machine composition

between Lisp and Max ► maxlisp

Olav-Andreas Marschall

Hovedfag - IT-SLP - Vår 2007
Universitetet i Oslo

Machine Composition : Between AI and Music

Lisp, Max, Maxlisp and other recombinations

Preface and Introduction

Chapter 1 – Concepts and Products - Algorithms and compositions

Fundamentals of AI, computations, music and compositions

Chapter 2 – Processes and Objects – Musical intelligence and representations

Naturalized musicality and natural tools for constructing musical worlds

Chapter 3 – Objects and Messages – Machine composition with MAX

Patches using Max/MSP, Pd and jMax

Chapter 4 – Knowledge and Laws – Informed Machine Composition

Cypher and other systems

Chapter 5 – Rules and Probability – Inductive Machine Composition

EMI and other experiments

Chapter 6 – Lists and Sounds – Musical Lisp Environments

CommonMusic, CSound and other environments

Chapter 7 – New directions of Composing Machines

Paradigms and strategies

Chapter 8 – Integration of AI and Music in Composition

Max + Lisp = maxlisp

Chapter 9 – Concluding remarks

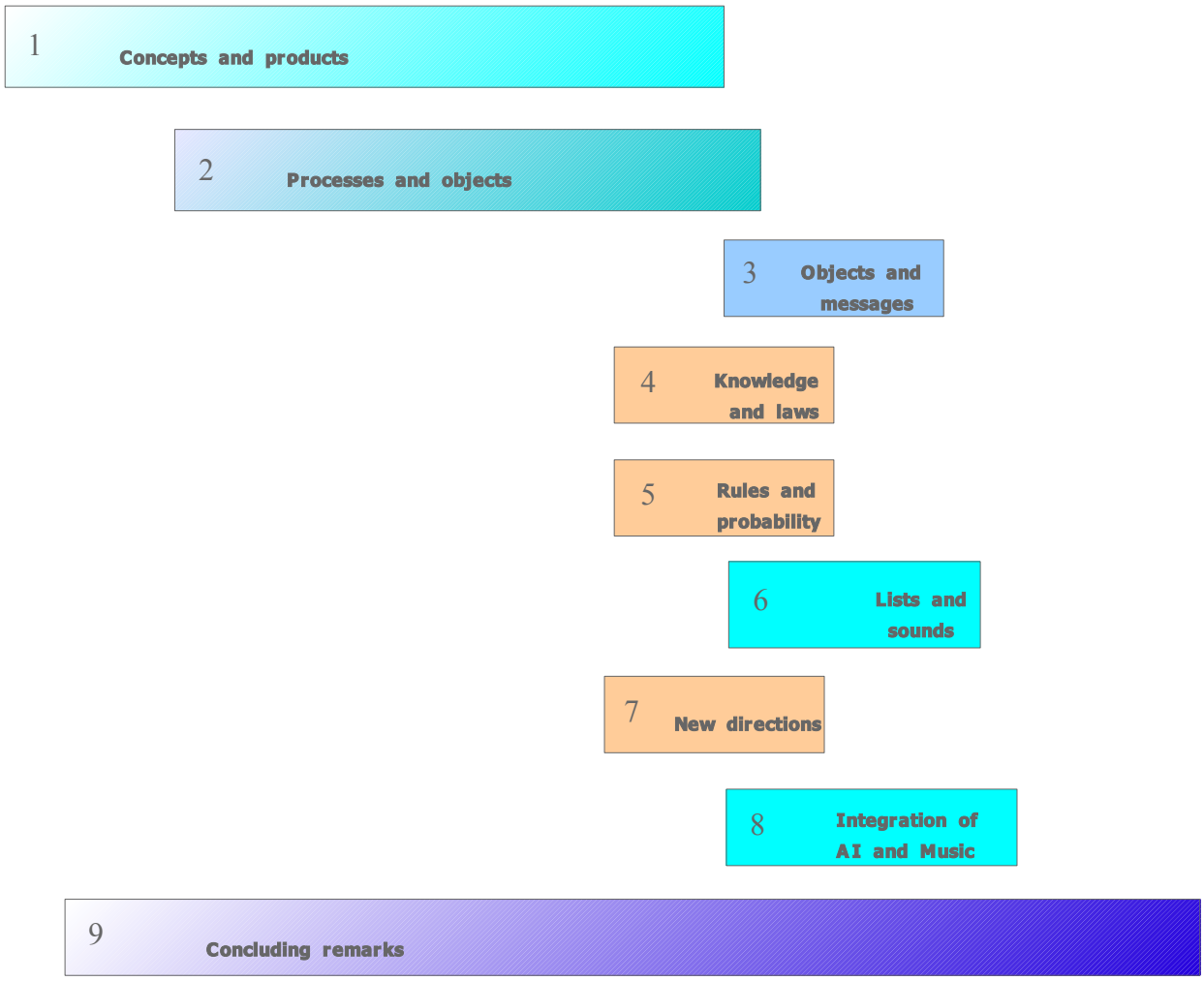
Between Phytagorean dreams and LaMettriean temptations

