 40: Final composition

8.2.5 Architecture, Interface and Functionalities

iMe was almost entirely implemented with Apple's Cocoa libraries and the Objective-C language, which address the user interface and facilitate the design of new simulations. A converter between MIDI and text files developed by Van Oostrum (1995) was also used for reading and listening tasks as well as the Synthesis ToolKit in C++ (STK) developed by Cook and Scavone (2008) for real-time MIDI.

iMe's main components are the Simulation, Agent, and Memory classes (Figure 41). Each iMe document has one Simulation object that has (among other objects) an array of agents, a Config (defines common attributes such as the STM size, MIDI ports, etc.) and a MusicStore (stores the musical pieces, etc.) object. The Agent class has a Percept object (controls the perception of the musical flow, etc.), a Memory object (stores the agents' musical knowledge, etc.), a Creator object (responsible for the generation of new musical material, etc.), and an array of Task (reading, listening, etc.) objects. The Agent class also controls a number of the agents' abilities (e.g. reading, listening to music, etc.). Finally, the Memory class stores, among other variables, the long- term and the short-term memories and controls the memory ability of adapting itself to the incoming musical memes.

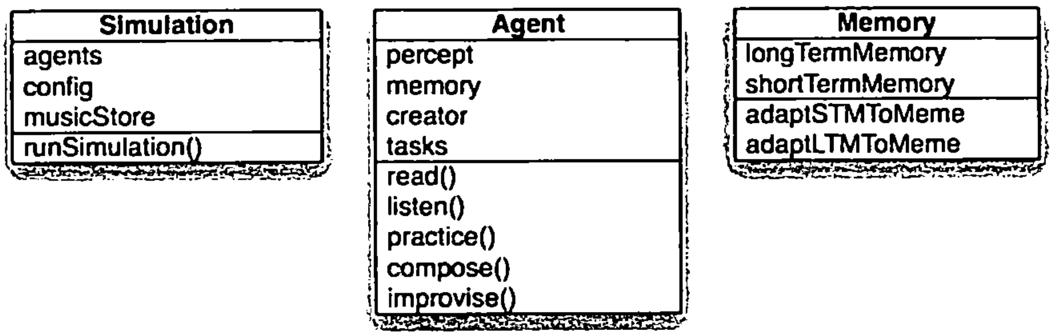


Figure 41: iMe's main classes

Within traditional menu items (File, Save, Open, etc.) simulations and all related data (agent's memories, system configuration, etc.) can be saved to computer files that can be subsequently opened, tested, modified, etc. Figure 42 shows iMe's Main Menu and, more specifically, the File menu, which contains some of these commands.

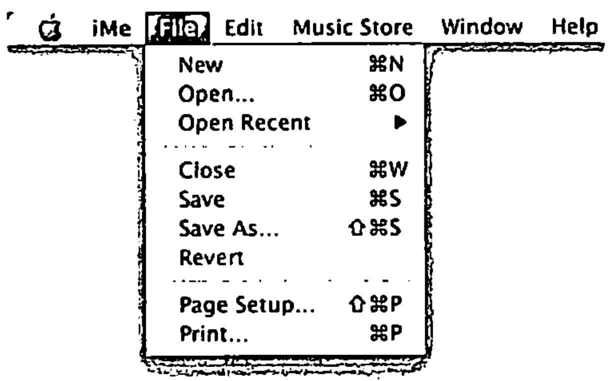


Figure 42: Main Menu

The Main Window (Figure 43) is the centre of control of most of the operations in iMe. On the lower left part of the window, the Agents' List shows the agents of the system. In Figure 43, there is only one agent, named "G8", which has a mark on the "active column" ("A"). Below this list a series of buttons allows the inclusion ("+"), removal ("-"), import ("i") and export ("e") of agents. When pressed, the button "d" (drawer) shows the Tasks Pane (Figure 44); the button "L" shows the log window of the currently selected agent. There are also buttons for accessing the Config Window ("Config") and the Compositional and

Performance Map ("C&P Map"). On the right hand side of these two buttons there is the "Go" button and a text field (black rectangle in Figure 43) that shows the cycles during the execution of the simulation.

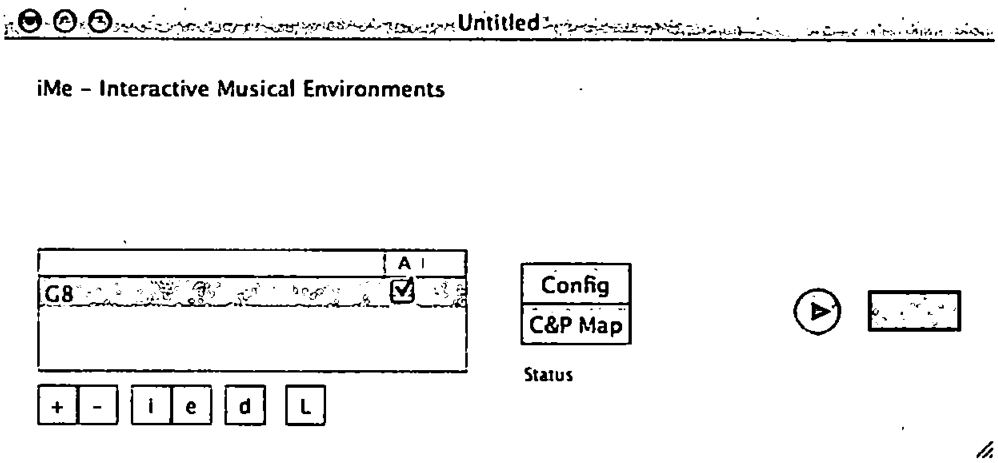


Figure 43: Main Window

Figure 44 shows the Tasks Pane that is opened when one agent is selected on the Agents' List or the button "d" is pressed. The data displayed in this Pane refers to the selected agent. A few buttons control the addition ("a"), removal ("r"), import ("i") and export ("e") of tasks for each one of the agents. The button "c" clears the agents' memory. Each task is displayed in individual rows in the tasks list together with the following information: cycle of execution ("Cycle"), number of times the task will be executed ("#Tasks"), the task type ("Type"), number of times practice tasks are executed ("#Prac"), the done or undone status ("D"), the values for the coefficients of attentiveness, character and emotiveness ("CfA", "CfC", and "CfE"), and parameters for the execution of the goals (e.g., Composer, Genre - p. 174). The mentioned coefficients have been designed to modulate the transformation of the memories but are not being used in the current version of iMe, reason why they have not been covered in this Thesis. Below the tasks list, a summary of the agents' memory content is displayed. In Figure 44 the agent's memory is empty.

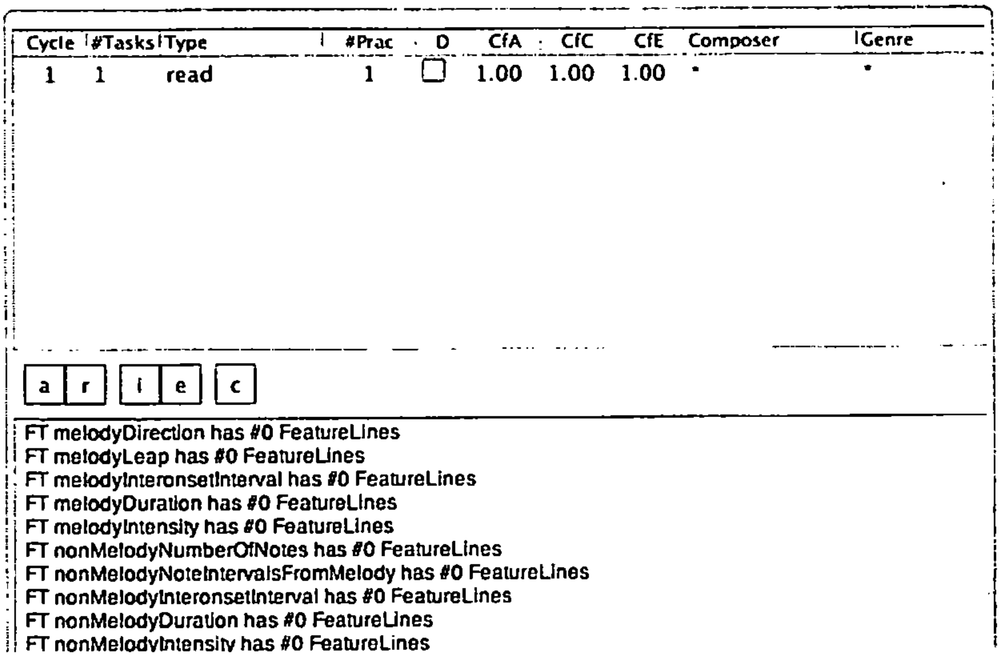


Figure 44:Tasks Pane

In addition to the above-mentioned windows, iMe has a number of others that control the general configuration of the system, including parameters for the memories (Figure 45), filters for feature extraction (Figure 46), and segmentation (Figure 47). Another resource, the Compositional and Performance Maps Window has been shown in section 8.2.4.1 (p. 187).

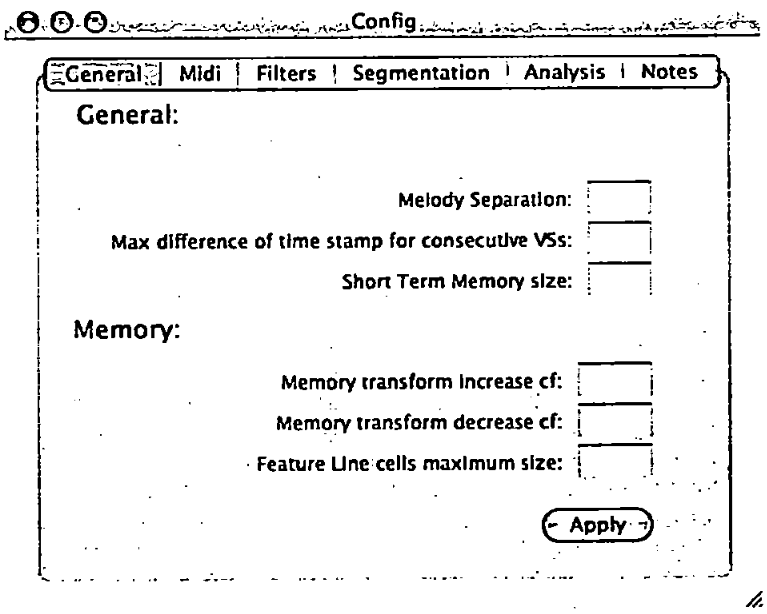


Figure 45: Config (General)

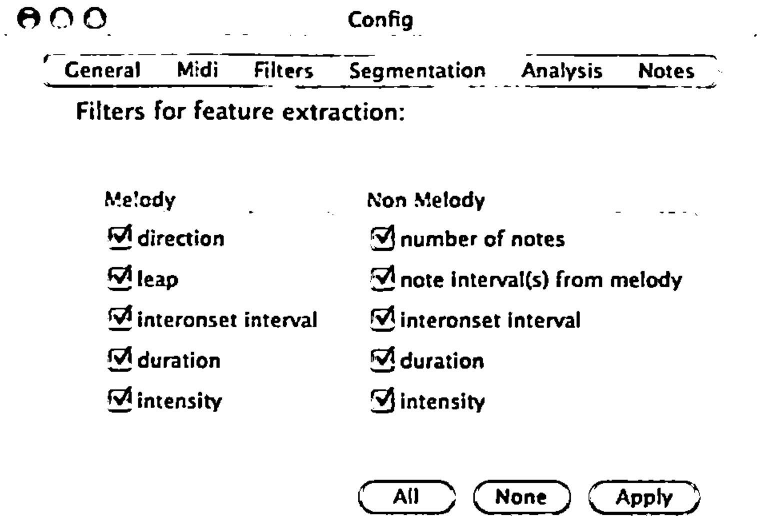


Figure 46: Config (Filters)

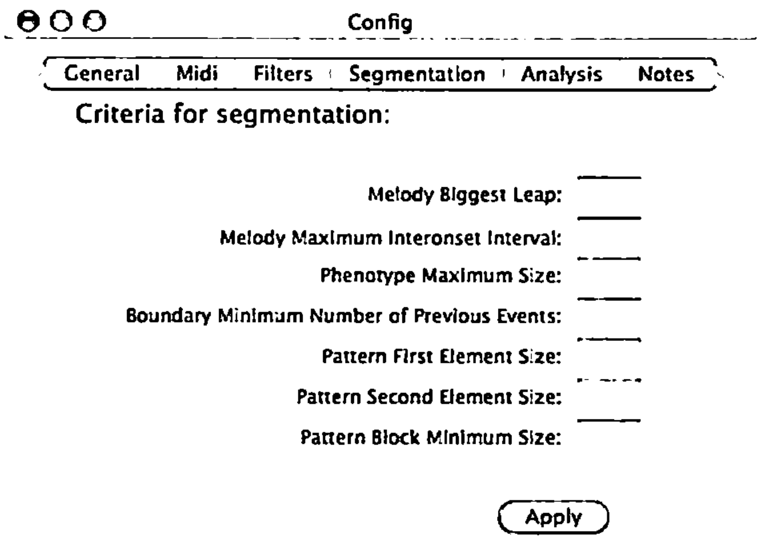


Figure 47: Config (Segmentation)

Finally, the Music Store Window (Figure 48) provides functionalities for the addition ("+"), and removal ("-") of pieces, the import ("i"), export ("e") and cleaning ("c") of entire music libraries. The "Gen MidiFile" generates MIDI files from the selected pieces of the Music Store. Pieces can be heard by pressing the "P" button; the slider on the right hand side controls the tempo of execution. In order to facilitate music input, a recorder was implemented directly in the

Music Store. The button "R" controls the recording of new music by human performers connected to the computer MIDI ports.

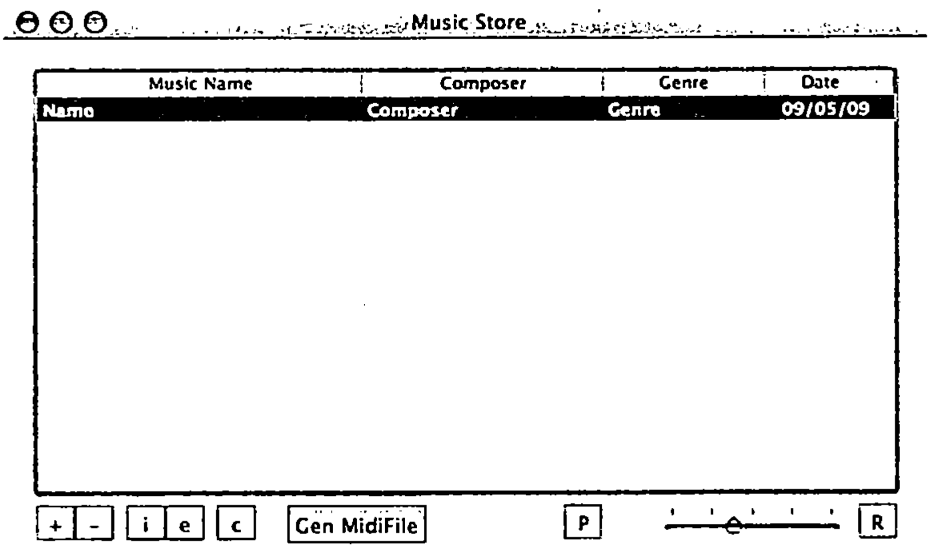


Figure 48: Music Store

8.2.6 Simulation

A great portion of this Chapter has been dedicated to describing the several complex elements that comprise iMe's structure. Designing a simulation, on the other hand, is much simpler to explain; it just involves defining (i) a number of agents, (ii) a number of tasks for each agent, and (iii) some initial music material for the Music Store. If there are any generative tasks, a CPM must also be designed.

Designing simulations is straightforward and a carefully crafted design can address complex musicological questions. The next Section describes some possible scenarios and questions that can be addressed with iMe.

iMe's potentials were explored and tested with the OMME's main objectives (p. 133) in mind. A series of experiments were performed in two general fields: musicology and creativity. Some of these experiments are described in the following Sections.

8.3.1 Musicology

Section 8.2.3 above explained that the agents' memories, especially long-term memories, are subject to a process of continuous adaptation that starts with the perception of a musical stream through sensory filters that extract basic musical features. This process produces a parallel stream of data (feature stream) that is then segmented into musical memes, which represent the most basic unit of musical information for purposes of storage and transmission.

In fact, in iMe's model, musical memes are a group of substrings of the feature stream (feature substrings) stored in the LTM. This idea allows some particular operations such as the reinforcement of weights and of connection pointers. For instance, consider that an agent perceives the three successive memes shown in Figure 49 in traditional notation:

Figure 49: Examples of memes (traditional notation).

Melodic duration and melodic inter-onset intervals are the same in all these three memes. Melodic direction is the same in memes 2 and 3 but different in meme 1. Melodic leaps are all different in meme 1 (2212221), meme 2 (2212212) and meme 3 (4343212); numbers represent half steps. Given these examples, in the agent's LTM, melodic duration and melodic inter-onset intervals are represented by just one FS (in their respective FTs) while melodic direction is represented by two FSs and melodic leap is represented by three FSs. It is possible to see that the common FSs in these cases will be more strengthened and have more connection pointers to other FSs and, vice-versa, less common FSs will be less strengthened and have less connection pointers to other FSs.

The example above illustrates the fact that (i) the sequence of events in one piece and (ii) the order several pieces of music are perceived by the agents directly affect the agents' memories. Memory, in other words, is a consequence of the agents' musical experience. I am reinforcing this aspect because I understand that agents' memories could be efficiently used as an abstract computational model of any music, something that could be very helpful for musicology.

Traditionally, musicology focuses on structural elements of music: looking for patterns, defining structures in various levels and classifying these structures allows the understanding of music within common frameworks (e.g., harmony, rhythm, motifs). Alternatively, as seen in Section 3.1 (p. 48) some recent computer-based musical theories have already been proposed (Lerdahl *et al.*, 1983; Cambouropoulos, 1998) considering cognitive aspects of music interaction. I believe that the OMME and iMe are a contribution in this trend.

Even though, as explained above, the agents' memories have a specific meaning, they would only be useful as a musicological model if it were possible to evaluate them in order to check for similarities and/or differences. This is why I implemented a measure of distance, described in the next Section.

8.3.1.1 Measure of Distance

I have explained in Section 7.2.4 above (p. 148) that in RGeme the measure of distance between two rhythmic memes was obtained using the "block distance". This measure allows the comparison of two sequences of the same length. To overcome this limitation, in cooperation with other researchers (Martins *et al.*, 2005) I developed an algorithm to measure the similarity between any sub-sequences in a general rhythmic space using a structure called Similarity Coefficient Vector. This subject was addressed in Section 3.1 "Music Perception", above.

In iMe, comparing two memory objects is much more complex than the previous cases because memories store strings of potentially any length that represent not only rhythm but also a number of other musical features. As an initial attempt to measure the distance between the memories of two agents, the following algorithm was implemented.

The precondition for this measurement is that the two memories have the same structure, i.e., the same number and type of feature tables. Feature tables are compared in pairs: FTn of agent "A" (A.FTn) is compared to FTn of agent "B" (B.FTn) (where, e.g., n = "melodic direction"). The distance between A.FTn and B.FTn is the sum of all the differences of weights of their FSs.

I) Initially, the algorithm calculates the difference of the weights of all feature lines where

(i) A.FTn.FLo = FL of index "o" in FT "n" of agent "A" and

(ii) B.FTn.FLp = FL of index "p" in FT "n" of agent "B",

And

(Feature substring of A.FTn.FLo) = (Feature substring of B.FTn.FLp)

II) After this, where (i) (FS of A.FTn.FLo) does not have a correspondent in B.FTn, and vice versa, (ii) (FS of B.FTn.FLp) does not have a correspondent in A.FTn, the difference of weight is the weight of the existing FSs.

The distance between A.FTn and B.FTn is the sum of all the differences of weights calculated in "I" and "II". The differences of all FTs are then combined using Euclidean distance to produce a measure of distance between the memories of agent "A" and agent "B".

8.3.1.2 Scenarios

With these tools (memory model and measure of distance), it is possible to draw a number of scenarios that would enable the analysis of particular musicological issues. These scenarios imply that the agents' memories have no information at the start of the simulations.

In the following paragraphs five of these scenarios are described where a different number of agents and musical pieces are involved. In these cases, the agents are (separately) interacting with the environment via the pieces available in the Music Store; but they do not communicate with each other as they only execute reading tasks. The general purpose of these types of simulations is to measure the distance between the styles of musical pieces and/or the evolution of the agents' musical styles. It is necessary to emphasise, however, that only tasks that involve the creation of new musical material (composition, improvisation) would entail the transmission of knowledge between the agents, which is not the case in the following examples.

a) One Agent, One Piece, One Task

If an agent executes only one reading task, at the end of the simulation the agent's memory represents at the same time (i) the musical style of the agent and (ii) the musical style of the piece.

b) One Agent, Two (Or More) Pieces, Two (Or More) Tasks

If the agent executes more than one reading task, regardless if it uses the same or different pieces, the memory will no longer represent the style of the piece but only the agent's evolved style.

c) Two Agents, One Piece, One Task

If agent "A" and agent "B" both read the same piece, and all the other variables in the system are set to the same values, the distance between their memories will be zero.

d) Two Agents, Two Pieces, One Task

If agent "A" reads piece "x" and agent "b" reads piece "y" ($x \neq y$), and at the end of the simulation, we measure the distance between their memories, the result will represent (i) the difference of musical style between agent "A" and agent "B", and (ii) the difference of musical style between piece "x" and piece "y".

e) Two Agents, Two (Or More) Pieces, Two (Or More) Tasks

Finally, if agent "A" and agent "B" execute each one more than one reading task, the difference between their memories will no longer represent the differences of style between them or between the pieces but only the difference between the evolved styles of agent "A" and agent "B".

During iMe's implementation, a series of simulations were done for each one of the previous scenarios and with different musical data sets. It was possible, for instance, to measure the distance of individual musical pieces (e.g. 2 two-voice inventions by J.S. Bach or nocturnes by F. Chopin) by comparing the memories of two different agents, after the execution of only one reading task each. At the end of one simulation, the resulting number (measure of distance) symbolized an abstract representation of the distance between the musical styles under the perspective of two listeners. Instead of listing in this Thesis a series of distance measures, however, in the following section two cases have been selected and described in order to illustrate the potential of this methodology in the musicological realm. The detailed account on how to interpret the result is, however, beyond the scope of this Thesis. This would require systematic studies and simulations, as proposed in Section 9.2 - Recommendations for Future Work.

8.3.1.3 Two Case Studies

a) Two Agents, One Set of Pieces, 100 Tasks

Two agents, "A" and "B", were given 100 reading tasks each and a set of pieces, the 12 two voice inventions by J.S. Bach. Each agent executed only one task per cycle, totalling 100 cycles. Both agents could freely choose from any of the 12 pieces, at any time. At the end of each cycle, the distance between their memories was computed. These distances were plotted in the graph shown in Figure 50, where the horizontal axis represents the time cycles and the vertical axis represents the distances between their memories.

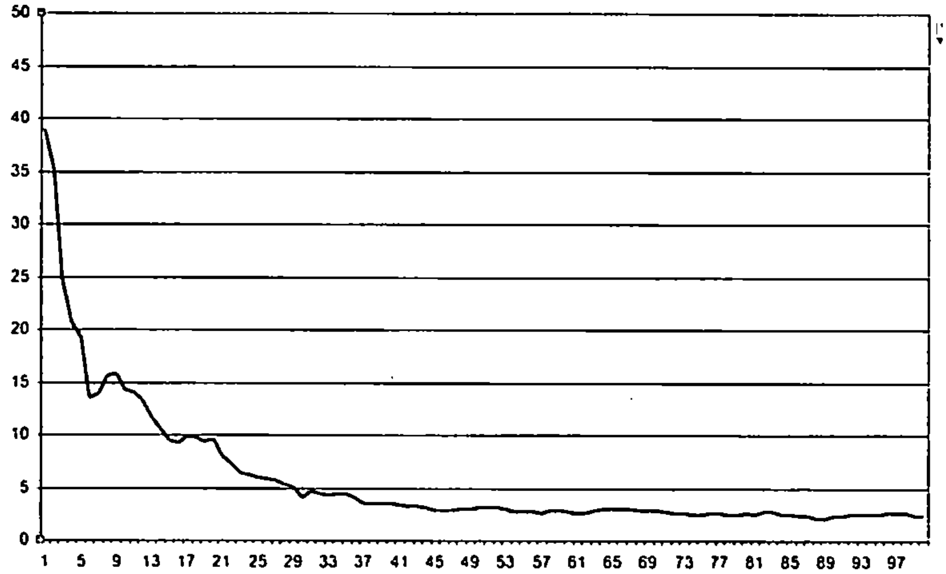


Figure 50: Style differences between two agents (same set of pieces).
(Axis x = time cycles, axis y = distance)

Analysis:

The graph of Figure 50 shows the decrease of the distances from 39 to values around 2.5. During the initial cycles, given the randomness of their choices, we expected that there would be major differences. After approximately 45 cycles, however, the difference stabilizes due to the fact that they were listening to the same set of pieces. It is not clear, however, why the differences stabilize around the value 2.5 and not zero. It would be reasonable to hypothesize that this number represents differences between all the pieces but it is not obvious how to demonstrate this.

b) Two Agents, Two Sets of Pieces, 100 Tasks

Two agents, "A" and "B", were given 100 reading tasks and two sets of pieces. Agent "A" should read only from set "a" and agent "B" should read only from set "b". Set "a" contained the 12 two voice inventions by J.S. Bach and set "b" contained 10 ragtime piano pieces. Each agent executed only one task per cycle, totalling 100 cycles. Both agents could freely choose from any piece of their respective sets. At the end of each cycle, the distance between their memories was computed. The results are shown in the graph of Figure 51:

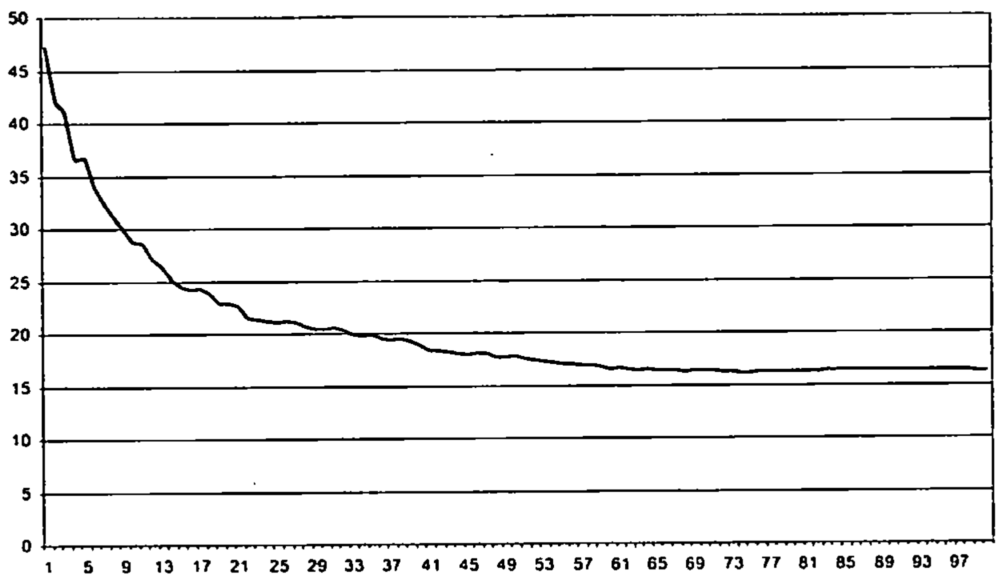


Figure 51: Style differences between two agents (different sets of pieces).
(Axis x = time cycles, axis y = distance)

Analysis:

The shape of this graph is very similar to the previous. In this case, differences started at 47.5 and stabilization was reached at around 16.5. It took a bit longer for the differences to get stabilized, around 65 cycles. In this case, it is clear that the number 16.5 represents the differences between the two sets of pieces.

8.3.2 Creativity

In addition to their potential in musicology, OMME-based systems can also give a significant contribution in the investigation of human creativity, and the interactivity between humans and machines. Section 8.2.4 ("Creativity") mentioned the fact that iMe was designed with a focus on an improvisational model of creativity and, in fact, the music generated by this system has indeed this character. During the series of simulations run during its development, iMe was experimented with many different musical databases, from long pieces to just a few notes, sometimes mixing completely different musical styles and textures and other times working with more consistent material. Different CPMs were also tried.

I can report that there were occasions where I was quite pleased with iMe's results while in other experiments it seemed the agent's compositions were not worth all the effort. To some extent, the remarks reported by Cope (1999) during his experiments with Emmy (p. 68) were replicated during iMe's development. Among these, in order to achieve a good sense of stability, the preparation of the musical database with a consistent material is paramount. Consistency here refers to music where specific parameters (e.g., texture, rhythm) do not vary too much.

A simulation that involves a collective-improvisation task can achieve this sort of consistency or stability. In fact, while a human performer improvises (and the agent listens to and learns from this improvisation), there is a natural concern with the fact that the emerging musical structures are conveniently exposed and well balanced among each other. One would expect that the agent should respond in a similar way.

In order to ultimately examine iMe's potentials in this area, the system was tested during a public performance. Other IMS have already been used in similar settings. Biles, for instance, uses GenJam (Biles, 1994) as a personal improvisational partner. I once had the opportunity to listen to his performance and, even though GenJam has a completely different approach compared to iMe, it achieved interesting results. The Continuator (Pachet, 2003) and OMax (Assayag *et al.*, 2006) have also being used in public performances, examples of which can be found in the Internet (Assayag, 2008; Pachet, 2008). These systems implement machine-learning techniques (Section 3.2.3, p. 54) that in some respects could be compared to some of iMe's solutions.

Therefore, in order to illustrate this Section, rather than presenting the score of an agent composition, the CD ("Appendices") that accompanies this Thesis includes a movie[7] of a public performance in which I played a piano improvisation with one of iMe's agents. The performance was part of the Peninsula Arts Contemporary Music Festival 2008 (ICCMR, 2008), a series of events that has been happening in February every year since 2005 at the University of Plymouth and is jointly organized by the Peninsula Arts and the Interdisciplinary Centre for Computer Music Research at the University of Plymouth.

As mentioned in sections 5.3 (Modes of Interaction (When To Play) - p. 101), and 5.4 (Musical Structure (What To Play) - p. 108), the design of an interactive performance involves the definition of a number of variables that affect what performers will play and how they will react to each other. In the case of the

[7] A copy of this movie has also been uploaded to http://www.youtube.com/marcgimenes

system iMe, different possibilities can be achieved by controlling the training material and the planning of an appropriate CPM. One could wish, for instance, to start the agent's memory from scratch, without any previous training. In this case, the agent would only learn from and respond to what happens during the performance. On the other hand, one would opt to train an agent with a specific set of pieces within a specific or several styles to be used during the performance.

Another variable that can be manipulated in order to efficiently design interactive performances in iMe is the mode of generation of new memes defined in the CPMs. The LTM mode of generation implies that long-term connections will be favoured. The STM mode of generation suggests that the most recently perceived (or generated) memes are the ones that the agent will use to generate the next meme in a composition. Within the third mode of generation (section 8.2.4.1, p. 187) agents reuse previously generated memes.

For the Plymouth performance, a special simulation was designed with one agent and two tasks: one reading task and one collective-improvising task. Initially, the agent's memory [8]is empty and, during the reading task, the agent reads a MIDI file containing only the melody of the musical piece *Stella by Starlight*, by Victor Young, in order to populate its memory with some material that could be used during the second task. *Stella by Starlight* is a traditional standard of the jazz repertoire, consisting of 32 bars, divided in 4 sections (A, B, C, and A'). Table 26 shows the chord progression of *Stella by Starlight* used in this performance:

---

[8] The memories after the first and the second tasks can be found in the CD that accompanies this Thesis (file "memories.xlsx")

Section	Chords			
A	E-7b5	A7b9	C-7	F7
	F-7	Bb7	Ebmaj7	Ab7
B	Bbmaj7	E-7b5, A7b9	D-7	Bb-7, Eb7
	Fmaj7	E-7b5, A7	A-7b5	D7b9
C	G		C-7	
	Ab7		Bbmaj7	
A'	E-7b5	A7b9	D-7b5	G7b9
	C-7b5	F7b9	Bbmaj7	

Table 26: *Stella by Starlight* chord progression.

During the second task, the agent collective-improvises following a CPM prepared with a sequence of 6 choruses of the piece. In the jazz jargon, one "chorus" corresponds to a complete cycle of the tune. Figure 52 shows an extract[9] of the CPM's window, where each row corresponds to one constraint and the columns correspond to the parameters of the constraints. For clarity, only the first seven parameters are shown (length, meme generation mode, chord root, chord type, chord bass, scale root, scale type). The other parameters (non-melody range location, non-melody range length, melody range location, melody range length, melody non-melody velocity ratio, melody velocity shift, melody duration shift, non-melody velocity shift, non-melody duration shift) are not shown in Figure 52. The letters (A, B, C, and A') on the right-hand side correspond to the sections of the piece; the first chorus starts at row 2:

[9] The full CPM can be found in the CD that accompanies this Thesis (file " cpm.xlsx")

len	mgm	cRo	cTy	cBa	sRo	sTy	
8.00	ltm	0	major	0	0	major	
4.00	ltm	4	diminished	0	4	mixolydian	A
4.00	ltm	9	major	0	9	major	
4.00	ltm	0	minor	0	0	dorian	
4.00	ltm	5	major	0	5	major	
4.00	ltm	5	minor	0	5	dorian	
4.00	ltm	10	major	0	10	major	
4.00	ltm	3	major	0	3	major	
4.00	ltm	8	major	0	8	major	
4.00	ltm	10	major	0	10	major	B
2.00	stm	4	diminished	0	4	mixolydian	
2.00	stm	9	major	0	9	major	
4.00	ltm	2	minor	0	2	dorian	
2.00	stm	10	minor	0	10	dorian	
2.00	stm	3	major	0	3	major	
4.00	ltm	5	major	0	5	major	
2.00	stm	4	diminished	0	4	mixolydian	
2.00	stm	9	major	0	9	major	
4.00	ltm	9	diminished	0	9	mixolydian	
4.00	ltm	2	major	0	2	major	
8.00	memeArray	7	major	0	7	major	C
8.00	memeArray	0	minor	0	0	dorian	
8.00	memeArray	8	major	0	8	major	
8.00	memeArray	10	major	0	10	major	
4.00	ltm	4	diminished	0	4	mixolydian	A'
4.00	ltm	9	major	0	9	major	
4.00	ltm	2	diminished	0	2	mixolydian	
4.00	ltm	7	major	0	7	major	
4.00	ltm	0	diminished	0	0	mixolydian	
4.00	ltm	7	major	0	7	major	

Figure 52: CPM of the public performance.
(len = length, mgm = meme generation mode, cRo = chord root, cTy = chord type, cBa = chord bass, sRo = scale root, sTy = scale type)

It is extremely difficult to precisely describe what happens in each and every moment of the concert. However, in order to understand the interactive processes involved in the simulation, Table 27 contains a description of the moments where it is possible to identify that particular musical structures are transmitted in both directions, from the agent to me, and vice-versa. I suggest that the movie contained in the CD ("Appendices") is watched following the descriptions contained in this table. The agent starts by playing only melodic elements that resembles the original tune and I start by playing some harmonic structures:

#	Time	Description
1	00:32	Start of the simulation. Agent plays first melodic notes. Marcelo plays chords.
2	00:56	Agent plays fast notes learned from Marcelo's performance.
3	01:10	Agent plays first harmonic elements learned from Marcelo's performance.
4	01:45	Marcelo responds to the agents' material
5	01:58	Marcelo introduces new melodic ideas
6	02:12	Agent responds to Marcelo's music structures and plays short, fast melodic sequences
7	02:33	Agent plays melodic and harmonic structures, Marcelo imitates.
8	02:41	Marcelo plays new chords. Public reacts positively.
9	02:50	A better sense of interactivity is achieved between the performers.
10	02:50	Agent takes the lead for some moments.
11	03:03	Marcelo plays scales and other new musical structures
12	03:28	Agents responds with short memes
13	03:31	Marcelo introduces new rhythmic material while agent continues with melodic elements
14	03:46	Agent uses meme from theme and then responds with Marcelo's musical structures (#13)
15	4:03	Agent's and Marcelo's improvisation intermingled.
16	4:26	Melodic dialogue between the two players.
17	5:28	Marcelo responds to the agent's short melodic memes
18	6:10	Marcelo plays descending chords on the left hand while agent plays short melodic memes
19	6:30	Agent "awakes" and plays fast melodic memes. Marcelo responds.
20	3:50	Agent responds to Marcelo fast melodic memes.
21	7:11	Marcelo stops and agent responds to Marcelo's structures
22	7:25	Agent stops playing. Marcelo prepares the end of the improvisation.
23	7:53	Marcelo plays last chord.

Table 27: Description of the improvisation.
(Time in minutes:seconds)

My evaluation of this performance is very positive. It was a personal challenge to leave the controlled setting of the laboratory and to introduce the system in an environment where potentially many problems could occur. The reaction of the public was positive as well; some people reported the perception that a real dialogue was happening. iMe also attracted the attention and was the subject of two publications by the media (Herald, 2008; Johnson, 2008).

8.4 Conclusion

This Chapter introduced the second implementation of an OMME-based system, the multi-agent musical system iMe (Interactive Musical Environments), especially designed to support interactivity from an improvisational point of view. Agents perform human-inspired activities and communicate between them and with external human performers. iMe covers most of the objectives defined by the OMME: it models human perception and cognition following theoretical models that explore cultural evolution and provides original tools for musicology, mainly focused on the ontogeny of music.

In iMe, intelligent agents, through musical information exchange, have the ability to perceive the environment, analyse it and take actions in order to alter the surrounding states as well as their own internal states: iMe also models music as an expression of human faculties. Developmental maps, the agents' ontogenies, can be designed directly via the user interface. As the system logs all the agents' activities, it is possible to evaluate the agent's musical evolution.

In addition to the applications in the musicological field, demonstrated in two case studies, iMe has also performed well in live performances. This fact was illustrated through a step-by-step analysis of a public performance in which I played a piano improvisation with one of the system's agents. This performance was part of the Peninsula Arts Contemporary Music Festival 2008, at the University of Plymouth, UK.

Chapter 9 Conclusion

The interest for the investigation introduced in this Thesis originates from the perception that during their lifetime human beings gradually develop their musical knowledge through countless interactions. This fact is especially noticeable when people, while studying music, start to replicate other peoples' styles, something I realized through a personal experience reported in the Introduction of this Thesis (p. 15). From this experience, I was impelled to explore the mechanisms that allow musical influence.