Short Summary

Appendix

Machine Composition: chapter chart

Concepts Science History Theory Tools Philosophy



Machine Composition

Between AI and Music – Lisp, Max, Maxlisp and other Recombinations

Preface and introduction

► *What is this paper about?*
This is an essay about algorithms and compositions; about music and machines. The first two chapters try to convince the reader of the naturalness, or at least the legitimacy of such combinations. From the very beginning, in *chapter 1*, algorithms are compared with compositions and vice versa, entities often thought of as cats and dogs. The answer is found one level higher up: cats and dogs *are living things (chapter 2)* and therefore essentially of *kind*. Chapter 2 is about empirical and conceptual issues regarding the nature and evolution of musical intelligence as well as key concepts of representation in music and AI.

The middle and main part of the text is about specific software tools and applications in this domain. Lisp and MAX, both widely used today, follow different computational styles to compose with algorithms: they are the prime subjects of chapter 3 and chapter 6.

In chapter 4, 5 and 7, I introduce some of the most influential academic accomplishments in practical Machine composing. Particularly Cypher and EMI have gotten reputations as prototypical systems of Machine Composition. These MC systems embed knowledge from musicology and AI, and make sense of 'machine musicianship' that comes in varying degrees of emerging musical intelligence.

In chapter 8, I take a look into 'maxlisp', an encapsulation of Lisp into MAX. This rather recent addition to the toolbox for music programmers makes it possible to combine the two probably most popular paradigms of musical programming. It is spiced up with some lines of code that may indicate a sample application of a pseudo *'Sonata-form-generator*' that may demonstrate some of the elegance of fusing Lisp-code with MAX-patches. This chapter concludes the practical section of this thesis. In the last and ninth chapter, I attempt to draw some modest conclusions and propose a few perspectives about Machine Composition (*MC*), creativity, and the philosophy of arts. I try to spin a web of both practical and philosophical implications collected so far.

► *Why is this subject of interest?*
The western intellectual tradition has sustained a rather sharp division between matters regarding machines and technology on the one hand, and the spirits living in arts and culture on the other. The deep-seated conflict between the “two cultures of Snow” seemed settled somehow in the post-modern times of inter-disciplinary collaboration. But tensions live on silently, sustaining separate paradigms for human-related fine arts and machine-related hard technology. Therefore, it should come as no surprise that arguments involving seemingly dissonant concepts from those fields can arouse bewilderment and opposition.

Many believe that such attitudes are still fertilized by common-sense conceptions related to Descartesian dualism (or dualistic 'folk psychology'): thinking of 'algorithm' as something that in *clear* and *rational* fashion describes the nature of *deterministic machines*, while looking at the arts (including music) as products of essentially *non-rational, free-wills* inhabiting *human artists*.