Among these mechanisms, people are born with the machinery, the perceptive and cognitive systems, which allow the acquisition of the musical knowledge. In fact, the human cognitive system (memory) is at the basis of music itself (Section 2.1, p. 32), viewed as the organisation of sounds, a typical and culturally conditioned intelligent human behaviour (Chapter 1, p. 23). In addition, humans are social beings, something that inevitably leads to the transmission of certain cultural traits from one mind to another, from one place to another. This fact fosters the emergence and dissemination of musical styles (Section 1.2, p. 25). From time to time new styles appear; some are more successful and live longer than others.

Having these considerations in mind, this Thesis proposes ("Introduction", p. 18) the achievement of three main objectives:

Objective 1: To use computational modelling to explore how music is perceived, cognitively processed and created by human beings;

Objective 2: To explore interactive musical systems as a method to model and achieve the transmission of musical influence in artificial worlds and between humans and machines; and

Objective 3: To experiment with artificial and alternative developmental musical routes in order to observe the evolution of musical styles.

In this Thesis, these objectives have been achieved through the proposition of a new paradigm for the design of computer interactive musical systems, the Ontomemetical Model of Music Evolution - OMME, aiming at artificially exploring the evolution of music centred on human perceptive and cognitive faculties. Not only this computational model has been proposed but, in order to demonstrate it and to achieve the objectives of this Thesis, two new systems, the Rhythmic Meme Generator (RGeme) and the Interactive Musical Environments (iMe) have been implemented as well.

Objective 1, concerned with the fact that music is essentially a human activity, has been specifically discussed in Chapter 2 (p. 31), where the human faculties pertaining to music have been addressed. From a computational perspective, the OMME particularly states that interactive musical systems must "regard music as an expression of human faculties" (OMME's Condition 2, Section 6.3.2, p. 134). The two above-mentioned systems, RGeme and iMe, fulfil Objective 1 specially through the modelling of perceptive (Sections 7.2.3 and 8.2.3), memory (Sections 7.2.3 and 8.2.3.1) and creative abilities (Sections 7.2.4 and 8.2.4).

Objective 2 regards the social characteristic of the human behaviour and the transmission of musical influence, something that the OMME undertakes under the concept of "interactivity", a subject comprehensively explored in Chapter 4 (p. 75) and Chapter 5 (p. 95). This study led to the definition of interactive musical systems as "computer-based systems that, through musical information exchange, have the ability to perceive the environment, analyse and take actions as to alter the surrounding states as well as to change their own internal states" (p. 93). The OMME adopts this notion of interactivity (Condition 1, Section 6.3.1, p. 133), as well as RGeme (Section 7.2.2 p. 142) and iMe (Section 8.2.2, p. 172) under the paradigm of intelligent agents (Section 3.3.3, p. 63).

Finally, Objective 3 was addressed within two major concepts, after which the (ontomemetical) model is named. The first one is memetics (Section 6.2, p. 128), so to speak, the "genetics of culture". It defines that culture evolution can be regarded as the exploration of an informational space where some basic elements (memes) are selectively transmitted from one brain to another, where they live in the form of neuronal configurations. The other major concept is ontogenesis (Section 6.1, p. 125), understood as a series of concatenated events, a temporal map that defines the development of individuals. OMME-based systems are therefore designed to artificially explore the evolution of music centred on the transmission of musical memes and, consequently, on human perceptive and cognitive faculties. OMME's Condition 3 (Section 6.3.3, p. 135) specifically states that in order to comply with the model, interactive computer musical systems must implement mechanisms that evaluate music evolution. RGeme (Section 7.3.4, p. 154) and iMe (Section 8.3.1, p. 207) also provide these mechanisms.

Summing up, the OMME defines that interactive musical systems must:

Condition 1 (Section 6.3.1, p. 133): Be interactive systems: interactive musical systems are specifically defined in this context as "computer-based systems that, through musical information exchange, have the ability to perceive the environment, analyse and take actions as to alter the surrounding states as well as change its own internal states" (p. 93).

Condition 2 (Section 6.3.2, p. 134): Regard music as an expression of human faculties: these faculties (especially memory) are at the foundation of the memetic transmission (p. 128).

Condition 3 (Section 6.3.3, p. 135): Implement mechanisms to evaluate music evolution. OMME aims at the exploration and understanding of music evolution.

These conditions and the objectives pursued in this Thesis keep pace with the general aims proposed for the OMME (p. 133), which are to:

Aim 1: Contribute to the understanding of natural phenomena such as human perception and cognition by computational modelling.

Aim 2: Contribute to the achievement of machine musicianship, and the interactivity between humans and machines.

Aim 3: Provide computational tools for musicology, mainly focused on theoretical models that explore cultural evolution.

To the best of my knowledge, as shown in Section 6.4 (p. 135), no computer system has ever been designed to specifically comply with the OMME's aims and conditions. RGeme and iMe are the first interactive musical systems to do so as indicated in Table 28, which uses the same criteria of the comparative table presented in Section 6.4 (Table 2, p. 138),

#	Feature	RGeme	iMe
1	Musical information exchange	Y	Y
2	Machine musicianship	Y	Y
3	Perception, analysis, action	Y	Y
4	Alteration of states (memory)	Y	Y
5	Human faculties (perception, cognition)	Y	Y
6	Musicology (musical evolution)	Y	Y

Table 28: Comparative Table of OMME-Based Systems
(Y = Yes)

RGeme pursues the three OMME's aims: (i) agents simulate human-like perceptive and cognitive faculties (Aim 1 - p. 145), (ii) in an environment for the exploration of machine musicianship, and the interactivity between humans and machines (Aim 2 - p. 142), where (iii) all activities are logged in order to observe the emergence and evolution of musical styles (Aim 3 - p. 149).

In RGeme, the three OMME's conditions have been fulfilled:

Condition 1: Agents perceive, analyse and communicate with the outside world and between them via music exchange (p. 142). Agents are also able to transmit and to receive musical influence, altering their internal states (memory - p. 145).

Condition 2: Agents execute human-inspired activities (listen to, practice and compose music - p. 143) and posses human-like perceptive and memory models (p. 145). In RGeme, the "Style Matrix" is an abstract memory representation that stores rhythmic memes and ultimately corresponds to their musical knowledge. As interactions occur, some of these memes are strengthened while others are weakened according the musical material the agents interact with.

Condition 3: A "Goal Matrix" defines a sequence of tasks for each agent and an "Evaluation Matrix" defines a set of rules (composer's name and/or year of composition) the agents use to choose from the music available in the system (p. 144). Goal and evaluation matrices forecast the musical ontogeny of the agents. RGeme tracks the evolution of musical styles by analysing the transformation of the agents' memories that follows the interactions.

With regard to design issues (Chapter 5, p. 95), RGeme is autonomous in the sense that, once a simulation is started, there is no need for human intervention. RGeme deals with monophonic rhythms, from which it encodes note onsets. There is no need to implement modules that address the "time and space awareness" or "modes of interaction" issues as interactivity is not achieved in real-time. The system was entirely implemented in C++ and runs in console mode, with no user interface.

iMe, the second implementation of a more expanded OMME-based system, is similar to RGeme with regard to the overall mechanism of interactivity. The perceptive, memory, and creative models are however much more complex. iMe also pursues the OMME's aims: (i) iMe models human-like perceptive and cognitive abilities (Aim 1 - p. 179), (ii) explores human creativity from an improvisational point of view, fosters interaction between humans and machines (Aim 2 - pp. 187 and p. 194), and (iii) logs all the agents' activities, from which it is possible to evaluate their musical evolution (Aim 3 - p. 207).

All three OMME's conditions have been observed:

Condition 1: iMe is a multi-agent interactive musical system (p. 169) where Agents perform human-inspired activities (read, practice, listen, perform, compose, solo-improvise, and collective-improvise - p. 173) and communicate between them and with the outside world. Agents are also able to transmit and to receive musical influence, altering their musical worldview (memory - p. 179).

Condition 2: iMe's agents abilities follow recent human perceptive and cognitive models (p. 175); they perceive the musical data flow via "feature filters" (parameters of the musical stream), which produce a "feature stream" of data further used for segmentation, storage (memory) and style definition purposes. The memory model implements a series of "Feature Tables" where substrings (components) of the original memes are stored.

Condition 3: iMe provides original tools for musicology, mainly focused on the ontogeny of music (developmental maps - p. 174) and on theoretical models that explore cultural evolution (p. 128). Developmental maps can be compared to the transformations of the agents' memories at the end of each time cycle (p. 207), therefore tracking stylistic developmental routes.

With regard to design issues (Chapter 5, p. 95), iMe has two major focuses: interactive performance (Section 8.3.2, p. 215) as well as musicology (Section 8.3.1.3, p. 212). Concerning the "time and space awareness" issue, iMe adopts a creative model called the Compositional and Performance Map, explained in detail in Section 8.2.4.1 (p. 187). The system was almost entirely implemented with Apple's Cocoa libraries and the Objective-C language and has a user interface, which facilitates the design of new simulations and the access to the agents' data. The user interface also provides tools for following real-time collective improvisations. Even though the system implements real-time as well as non real-time interactivity, there is currently no implemented module that addresses the "modes of interaction" issue, something recommended for future improvements (Section 9.2, p. 235).

The following Sections identify the contributions to knowledge introduced in this Thesis.

9.1 Contributions to knowledge

The contributions to knowledge introduced in this Thesis are divided into three categories - theoretical, technical and demonstrative - described below.

9.1.1 Theoretical Contributions to Knowledge

9.1.1.1 The Ontomemetical Model of Music Evolution

This Thesis introduces (Chapter 6, p. 125) a new paradigm for the design of computer interactive musical systems, the Ontomemetical Model of Music Evolution - OMME. OMME-based systems are founded on the notions of "ontogenesis" (Section 6.1) and "memetics" (Section 6.2) and must comply with three conditions: (i) be interactive systems (p. 133), (ii) regard music as an expression of human faculties (p. 136), and (iii) implement mechanisms to evaluate music evolution (p. 137). Additionally (p. 133), OMME-based systems must pursue to (i) contribute to the understanding of natural phenomena such as human perception and cognition by computational modelling, (ii) contribute to the achievement of machine musicianship, and the interactivity between humans and machines, and (iii) provide computational tools for musicology, mainly focused on theoretical models that explore cultural evolution.

9.1.1.2 A plausible definition of interactive musical systems

The concept of interactivity has become overused in recent years, which led to the misunderstanding and misuse of this idea. This Thesis (Chapter 4 and

Chapter 5) explores a number of interactive systems in order to find the core elements of interactivity and proposes the following definition for this term: "Interactive musical systems are computer-based systems that, through musical information exchange, have the ability to perceive the environment, analyse and take actions as to alter the surrounding states as well as to change their own internal states" (p. 93).

9.1.1.3 Survey of computer models that explore musical creativity

This Thesis (Chapter 3) introduces a critical and systematic overview of the state of the art computer models that explore musical perception and creativity using Artificial Intelligence approaches. Artificial Intelligence systems involve numerous fields (e.g., Psychology, Music) and methods (e.g., rule-based, grammar-based, and machine-learning systems) in order to reproduce human intelligence. Evolutionary computation and A-life models try to replicate biological phenomena via computer simulations in order to shed light into areas such as the origins of living organisms, emergent behaviour and self-organisation.

9.1.1.4 Survey of methodologies for design and implementation of interactive musical systems

This Thesis (Chapter 5) presents a systematic exposition of some of the major issues with regard to the design and implementation of interactive musical systems and explains the techniques many researchers have used to tackle them. Seven subjects have been specifically addressed: (i) general goals (p. 96), (ii) spatial localization and synchronization (p. 98), (iii) modes of interaction (p. 101), (iv) musical material (p. 108), (v) interface design (p. 109), (vi) music representation (p. 113), and (vii) computer languages and environments (p. 117).

9.1.2 Technical Contributions to Knowledge

9.1.2.1 New techniques for the representation and segmentation of musical sequences

This Thesis introduces (Section 8.2.1, p. 170) a new solution for the representation of music based on "feature streams" that is used in iMe and takes into account how sounds are captured by the ears are stored in the brain. Agents perceive changes in the "sound flow" between two consecutive moments in time and extract "sensory information" from them according to "feature filters". iMe's current version uses the symbolic representation of the MIDI protocol and implements 5 filters that deal with notes from the melody (melodic direction, melodic leap, melodic inter-onset interval, melodic duration, melodic intensity) and another 5 filters that deal with non-melodic notes (non-melodic number of notes, non-melodic note intervals from melody, non-melodic inter-onset interval, non-melodic duration, non-melodic intensity).

In addition, this Thesis (Section 8.2.3, p. 175) introduces a new algorithm for the segmentation of the musical flow based on the behaviour of the above-mentioned "feature stream", a continuous stream of musical data produced by the agent's "feature filters" that emulate the human "feature extraction" (Section 2.1, p. 32). The algorithm is based on the principles of Gestalt Psychology and is explained in detail in page 37.

9.1.2.2 An ontomemetical memory model

This Thesis (Section 8.2.3.1, p. 179) introduces a new memory model (long-term memory) implemented by iMe that uses a set of "Feature Tables" (FTs). Each FT is responsible for storing a specific musical feature and each row in a FT, called a "Feature Line", keeps a record of (i) the feature substring, (ii) dates of interaction, (iii) number of contacts, (iv) relative weight of the FS, and (v) connection pointers.