Nonetheless, there exists today a vast and impressive literature related to the field of *Machine Composition*. This pervasive and growing interest happens to be supported by the enormous possibilities of the emerging digital technologies. Perceptions of music as an art of beauty, populated by geniuses, have been challenged and are continuously provoking philosophical and aesthetic questions that cannot easily be suppressed.

But numerous approaches from different fields, angles and perspectives have still not led to the emergence of standard text books and leave us therefore with multiple conceptual systems and methodological approaches. Many conceive themselves correctly as pioneers and invent idiosyncratic terminology and schemes. The many wheels rediscovered by their numerous inventors make it impossible to survey the whole field within this limited space. I therefore choose to concentrate on several prototypical Machine Composition (MC) systems and tools that are put into several contexts of varying generality.

► *Why did I choose this subject and what is found in its chapters?*
I play instruments, experiment with sequencers on computers, and "talk Lisp" to microprocessors telling them exactly what to do. Observing computational functions that work according to one's intentions and listening to musical pieces that express the unifying idea behind a human work of art are not that different after all. A thesis about music composition and AI could therefore look like a logical conjunction of facts from each domain. It seems to me that the ubiquitous efforts to construct bridges between music and AI need to be taken seriously. I try to communicate the tension and investigate upon possibilities to pacify and dissolve some of the entangled issues.

► *Machine Composition as opposed to other titles?*
A comment about the use of key terminology. I use *machine* composing (MC) instead of the following equally well established alternatives: *virtual* music, *computer* music, *assisted* or *artificial* composition or *artificial* music. 'Artificial', refers to something that is not natural, as in 'Artificial Intelligence'. It seems kind of counterproductive in this context, since I claim the continuity of mankind with nature and evolution. More so, since 'art' and hence 'artifacts' are tied to musical compositions in general, 'artificial' seems to denote both too much and too little at the same time. Using 'virtual' as in 'virtual reality' or 'virtual pitch', with a similar meaning as 'quasi' or 'pseudo', poses similar concerns since it presumes the unreal or non-natural, associations I try to avoid as premises. It may be objected, that the concept of 'machine' carries similar connotations that reflect such dilemmas. We may remind ourselves of the conceptual evolution of the 'machine' meme from the days of 'Man a Machine' (LaMettrie, 1748) up to our robotic times where high-spirited accounts of 'Darwin's dangerous idea' (Dennett, 1996) prosper. And perhaps we can use 'machine' with sobriety, describing matter of facts and without associations of the comforting, but non-enlightening dualisms between the natural world and ourselves?

► *Who I owe words of thank and gratitude*
In a general sense, this thesis defends the view that absolutely all ideas and plans ultimately derive and descend from the great pool of recombinatorial evolution. Therefore I find the idea rather strange to reduce the causes of this paper and effort to a few persons. Still I would like to thank cand.philol Martin Bould, LL.M. Ola Tellesbø and Ph.D.Tor Martin Møller for their reading and commenting on parts of this paper, and last but not least cand.scient Terje Reite for his efforts to plough through the beast of chapter 9 and the encouraging words during our daily walks around campus. Then I will certainly thank Brad Garton for his kind reviewing of chapter 8 and expressing interest in this project. The unparalleled gratitude, on the other hand, must go to my supervisor of this master's degree, Herman Ruge Jervell, a seemingly neverending source of patience and wisdom, without whose support I certainly would never have completed this thesis.

► *This text assumes some elementary acquaintance with:*
Computational theory, programming/Lisp, AI, analysis and theory of music, music technology.

► *What this thesis is **not** about:*
Music synthesis, music performance, particular musical styles and works.