9.1.2.3 A new algorithm for measuring distances between symbolic memories

This Thesis (Section 8.3.1.1, p. 209) proposes a measure of distance between the memories of two agents, by using the "block distance" between corresponding pairs of Feature Tables.

In addition, in cooperation with Martins, Manzolli and Maia Jr. (Martins *et al.*, 2005), an algorithm is proposed to measure the similarity between sub-sequences in a general rhythmic space using a structure called Similarity Coefficient Vector. In this model, all sub-sequences of a given rhythm are compared: a hierarchical subdivision of rhythm sequences is done in several levels, and a Distance Matrix for each level is computed using the "block distance". The information about the similarity of the rhythmic substructures is retrieved from the matrices and coded into the Similarity Coefficient Vector (SCV) whose entries estimate the similarity between rhythm sequences in different k-levels.

9.1.2.4 Improvisation model

This Thesis (Section 8.2.4.1, p. 187) introduces a new generative model called the "Compositional and Performance Map" (CPM), based on local constraints that address particular aspects of improvisation: the generation of new musical ideas (memes) and the performance of these ideas. Generative parameters include the mode of meme generation, meme transformations (e.g., inversions, mirroring, transpositions), local chords and scales, etc. Performance parameters include the ratio between the intensity of the left hand (non-melodic notes) and the right hand (melodic notes), shifts for note onsets, and the tempo (speed of execution).

9.1.3 Demonstrative Contributions to Knowledge

This Thesis also introduces demonstrative contributions to knowledge, the interactive musical systems Rhythmic Meme Generator - RGeme (Chapter 7, p. 141), and the Interactive Musical Environments - iMe (Chapter 8, p. 169), which are used to illustrate how OMME-based systems can be implemented.

Both systems are multi-agent interactive musical systems; agents implement perceptive and cognitive modules and are able to evolve their musical styles from the performance of musical tasks. In RGeme agents perform non real-time tasks (listen to, practice and compose music) while in iMe, in addition to non real-time tasks (read and compose music), agents can also perform real-time tasks (listen, perform, solo-improvise and collective-improvise).

Finally, the following Section presents some recommendations for future work.

9.2 Recommendations for Future Work

9.2.1 Musical Data Bases

The case studies introduced in this Thesis (pp. 149 and 212) demonstrate that OMME-based systems have an extraordinary potential in many fields of systematic musicology. It was possible during my PhD to design and implement a series of experiments using a number of different musical materials. In the years to come, the understanding of these experiments could be refined with the analysis of broader collections of pieces that would include a higher diversity of genres, instrumentation, and musical forms, among other factors.

9.2.2 Interactive Musical Environments - iMe

The iMe system could be improved with the extension of the current or implementation of new modules. The following paragraphs point to some possible directions.

a) Moving Towards Self-Sufficiency

The current implementation of iMe demands the manual control of a number of parameters (e.g., number of agents, number and type of tasks) with the concern of approaching specific musicological and/or creativity issues among which are previously defined ontogenies. In fact, this strategy drives the evolution of the musicality of the agents and of the system as a whole. Another interesting possibility would be to start the system with fewer restrictions (e.g., global time span or number of cycles) in order to observe this evolution in a more self-sufficient and less constrained environment.

In order to achieve this goal, agents could be allowed to reproduce and generate offspring where some of the characteristics of the parents (e.g., character traits) would be transmitted and evolve. In addition, agents could be allowed to take their own decisions with respect to the tasks they perform (e.g., number of tasks, moment of execution). Agents could also possess some psychological parameters that would represent social pressure, rewarding or punishing the choices they make during their careers (e.g., production of some musical aspects).

b) Expansion of the Musical Tasks

iMe already implements seven different music-related tasks (reading, practicing, listening, performing, composing, solo-improvising, and collective-improvising). However, there are other processes that involve the human cognitive system (e.g., recollecting, reminding) that have not been addressed. In a more self-sufficient system it would interesting to implement tasks replicating these processes as they can influence the development of the musicality of the agents.

c) Perception of Higher-Level Musical Structures

iMe's current model of perception is mainly focused on local continuities, from one meme to another, and on the relationship between different musical parameters. In fact, the agents were not taught to perceive and/or to learn higher-level structures such as musical phrases and sections or the fact that memes originally belong to certain sections of the original music. Future versions of the system could address these issues as they can be used as parameters in the development of the agents' musicality and in generative tasks.

d) Enhancement of the Creative Model

iMe's current model of creativity mainly focus on improvisation and is developed around Compositional and Performance Maps (Section 8.2.4.1, p. 187), which are fixed for an entire simulation. It would be convenient, however, that agents had the ability to define in real-time some of the constraints of the CPMs such as the meme generation mode, scale type, and note ranges, among others. This change would improve the quality and diversity of the agents' improvisation.

In addition to that, the compositional tasks could be modified as to allow the agents to rewrite sections of the compositions, revisiting their early decisions. In this case, iMe would be able to address at the same time - within different tasks - not only improvisation but also, in a more strict sense, (algorithmic) composition.

e) Real-time interactivity

In real-time collective-improvisations, agents are not currently able to react to changes in the ongoing musical flow with regard to texture, density, etc., and, therefore, in these tasks, musical dialogues are more apparent than "real". A better sense of real-time interactivity would be achieved with the improvement of the agent's perceptive abilities via the implementation of different modes of interaction (Section 5.3, p. 101).

In addition to that, the collective-improvisation tasks could be expanded as to allow agents to improvise among themselves in real-time, something that was not addressed in the current implementation of iMe.

Appendices

1 Compact Disc

The CD that accompanies this Thesis contains the following folders and files:

1 `RGeme`
- 1 - `RGeme Case Study` (data corresponding to Section *7.3*, p. *149*)
  - `Musical Pieces` (MIDI files of musical pieces by Chiquinha Gonzaga, Ernesto Nazareth and Jacob do Bandolim used in simulations)
  - `RGeme Case Study 1.xls` (agent's log file)
- 2 - `RGeme Additional Case Studies` (data corresponding to Section *7.4*, p. *159*)
  - `Music` (musical pieces by J.S. Bach and Tom Jobim used in simulations)
  - `Simulations` (log files and goal matrices of each simulation)

2 `iMe` (data corresponding to Section *8.3.2*, p. *215*)
- `agent.mid` (agent's recorded performance)
- `cpm.xlsx` (Compositional and Performance Map)
- `marcelo.mid` (Marcelo's recorded performance)
- `memories.xlsx` (agent's memories)
- `ICCMR_Concert.m4v` (concert's movie)

2 Record of Activities

Papers and Posters

Gimenes, M., Miranda, E. and Johnson, C. (2005). 'A Memetic Approach to the Evolution of Rhythms in a Society of Software Agents'. *Proceedings of the Brazilian Symposium on Computer Music*. Belo Horizonte, Brazil.

Martins, J., Gimenes, M., Manzolli, J. and Maia Jr, A. (2005). 'Similarity Measures for Rhythmic Sequences'. *Proceedings of the Brazilian Symposium on Computer Music*. Belo Horizonte, Brazil.

Coutinho, E., Gimenes, M., Martins, J. and Miranda, E. (2005). 'Computational Musicology: An Artificial Life Approach"'. *Proceedings of the Portuguese Workshop on Artificial Life and Evolutionary Algorithms*. Covilhã, Portugal.

Gimenes, M., Miranda, E. and Johnson, C. (2005). 'Towards an intelligent rhythmic generator based on given examples: a memetic approach'. *Proceedings of the Digital Music Research Network Summer Conference*. Glasgow, UK.

Martins, J. M., Gimenes, M. and Zhang, Q. (2005) Book Review: 'Music Query. Methods, Models and User Studies' Edited by Walter B. Hewlett and Eleanor Selfridge-Field. *Computing in Musicology*. The MIT Press. Cambridge, 2004.

Gimenes, M. (2005). 'Musical Ontogenesis and the computer modelling of musical influence' (Poster). *Digital Music Research: One-day Workshop and Roadmap Launch*. Queen Mary, University of London.

Gimenes, M., Miranda, E. and Johnson, C. (2005). 'On the Learning Stages of an Intelligent Rhythmic Generator'. Sound and Music Computing Conference. Salerno, Italy.

Gimenes, M., Miranda, E. and Johnson, C. (2006). 'The development of musical styles in a society of software agents'. *Proceedings of the International Conference on Music Perception and Cognition*. Bologna, Italy.

Gimenes, M. (2006). 'The development of musical styles in a society of software agents' (Poster). *International Conference on Music Perception and Cognition*. Bologna, Italy.

Gimenes, M., Miranda, E. and Johnson, C. (2007). 'The Emergent Musical Environments: An Artificial Life Approach'. *Proceedings of the Workshop on Music and Artificial Life (ECAL)*. Lisbon, Portugal.

Gimenes, M., Miranda, E. and Johnson, C. (2007). 'Musicianship for Robots with Style'. *Proceedings of the New Interfaces for Musical Expression*. New York, USA.

Gimenes, M. and Miranda, E. (2008). 'An A-Life Approach to Machine Learning of Musical Worldviews for Improvisation Systems'. *Proceedings of the 5th Sound and Music Computing Conference*. Berlin, Germany.

Gimenes, M. and Miranda, E. (2008). 'Emergent Worldviews: An Ontomemetical Approach to Musical Intelligence', in: Miranda, E. (ed). (under preparation).

Public Performances

Gimenes, M. (2006). Peninsula Arts Contemporary Music Festival 2006. Peninsula Arts and Interdisciplinary Centre for Computer Music Research, Plymouth, UK.

Gimenes, M. and Miranda, E. (2006). BCI got Rhythm (Two Pianos and Brain Cap Improvisation). Sonic Fusion Festival, Edinburgh, UK.

Gimenes, M. (2008). Peninsula Arts Contemporary Music Festival 2008. Peninsula Arts and Interdisciplinary Centre for Computer Music Research, Plymouth, UK.

Seminars and Conference Presentations

Gimenes, M. (2005). 'Towards an intelligent rhythmic generator based on given examples: a memetic approach'. Digital Music Research Network Summer Conference. Glasgow, UK.

Gimenes, M. (2005). 'Music Learning Environments'. Interdisciplinary Centre for Computer Music Research Seminar, Plymouth, UK.

Gimenes, M. (2005). 'Simulating Musical Ontogenesis in an Artificial Society'. Interdisciplinary Centre for Computer Music Research Seminar, Plymouth, UK.

Gimenes, M. (2006). 'BCI got Rhythm: Improvisation for Two Pianos and Brain-Computer Music Interface'. Workshop "Improvisation and Computers". International Conference on New Interfaces for Musical Expression, Paris, France.

Gimenes, M. (2006). 'Cultural Influences & Cognitive Processes in Music'. Interdisciplinary Centre for Computer Music Research Seminar, Plymouth, UK.

Gimenes, M. (2006). 'The development of musical styles in a society of software agents'. International Summer School in Systematic Musicology (ISSSM 2006). Ghent, Belgium.

Gimenes, M. (2007). 'Musicianship for Robots with Style'. New Interfaces for Musical Expression. New York, USA.

Gimenes, M. (2007). 'The Emergent Musical Environments: An Artificial Life Approach'. Workshop on Music and Artificial Life (ECAL). Lisbon, Portugal.

Gimenes, M. (2007). 'Emergência e desenvolvimento de estilos musicais em mundos artificiais'. Núcleo Interdisciplinar de Comunicação Sonora. Campinas, Brazil.

Gimenes, M. (2008). 'An A-Life Approach to Machine Learning of Musical Worldviews for Improvisation Systems'. 5th Sound and Music Computing Conference. Berlin, Germany.

Gimenes, M. (2008). 'An Approach to Machine Musicianship'. Informatics Research Institute Seminars. University of Exeter. Exeter, UK.

Gimenes, M. (2008). 'Emergent worldviews: An ontomemetical approach to musical intelligence'. International Music Computing Research Workshop. The Open University. Milton Keynes, UK.

Gimenes, M. (2008). 'Interactive Composition with Learned Ontogenesis'. Workshop on Artificial Intelligence Systems for Composition (Interdisciplinary Centre for Computer Music Research). Plymouth, UK.

Other Conferences Attended

2005. European Summer School in Information Retrieval (ESSIR 2005). Dublin, Ireland.

2005. 6th International Conference on Music Information Retrieval. London, UK.

2006. International Summer School in Systematic Musicology (ISSSM 2006). Ghent University. Ghent, Belgium

2007. Apple Worldwide Developers Conference 2007. San Francisco, USA.

Press Exposure

The Evening Herald (2008). 'The one man jam is here'. Available at: www.thisisplymouth.co.uk/news/man-jam/article-194474-detail/article.html. (Accessed: 24/06/2008).

EET (2008). 'Virtual musician learns any playing style'. Available at: http://www.eetimes.com/showArticle.jhtml?articleID=210300239. (Accessed: 1010/2008).

Bibliography

Addessi, A. R., Ferrari, L., Carlotti, S. and Pachet, F. (2006). 'Young children musical experience with a flow machine'. *Proceedings of the International Conference on Music Perception and Cognition*. Bologna, Italy.

Addessi, A. R. and Pachet, F. (2005). 'Experiments with a Musical Machine: Musical Style Replication in 3/5 year old Children', *British Journal of Music Education*, 22 (1), pp. 21-46.

Amatriain, X. (2004). *An Object-Oriented Metamodel for Digital Signal Processing with a focus on Audio and Music*. PhD, Universitat Pompeu Fabra.

Andrews, R. (1987). *The Routledge Dictionary of Quotations*. Routledge.

Anon (2005a). Oxford Dictionary of English.

Anon (2005b). The New Oxford American Dictionary. (2nd ed).

Arshavsky, Y. I. (2003). 'When did Mozart Become a Mozart? Neurophysiological Insight Into Behavioral Genetics', *Brain and Mind*, 4, pp. 327-339.

Assayag, G. (2008). 'The OMax Project Page'. Available at: http://recherche.ircam.fr/equipes/repmus/OMax/. (Accessed: 08/10/2008).

Assayag, G., Bloch, G., Chemillier, M., Cont, A. and Dubnov, S. (2006). 'Omax Brothers: a Dynamic Topology of Agents for Improvisation Learning'. *Proceedings of the Workshop on Audio and Music Computing for Multimedia*. Santa Barbara, USA.

Assayag, G. and Dubnov, S. (2004). 'Using Factor Oracles for machine Improvisation', *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 8 (9), pp. 604-610.

Assayag, G., Dubnov, S. and Delerue, O. (1999). 'Guessing the Composer's Mind: Applying Universal Prediction to Musical Style'. *Proceedings of the International Computer Music Conference*. Beijing, China.

Austin, J. (2006). *How to Do Things with Words (William James Lectures)*. (2nd ed) Harvard University Press, Cambridge, Massachusetts.

Baker, M. (1989a). 'A cognitive model for the perception of musical grouping structures', *Contemporary Music Review* (Music and the Cognitive Sciences).

Baker, M. (1989b). 'An Artificial Intelligence approach to musical grouping analysis', *Contemporary Music Review*, 3 (1), pp. 43-68.

Biles, J. A. (1994). 'GenJam: A Genetic Algorithm for Generating Jazz Solos'. *Proceedings of the International Computer Music Conference*. Aarhus, Denmark.

Biles, J. A. (1998). 'Interactive GenJam: Integrating Real-Time Performance with a Genetic Algorithm'. *Proceedings of the International Computer Music Conference*. Univ. of Michigan, Ann Arbor, USA.

Biles, J. A. (1999). 'Life with GenJam: Interacting with a Musical IGA'. *Proceedings of the International Conference on Systems, Man, and Cybernetics*. Tokyo, Japan.

Blackmore, S. (1999). *The Meme Machine*. Oxford University Press, Oxford.

Blackwell, T. (2006). 'Swarming and Music', in: Miranda, E. and Biles, J.A. (eds). *Evolutionary Computer Music*. London: Springer.

Bloch, J. J. and Dannenberg, R. B. (1985). 'Real-Time Computer Accompaniment of Keyboard Performances'. *Proceedings of the International Computer Music Conference*. Vancouver, Canada.

Blomberg, S., Rosander, M. and Andersson, G. (2006). 'Fears, Hyperacusis and Musicality in Williams Syndrome', *Research in Developmental Disabilities*, 27, pp. 668-680.

Bod, R. (2001). 'A Memory-Based Model For Music Analysis: Challenging The Gestalt Principles'. *Proceedings of the International Computer Music Conference*. Havana, Cuba.

Boden, M. A. (1994a). 'Agents and Creativity', *Communications of the ACM*, 37 (7), pp. 117-121.

Boden, M. A. (1994b). 'Precis of The creative mind: Myths and mechanisms', *Behavioral and Brain Sciences*, 17 (3), pp. 519-570.

Boden, M. A. (2003). *The Creative Mind: Myths and Mechanisms*. (2nd ed) Routledge.

Borchers, J. (2000). 'A pattern approach to interaction design'. *Proceedings of the Symposium on Designing Interactive Systems - Conference on Designing interactive systems: processes, practices, methods, and techniques*. New York, USA.

Borchers, J. and Muhlhauser, M. (1998). 'Design Patterns for Interactive Musical Systems', *IEEE Multimedia*, 5 (3), pp. 36-46.

Boroda, M. B. (1992). 'Towards a Phrase Type Melodic Unit in Music', *Musikometrika*, 50 (4), pp. 15-82.

Boulanger, R. (2000). *The Csound Book: Perspectives in Software Synthesis, Sound Design, Signal Processing and Programming*. MIT Press.

Brooks Jr., F., Hopkins Jr., A., Neumann, P. and Wright, W. (1993). 'An experiment in musical composition', in: Schwanauer and Levitt (eds). *Machine Models of Music*. MIT Press, pp 23-40.

Brown, C. (1999). 'Talking Drum: A Local Area Network Music Installation', *Leonardo Music Journal*, 9, pp. 23-28.

Cambouropoulos, E. (1997). 'Musical rhythm: A formal model for determining local boundaries, accents and metre in a melodic surface', in: Leman, M. (ed). *Music, Gestalt, and Computing: Studies in Cognitive and Systematic Musicology*. Berlin: Springer Verlag, pp 277-293.

Cambouropoulos, E. (1998). *Towards a General Computational Theory of Musical Structure*. PhD, University of Edinburgh.

Camilleri, L., Carreras, F. and Duranti, C. (1990). 'An Expert System Prototype for the Study of Musical Segmentation', *Interface*, 19 (2-3), pp. 147-154.

Camurri, A., Trocca, R. and Volpe, G. (2002). 'Interactive Systems Design: A KANSEI based Approach'. *Proceedings of the Conference on New Interfaces for Musical Expression*. Dublin, Ireland.

Carpinteiro, O. A. S. (1995). 'A neural model to segment musical pieces'. *Proceedings of the Brazilian Symposium on Computer Music*.

Casal, D. P. and Morelli, D. (2007). 'Remembering the future: towards an application of genetic co-evolution in music improvisation'. *Proceedings of the MusicAL Workshop, European Conference on Artificial Life*. Lisbon, Portugal.

Cavalli-Sforza, L. L. and Feldman, M. W. (1981). 'Cultural Transmission and Evolution: A Quantitative Approach'*Monographs in Population Biology*. Princeton: Princeton University Press.

Cemgil, A. T. and Kappen, B. (2001). 'Bayesian Real-Time Adaptation For Interactive Performance Systems'. *Proceedings of the International Computer Music Conference*. Havana, Cuba.

Chouvel, J. M. (1990). 'Musical Form: From a Model of Hearing to an Analytic Procedure', *Interface*, 22, pp. 99-117.

Collins, N. M. (2006). *Towards Autonomous Agents for Live Computer Music: Realtime Machine Listening and Interactive Music Systems*. PhD, University of Cambridge.

Conklin, D. and Witten, I. H. (1995). 'Multiple Viewpoint Systems for Music Prediction', *Journal of New Music Research*, 24 (1), pp. 51-73.

Cook, P. R. and Scavone, G. P. (2008). 'The Synthesis ToolKit in C++ (STK)'. Available at: http://ccrma.stanford.edu/software/stk/. (Accessed: 09/10/2008).

Cope, D. (1991). 'Recombinant Music: Using the Computer to Explore Musical Style', *Computer*, 24 (7), pp. 22-28.

Cope, D. (1999). 'One approach to musical intelligence', *IEEE Intelligent Systems*, 14 (3), pp. 21-25.

Cope, D. (2000). Classical Music Composed by Computer: Experiments in Musical Intelligence (Audio CD). Centaur.

Cope, D. (2004). 'A Musical Learning Algorithm', *Computer Music Journal*, 28 (3), pp. 12-27.

Coutinho, E., Gimenes, M., Martins, J. and Miranda, E. (2005). 'Computational Musicology: An Artificial Life Approach'. *Proceedings of the Portuguese Workshop on Artificial Life and Evolutionary Algorithms.* Covilhã, Portugal.

Cox, A. (2001). 'The mimetic hypothesis and embodied musical meaning', *MusicæScientiæ*, 2, pp. 195-212.

Csikszentmihalyi, M. (1991). *Flow: The Psychology of Optimal Experience.* SOS Free Stock.

Cycling'74 (2008). 'Cycling '74 Website'. Available at: http://www.cycling74.com/. (Accessed: 02/09/2008).

Dannenberg, R. B. (1986). 'Arctic: A Functional Language for Real-Time Control ', *Computer Music Journal*, 10 (4), pp. 67-78.

Dannenberg, R. B. (1989). 'Real Time Control For Interactive Computer Music and Animation'. *Proceedings of the The Arts and Technology II: A Symposium.* Connecticut College.

Dannenberg, R. B. (1996). 'The CMU MIDI Toolkit'. *Proceedings of the International Computer Music Conference.* Hong Kong University of Science and Technology, China.

Dannenberg, R. B. (1997). 'The Implementation of Nyquist, A Sound Synthesis Language', *Computer Music Journal*, 21 (3), pp. 71-82.

Dannenberg, R. B. (2002). 'A Language for Interactive Audio Applications'. *Proceedings of the International Computer Music Conference.* Gothenburg, Sweden.

Dannenberg, R. B. and Brandt, E. (1996). 'A Flexible Real-Time Software Synthesis System'. *Proceedings of the International Computer Music Conference*. Hong Kong University of Science and Technology, China.

Dannenberg, R. B. and Mont-Reynaud, B. (1987). 'Following a jazz improvisation in real time.'. *Proceedings of the International Computer Music Conference*. Urbana, USA.

Darwin, C. (1998). *The Origin of Species.* (new ed) Wordsworth Editions Ltd.

Dawkins, R. (1989). *The Selfish Gene.* Oxford University Press, Oxford.

Deutsch, D. (1982a). 'Grouping Mechanisms in Music', in: Deutsch, D. (ed). *The Psychology of Music.* New York: Academic Press.

Deutsch, D. (1982b). 'The Processing of Pitch Combinations', in: Deutsch, D. (ed). *The Psychology of Music.* New York: Academic Press.

Deutsch, D. and Feroe, J. (1981). 'The Internal Representation of Pitch Sequences in Tonal Music', *Psychological Review*, 88, pp. 503-522.

Dhande, S. S. (2003). *A Computational Model to Connect Gestalt Perception and Natural Language* Master, Massachusetts Institute of Technology.

Dobrian, C. (2004). 'Strategies for Continuous Pitch and Amplitude Tracking in Realtime Interactive Improvisation Software'. *Proceedings of the Sound and Music Computing*. Paris, France.

Dubnov, S. and Assayag, G. (2005). 'Improvisation Planning And Jam Session Design Using Concepts Of Sequence Variation And Flow Experience'. *Proceedings of the Sound and Music Computing*. Salerno, Italy.

Dubnov, S., Assayag, G., Lartillot, O. and Bejerano, G. (2003). 'Using Machine-Learning Methods for Musical Style Modeling', *IEEE Computer*, 10 (38), pp. 73-80.

Durand, N. (2003). *Apprentissage du style musical et interaction sur deux échelles temporelles.* DEA, Université Pierre et Marie Curie, Paris VI.

Ebcioglu, K. (1988). 'An expert system for harmonizing four-part chorales', *Computer Music Journal*, 12 (3), pp. 43-51.

Evans, B. (2005). The Universal Mind Of Bill Evans (DVD). Records, D.

Feng, J. (2004). *Computational Neuroscience: A Comprehensive Approach*. CRC Press.

Flanagan, O. J. (1991). *The Science of the Mind.* (2nd ed) MIT Press.

Forte, A. (1983). *Introduction to Schenkerian Analysis: Form and Content in Tonal Music.* W. W. Norton & Company.

Francois, A. R. J., Chew, E. and Thurmond, D. (2007a). *MIMI, A Musical Improvisation System That Provides Visual Feedback to the Performer*, University of Southern California, Los Angeles.

Francois, A. R. J., Chew, E. and Thurmond, D. (2007b). 'Visual Feedback in Performer-Machine Interaction for Musical Improvisation'. *Proceedings of the Conference on New Interfaces for Musical Expression*. New York, USA.

Franklin, J. A. (2001). 'Multi-Phase Learning for Jazz Improvisation and Interaction'. *Proceedings of the Eighth Biennial Symposium for Arts & Technology*. Connecticut, USA.

Gabora, L. (1997). 'The Origin and Evolution of Culture and Creativity', *Journal of Memetics*.

Garnett, G. E. (2001). 'The Aesthetics of Interactive Computer Music', *Computer Music Journal*, 25 (1), pp. 21-33.

Gimenes, M. (2003a). 'As Estruturas Verticais na Improvisacao de Bill Evans'. *Proceedings of the Brazilian Symposium on Computer Music*. Campinas, Brazil.

Gimenes, M. (2003b). *Aspectos estilisticos do jazz atraves da identificacao de estruturas verticais encontradas na obra de Bill Evans*. Master, Universidade Estadual de Campinas.

Gimenes, M. and Miranda, E. (2008). 'An A-Life Approach to Machine Learning of Musical Worldviews for Improvisation Systems'. *Proceedings of the 5th Sound and Music Computing Conference*. Berlin, Germany.

Gimenes, M., Miranda, E. and Johnson, C. (2005a). 'A Memetic Approach to the Evolution of Rhythms in a Society of Software Agents'. *Proceedings of the Brazilian Symposium on Computer Music*. Belo Horizonte, Brazil.

Gimenes, M., Miranda, E. and Johnson, C. (2005b). On the Learning Stages of an Intelligent Rhythmic Generator. Sound and Music Computing Conference. Salerno, Italy.

Gimenes, M., Miranda, E. and Johnson, C. (2005c). 'Towards an intelligent rhythmic generator based on given examples: a memetic approach'. *Proceedings of the Digital Music Research Network Summer Conference*. Glasgow, UK.

Gimenes, M., Miranda, E. and Johnson, C. (2007a). 'Musicianship for Robots with Style'. *Proceedings of the New Interfaces for Musical Expression*. New York, USA.

Gimenes, M., Miranda, E. and Johnson, C. (2007b). 'The Emergent Musical Environments: An Artificial Life Approach'. *Proceedings of the Workshop on Music and Artificial Life (ECAL)*. Lisbon, Portugal.

Gonzalez, M. E. Q., Del-Masso, M. C. S. and Piqueira, J. R. C. (2001). *Encontro com as Ciências Cognitivas - III Encontro Internacional de Ciências Cognitivas*. Marilia Publicações e Cultura Acadêmica/Editora Unesp, Marilia, Brazil.

Good, M. (2006). Lessons from the Adoption of MusicXML as an Interchange Standard. XML 2006 Conference. Boston, USA.

Gould, S. J. (1977). *Ontogeny and Phylogeny*. Harvard University Press.

Griffeath, D. and C., M. (2003). *New Directions in Cellular Automata*. Oxford University Press.

Handel, S. (1989). *Listening: An Introduction to the Perception of Auditory Events*. The MIT Press.

Hasty, C. F. (1978). *A theory of segmentation developed from late works of Stefan Wolpe*. PhD, Yale University.

Hebb, D. O. (1949). *The Organization of Behavior: A Neuropsychological Theory*. (new ed) Wiley, New York.

Herald (2008). 'The one man jam is here'. Available at: www.thisisplymouth.co.uk/news/man-jam/article-194474-detail/article.html. (Accessed: 24/06/2008).

Hiller, L. and Isaacson, L. (1959). *Experimental Music*. McGraw-Hill, New York.

Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems - An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. The MIT Press.

Horowitz, D. (1994). 'Generating Rhythms with Genetic Algorithms'. *Proceedings of the International Computer Music Conference*. Aarhus, Denmark.

Hsu, W. (2006). 'Managing Gesture and Timbre for Analysis and Instrument Control in an Interactive Environment'. *Proceedings of the Conference on New Interfaces for Musical Expression*. Paris.

Hsu, W. (2007). 'Design Issues in Interaction Modeling for Free Improvisation'. *Proceedings of the Conference on New Interfaces for Musical Expression*. New York.

Huber, D. M. (2000). *The MIDI Manual: A Practical Guide to MIDI in the Project Studio*. Focal Press.

Huron, D. (1994). 'Humdrum - Frequently Asked Questions'. Available at: http://www.lib.virginia.edu/artsandmedia/dmmc/Music/Humdrum/humdrumFAQ.html. (Accessed: 08/10/2008).

Huron, D. (2002a). 'Listening Styles and Listening Strategies (Handout)'. *Proceedings of the Society for Music Theory Conference*. Columbus, Ohio, USA.

Huron, D. (2002b). 'Music Information Processing Using the Humdrum Toolkit: Concepts, Examples, and Lessons', *Computer Music Journal*, 26 (2), pp. 11-26.

Huron, D. (2008). 'Music 829D: Music and Emotion'. Available at: http://csml.som.ohio-state.edu/Music829D/music829D.html. (Accessed: 08/10/08).

ICCMR (2008). 'Peninsula Arts Contemporary Music Festival 2008'. Available at: http://cmr.soc.plymouth.ac.uk/event2008.htm. (Accessed: 10/10/2008).

Impett, J. (2001). 'Interaction, simulation and invention: a model for interactive music'. *Proceedings of the Workshop on Artificial Models for Musical Applications*. Cosenza, Italy.

Isaacson, E. (1996). 'Neural network models for the study of post-tonal music'. *Proceedings of the Joint International Conference on Cognitive and Systematic Musicology*. Brugge, Belgium.

Jacob, B. L. (1995). 'Composing with genetic algorithms'. *Proceedings of the International Computer Music Conference*. Banff Centre for the Arts, Canada.

Jan, S. (2000). 'Replicating sonorities: towards a memetics of music', *Journal of Memetics*, 4.

Jan, S. (2004). 'Meme Hunting with the Humdrum Toolkit: Principles, Problems and Prospects', *Computer Music Journal*, 28 (4), pp. 68-84.

Johnson, R. C. (2008). 'Virtual musician learns any playing style'. Available at: http://www.eetimes.com/showArticle.jhtml?articleID=210300239. (Accessed: 1010/2008).

Johnson-Laird, P. N. (1991). 'Jazz Improvisation: A Theory at the Computational Level', in: Howell, P., West, R. and Cross, I. (eds). *Representing Musical Structure*. Academic Press, pp 291-325.

Johnson-Laird, P. N. (2002). 'How Jazz Musicians Improvise', *Music Perception*, 19 (3), pp. 415–442.

Jones, M. T. (2008). *Artificial Intelligence - A System Approach*. Infinity Science Press, Hingham, Massachusetts.

Jorda, S. (1991). 'A Real-Time Midi Composer And Interactive Improviser By Means Of Feedback Systems'. *Proceedings of the International Computer Music Conference.* McGill University, Canada

Jorda, S. (2001). 'Improvising with Computers: A Personal Survey (1989-2001)'. *Proceedings of the International Computer Music Conference.* Havana, Cuba.

Jorda, S. (2003). 'Interactive Music Systems For Everyone: Exploring Visual Feedback As A Way For Creating More Intuitive, Efficient And Learnable Instruments'. *Proceedings of the Stockholm Music Acoustics Conference.* Stockholm, Sweden.

Kirke, A. and Miranda, E. (2007). 'Evaluating Mappings for Cellular Automata Music'. *Proceedings of the Workshop on Music and Artificial Life (ECAL).* Lisbon, Portugal.

Koelsch, S., Fritz, T., Schulze, K., Alsop, D. and Schlaug, G. (2005). 'Adults and children processing music: An fMRI study', *NeuroImage*, 25, pp. 1068-1076.

Lartillot, O., Dubnov, S., Assayag, G. and Bejerano, G. (2001). 'Automatic Modeling of Musical Style'. *Proceedings of the International Computer Music Conference.* Havana, Cuba.

Lerdahl, F. and Jackendoff, R. (1983). *A Generative Theory of Tonal Music.* MIT Press, Cambridge, Mass.

Lewis, G. E. (2000). 'Too Many Notes: Computers, Complexity and Culture in Voyager', *Leonardo Music Journal*, 10, pp. 33-39.

Lima, E. T. d., Guigue, D. and Ferneda, E. (2000). 'Automatic Segmentation of Musical Flows: A Rational Agents Approach'. *Proceedings of the Brazilian Symposium on Computer Music.* Curitiba, Brazil.

Luck, G. (2008). 'Modelling the relationships between emotional responses to, and musical content of, music therapy improvisations', *Psychology of Music*, 36 (1), pp. 25-45.

Maes, P. (1991). *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back.* MIT Press.

Manzolli, J., Costa, M. O., Ramos, F. L., Fornari, J. E. and Sharoni, D. (2002). 'Solutions for distributed musical instruments on the web'. *Proceedings of the Conference on the Principles and Practice of programming*. Dublin, Ireland.

Manzolli, J. and Jr., A. M. (1995). 'Interactive Composition Using Markov Chain And Boundary Functions'. *Proceedings of the Brazilian Symposium on Computer Music*. Betim, Brazil.

Manzolli, J., Wassermann, K., Blanchard, M. and Verschure, P. (2000). 'A Robot Based Interactive Composition System'. *Proceedings of the Brazilian Symposium on Computer Music*. Curitiba, Brazil.

Marsden, A. (2007). 'Automatic Derivation Of Musical Structure: A Tool For Research On Schenkerian Analysis'. *Proceedings of the International Conference on Music Information Retrieval*. Viena, Austria.

Martins, J., Gimenes, M., Manzolli, J. and Maia Jr, A. (2005). 'Similarity Measures for Rhythmic Sequences'. *Proceedings of the Brazilian Symposium on Computer Music*. Belo Horizonte, Brazil.

McAdams, S. (1984). 'The auditory Image: A metaphor for musical and psychological research on auditory organisation',. in: Crozier, W.R. and Chapman, A.J. (eds). *Cognitive Processes in the Perception of Art*. Amsterdam: North-Holland Press.

McCartney, J. (1996). 'SuperCollider: a new real time synthesis language'. *Proceedings of the International Computer Music Conference*. Hong Kong University of Science and Technology, China.

McCartney, J. (2002). 'Rethinking the Computer Music Language: SuperCollider', *Computer Music Journal*, 26 (4), pp. 61-68.

McIntyre, R. A. (1994). 'Bach in a box: the evolution of four part Baroque harmony using thegenetic algorithm'. *Proceedings of the IEEE World Congress on Computational Intelligence*. Orlando, USA.

Mendel, G. (1865). Experiments on Plant Hybridization. Meetings of the Natural History Society of Brünn. Moravia, Czech Republic. Available online at: http://www.mendelweb.org/Mendel.html.

Meyer, L. (1961). *Emotion and Meaning in Music*. Chicago University Press.

Meyer, L. (1989). *Style and Music: Theory, History, and Ideology.* University of Pennsylvania Press, Philadelphia.

Miller, G. (1956). 'The magical number seven, plus or minus two: Some limits on our capacity for processing information', *Psychological Review*, 63, pp. 81-97.

Minelli, A. (2003). *Development of Animal Form: Ontogeny, Morphology, and Evolution.* Cambridge University Press, West Nyack, NY, USA.

Minsky, M. (1988). *The Society of Mind.* Pocket Books.

Miranda, E. (1999). 'The artificial life route to the origins of music', *Scientia*, 10 (1), pp. 5-33.

Miranda, E. (2001). *Composing music with computers.* Focal Press, Oxford.

Miranda, E. (2002a). *Computer sound design: synthesis techniques and programming.* (2nd ed) Focal Press, Oxford.

Miranda, E. (2002b). 'Emergent Sound Repertoires in Virtual Societies', *Computer Music Journal*, 26 (2), pp. 77-90.

Miranda, E. (2003). 'On the evolution of music in a society of self-taught digital creatures', *Digital Creativity*, 14 (1), pp. 29-42.

Miranda, E., Kirby, S. and Todd, P. (2003a). 'On Computational Models of the Evolution fo Music: From the Origins of Musical Taste to the Emergence of Grammars', *Contemporary Music Review*, 22 (2), pp. 91-111.

Miranda, E. and Todd, P. M. (2003b). 'A-Life and Musical Composition: A Brief Survey'. *Proceedings of the Brazilian Symposium on Computer Music.* Campinas, Brazil.

Morales-Mazanares, R., Morales, E. and Wessel, D. (2005). 'Combining Audio And Gestures For A Real-Time Improviser'. *Proceedings of the International Computer Music Conference.* Barcelona, Spain.

Moroni, A., Manzolli, J., Zuben, F. V. and Gudwin, R. (2000). 'Vox Populi: An Interactive Evolutionary System for Algorithmic Music Composition', *Leonardo Music Journal*, 10, pp. 49-54.

Murray-Rust, D. (2006). 'MAMA: An architecture for interactive musical agents'. *Proceedings of the European Conference on Artificial Intelligence*. Riva Del Garda, Italy.

Murray-Rust, D. and Smaill, A. (2005). 'Musical Acts and Musical Agents'. *Proceedings of the MUSICNETWORK Open Workshop*. Wien, Austria.

Muscutt, K. (2007). 'Composing with Algorithms An Interview with David Cope', *Computer Music Journal*, 31 (3), pp. 10-22.

Narmour, E. (1990). *The Analysis and Cognition of Basic Melodic Structures.* University Of Chicago Press.

Nattiez, J.-J. (1990). *Music and Discourse: Toward a Semiology of Music.* Princeton University Press, Princeton.

Neisser, U. (1967). *Cognitive psychology*. Meredith Publishing Company.

Orio, N., Lemouton, S. and Schwarz, D. (2003). Score Following: State of the Art and New Developments. Conference on New Interfaces for Musical Expression. Montreal, Canada.

Pachet, F. (1994). 'The MusES system: an environment for experimenting with knowledge representation techniques in tonal harmony'. *Proceedings of the Brazilian Symposium on Computer Music*. Baxambu, Brazil.

Pachet, F. (1998). 'Sur la structure algebrique des sequences d'accords de Jazz'. *Proceedings of the Journees d'Informatique Musicale*. Agelonde, France.

Pachet, F. (2000). 'Rhythms as emerging structures'. *Proceedings of the International Computer Music Conference*. Berlin, Germany.

Pachet, F. (2002a). 'Interacting with a Musical Learning System: The Continuator', in: Anagnostopoulou, C., Ferrand, M. and Smaill, A. (eds). *Music and Artificial Intelligence, Lecture Notes in Artificial Intelligence.* Springer Verlag, pp 119-132.

Pachet, F. (2002b). 'Playing With Virtual Musicians: The Continuator in Practice', *IEEE Multimedia*, 9 (3), pp. 77-82.

Pachet, F. (2002c). 'The continuator: Musical interaction with style'. *Proceedings of the International Computer Music Conference*. Gothenburg, Sweden.

Pachet, F. (2003). 'The Continuator: Musical Interaction With Style', *Journal of New Music Research*, 32 (3), pp. 333-341.

Pachet, F. (2004). 'Beyond the Cybernetic Jam Fantasy: The Continuator', *IEEE Computer Graphics and Applications*, 24 (1), pp. 31-35.

Pachet, F. (2006a). 'Enhancing Individual Creativity with Interactive Musical Reflective Systems', in: Deliège, I. and Wiggins, G. (eds). *Musical Creativity: Current Research in Theory and Practice*. Psychology Press.

Pachet, F. (2006b). 'Interactions reflexives: du «c'est marrant» aux machines a Flow', in: Orlarey, Y. (ed). *Actes des rencontres musicales pluridiscipinaires*. Lyon, France: Grame.

Pachet, F. (2008). 'A Musical Turing Test'. Available at: http://www.csl.sony.fr/~pachet/Continuator/VPRO/VPRO.htm. (Accessed: 10/08/2008).

Packard, A. S. (2007). *Lamarck, the Founder of Evolution: His Life and Work*. Dodo Press.

Paine, G. (2002). 'Interactivity, where to from here?', *Organised Sound Journal*, 7 (3), pp. 295-304.

Parncutt, R. (2008). 'Prenatal development and the phylogeny and ontogeny of musical behaviour', in: Hallam, S., Cross, I. and Thaut, M. (eds). *Oxford Handbook of Music Psychology*. Oxford, UK: Oxford University Press.

Payne, J. D., Stickgold, R., Swanberg, K. and Kensinger, E. A. (2008). 'Changes in Emotional Memory Across Wake and Sleep', *Psychological Science* (in press).

Pearce, M. and Wiggins, G. (2004). 'Rethingking Gestalt Influences on Melodic Expectancy'. *Proceedings of the International Conference on Music Perception and Cognition.*

Peters II, R. A., Wilkes, D. M., Gaines, D. M. and Kawamura, K. (1999). 'A Software Agent Based Control System for Human Robot Interaction'. *Proceedings of the International Symposium on Humanoid Robots.*

PGMusic (2008). 'PG Music Website'. Available at: http://www.pgmusic.com/. (Accessed: 09/10/2008).

Pinkerton, R. (1956). 'Information theory and melody', *Scientific American*, 194 (2), p 77.

Polansky, L. (1978). 'A hierarchical gestalt analysis of Ruggle's Portals'. *Proceedings of the International Computer Music Conference.* Evanston, USA.

Pressing, J. (1987). 'Improvisation: Methods and Models', in: Sloboda, E.J. (ed). *Generative processes in music.* Oxford University Press.

Puckette, M. (1996). 'Pure Data'. *Proceedings of the International Computer Music Conference.* Hong Kong University of Science and Technology, China.

Puckette, M. (2002). 'Max at seventeen', *Computer Music Journal*, 26 (4), pp. 31-43.

Puckette, M. and Lippe, C. (1982). 'Score following in practice'. *Proceedings of the International Computer Music Conference.* San Francisco, USA.

Ramalho, G. L. (1997). *Construction D'un Agent Rationnel Jouant Du Jazz.* PhD, Université Paris VI, France.

Raphael, C. (1999). 'A Probabilistic Expert System for Automatic Musical Accompaniment', *Journal of Computational and Graphical Statistics*, 10 (3), pp. 487-512.

Rissanen, J. (1983). 'A universal data compression system', *IEEE Transactions on Information Theory*, 29 (5), pp. 656-664..

Ron, D., Singer, Y. and Tishby, N. (1996). 'The Power of Amnesia: Learning Probabilistic Automata with Variable Memory Length', *Machine Learning*, 25, pp. 117-149.

Rowe, R. (1993). *Interactive Music Systems: Machine Listening and Composing.* MIT Press.

Rowe, R. (1996). 'Incrementally Improving Interactive Music Systems', *Contemporary Music Review*, 13 (2), pp. 47-62.

Rowe, R. (1999). 'The Aesthetics of Interactive Music Systems', *Contemporary Music Review*, 18 (3), pp. 83-87.

Rowe, R. (2004). *Machine Musicianship.* MIT Press, Cambridge, MA.

Russell, S. J. and Norvig, P. (2002). *Artificial Intelligence: A Modern Approach.* Prentice Hall.

Saltz, D. Z. (1997). 'The Art of Interaction: Interactivity, Performativity, and Computers', *The Journal of Aesthetics and Art Criticism*, 55 (2), pp. 117-127.

Schoenberg, A. (1983). *Fundamentals of Musical Composition.* Faber and Faber Limited, London.

Schottstaedt, W. (1994). 'Machine Tongues XVII: CLM: Music V Meets Common Lisp', *Computer Music Journal*, 18 (2), pp. 30-37.

Schwarz, D. (2003). 'Score Following Commented Bibliography'. Available at: http://recherche.ircam.fr/equipes/temps-reel/suivi/bibliography.html. (Accessed: 08/10/2008).

Scott, A. (1995). *Stairway to the Mind: The Controversial New Science of Consciousness.* Springer.

Snyder, B. (2000). *Music and Memory: An Introduction.* MIT Press, Cambridge, MA.

Stevens, J. (2008). 'The Bill Evans Webpages - Evans Quotes'. Available at: http://www.billevanswebpages.com/billquotes.html. (Accessed: 08/10/2008).

Storr, A. (1993). *Music and the Mind.* Harper Collins, London.

Styles, E. (2005). *Attention, Perception and Memory: An Integrated Introduction.* (1st ed) Psychology Press.

Temperley, D. (2004). *Cognition of Basic Musical Structures.* The MIT Press.

Tenney, J. and Polansky, L. (1980). 'Temporal Gestalt Perception in Music', *Journal of Music Theory*, 24 (2), pp. 205-241.

Thom, B. (2000a). 'BoB: an Interactive Improvisational Music Companion'. *Proceedings of the International Conference on Autonomous Agents.* Barcelona, Spain.

Thom, B. (2000b). 'Unsupervised Learning and Interactive Jazz/Blues Improvisation'. *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence.*

Thom, B. (2001). *BoB: An Improvisational Music Companion.* PhD, Carnegie Mellon University.

Thom, B. (2003). 'Interactive Improvisational Music Companionship: A User-Modeling Approach', *User Modeling and User-Adapted Interaction*, 13 (1-2), pp. 133-177.

Thom, B., Spevak, C. and Hothker, K. (2002). 'Melodic segmentation: evaluating the performance of algorithms and musical experts'. *Proceedings of the International Computer Music Conference.* Gothenburg, Sweden.

Thomas, D. A. (1995). *Music and the Origins of Language.* Cambridge University Press, Cambridge.

Titchener, J. M. and Broyles, M. E. (1973). 'Meyer, Meaning, and Music', *The Journal of Aesthetics and Art Criticism*, 32 (1), pp. 17-25.

Todd, P. M. and Werner, G. M. (1999). 'Frankensteinian Methods for Evolutionary Music Composition', in: Griffith, N. and Todd, P.M. (eds). *Musical networks: Parallel distributed perception and performance.* Cambridge, MA: MIT Press/Bradford Books.

Tokui, N. and Iba, H. (2000). 'Music Composition with Interactive Evolutionary Computation'. *Proceedings of the International Generative Art*. Milan, Italy.

Trivino-Rodriguez, J. L. and Morales-Bueno, R. (2001). 'Using Multiattribute Prediction Suffix Graphs to Predict and Generate Music', *Computer Music Journal*, 25 (3), pp. 62-79.

Ueda, L. K. and Kon, F. (2004). 'Andante: Composition and Performance with Mobile Musical Agents'. *Proceedings of the International Computer Music Conference*. Miami, USA.

Van Oostrum, P. (1995). *MF2T/T2MF*.

Vercoe, B. (1984). 'The Synthetic Performer in the Context of Live Performance'. *Proceedings of the International Computer Music Conference*. Paris, France.

Vercoe, B. (1986). *Csound : a manual for the audio processing system and supporting programs with tutorials*. Massachusetts Institute of Technology.

Vercoe, B. and Puckette, M. (1985). 'The synthetic rehearsal: Training the synthetic performer'. *Proceedings of the International Computer Music Conference*. Vancouver, Canada.

Verth, V. (2008). *Essential Mathematics for Games and Interactive Applications: A Programmer's Guide*. (2nd ed) Morgan Kaufmann.

Walker, W., Hebel, K., Martirano, S. and Scaletti, C. (1992). 'ImprovisationBuilder: improvisation as conversation'. *Proceedings of the International Computer Music Conference*. San Jose State University, USA

Walker, W. F. (1994). *A Conversation-Based Framework For Musical Improvisation*. PhD, University of Illinois.

Walker, W. F. (1997). 'A Computer Participant in Musical Improvisation'. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Atlanta, Georgia, USA.

Wallin, N. J., Merker, B. and Brown, S. (Eds) (2000). *The Origins of Music*. MIT Press, Cambridge, MA.

Weinberg, G. (2005). 'Interconnected Musical Networks: Toward a Theoretical Framework', *Computer Music Journal*, 29 (2), pp. 23-39.

Weinberg, G. and Driscoll, S. (2006). 'Toward Robotic Musicianship', *Computer Music Journal*, 30 (4), pp. 28-45.

Weinberg, G. and Driscoll, S. (2007a). 'The Design of a Robotic Marimba Player - Introducing Pitch into Robotic Musicianship'. *Proceedings of the International Conference on New Interfaces for Musical Expression*. New York, New York.

Weinberg, G. and Driscoll, S. (2007b). 'The Interactive Robotic Percussionist - New Developments in Form, Mechanics, Perception and Interaction Design'. *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction*. Arlington, Virginia, USA.

Weinberg, G., Godfrey, M., Rae, A. and Rhoads, J. (2007c). 'A Real-Time Genetic Algorithm In Human-Robot Musical Improvisation'. *Proceedings of the CMMR*. Copenhagen.

Welch, G. F. (2000). 'The Ontogenesis of Musical Behaviour: A Sociological Perspective', *Research Studies in Music Education*, 14 (1), pp. 1-13.

Wiggins, G. (1998). 'Music, syntax and the meaning of "meaning"'. *Proceedings of the First Symposium on Music and Computers*. Corfu, Greece.

Wiggins, G. (1999). 'Automated generation of musical harmony: what's missing?'. *Proceedings of the International Joint Conference on Artificial Intelligence*.

Wiggins, G. (2001). 'Towards a more precise characterisation of creativity in AI'. *Proceedings of the ICCBR 2001 Workshop on Creative Systems*. Vancouver, Canada.

Wiggins, G. (2006). 'A Preliminary framework for description, analysis and comparison of creative systems', *Knowledge-Based Systems*, 19, pp. 449-458.

Wiggins, G., Harris, M. and Smail, A. (1989). 'Representing Music for Analysis and Composition'. *Proceedings of the 2nd IJCAI AI/Music Workshop*. Detroit, USA.

Wiggins, G., Miranda, E., Smaill, A. and Harris, M. (1993). 'Surveying Musical Representation Systems: A Framework for Evaluation', *Computer Music Journal*, 17 (3), pp. 31-42.

Wiggins, G., Papadopoulos, A., Phon-Amnuaisuk, S. and Tuson, A. (1999). 'Evolutionary Methods for Musical Composition', *International Journal of Computing Anticipatory Systems*.

Winkler, T. (2001). *Composing Interactive Music: Techniques and Ideas Using Max*. MIT Press.

Woods, W. (1970). 'Transition Network Grammars for Natural Language Analysis', *Communications of the ACM*, 13 (10), pp. 591-606.

Wright, M. (2005). 'Open Sound Control: an enabling technology for musical networking', *Organised Sound*, 10 (3), pp. 196-200.

Wright, M. and Freed, A. (1997). 'Open Sound Control: a new protocol for communicating with sound synthesizers'. *Proceedings of the International Computer Music Conference*. Aristotle University, Thessaloniki, Greece.

Wulfhorst, R. D., Fritsch, E. and Vicari, R. M. (2001). 'SeVEM - Separador de Vozes em Execucoes Musicais Polifonicas'. *Proceedings of the Brazilian Symposium on Computer Music*. Fortaleza, Brazil.

Wulfhorst, R. D., Nakayama, L. and Vicari, R. M. (2003). 'A Multiagent approach for Musical Interactive Systems'. *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems*. New York, USA.