► A most informal guide for readers that want to jump right in or through this thesis:

- If you find the claim that composing really isn't that different from algorithms in computers interesting, offending or strange, you might want to read **chapter 1.**
- If you want to read about arguments that keep the origin of music (as well as that of 'Homo sapiens') in a continuous line with music technology, you might want to read **chapter 2.**
- If you are interested in basic forms and methods of machines that compose, you might like to read about the software tool 'Max' in **chapter 3.**
- If you are interested to know how humans and machines can play together in sensible ways, you may read **chapter 4.**
- If you think that machines will not be able to compose like historic geniuses did, you will probably be interested in reading the lessons of "Emmy", that has confused lots of people by not being able to distinguish between *her* ("Emmy composing like Mozart") and *Mozarts* own achievements. This is the narrative of **chapter 5.**
- If you would like to see how people had to program to make music before around 1990, you will find an example in **chapter 6.**
- If you are curious about the idea of dying *and* procreating algorithms and their programs, or want to read about a new type of "intelligent or active" instrument, **chapter 7** is the place to read.
- How can future machines be programmed in multiple ways and paradigms? This question is the subject of both **chapter 7** and **chapter 8.**
- If you are somehow philosophically minded and have a lot of questions about the whole thing, you may read about the whole landscape consisting of Machine Composition systems and about what may make a Machine compose in *beautiful* ways? These and other questions are, if not answered, spotlighted and touched upon in the final, but non-concluding **chapter 9.**
- **But if you are interested to know as much as possible about the whole issue of composing machines, you may well turn the pages from the beginning to the end to assure you are updated on all the definitions and theories that are needed in the study of this exciting subject.**

Abbreviations used:
MC : Machine Composition, **MCs** : Machine Composition system, **CM**: Computer Music, **AI** : Artificial Intelligence, **AIMA**: Artificial Intelligence, A modern approach (standard textbook), **TM**: Turing Machine, **AA** : Analysis of algorithms, **PP**: Paradigms of Programming, **MIDI** : digital music communication protocol, **CMN**: Common Music Practice, **CIM**: Composing Interactive Music, book by Winkler, **IMS :** Interactive Music Systems (book by Rowe in chapter 4), **Cypher:** name of knowledge-based protoypical MCs (Rowe; chapter 4),
EMI: Experiments in Musical Intelligence/prototypical inductive learning system (Cope; chapter 5),
CM: CommonMusic/Lisp-based MC environment (chapter 6), **CLM:** CommonLispMusic/ subset of CM dedicated to control microsound/synthesis of Csound units, **OOP:** Object-Oriented Programming, **GA**: Genetic algorithms,
GTTM: Generative theory of tonal music/ standard theory of deep structural meaning in generative/rule-based form (Jackendoff/Lerdahl), **PR:** Preference rules, in connection to GTTM, but also probabilistic systems (chapter 9),
CA: Cellular automata, **ANN:** Artificial network computing, **AN:** Association networks,
ALife: Artificial Life, **EM**: Emotion and Meaning (book by Meyer; chapter 9), **LA**: Languages of Art (book by Goodman; chapter 9).

Chapter 1
Concepts and products: algorithms and compositions
Fundamentals of computing, AI, music and compositions

1. Introduction
2. Algorithmic sources
3. Analysis of algorithms – quantifying algorithmic power and limits
4. Algorithmic solutions to Sudoku puzzles
5. Programming and composing
6. Programming styles and types
7. Algorithmic perspectives and music history
8. Musical programming and computational compositions (compugrams and algositions)
9. Conclusion: What makes algorithms and compositions different?

Mathematics and music! The most glaring possible opposites of human thought!
And yet connected, mutually sustained!
It is as if they would demonstrate the hidden consensus of all the actions of our mind,
which in the revelations of genius makes us forefeel unconscious utterances
of a mysteriously active intelligence.
(Hermann v. Helmholtz, "On the Physiological Causes of Harmony in Music", 1857)

1.1.Introduction

Algorithms provide the skeletons in computer programming and compositional plans provide the skeletons in musical productions. This is the central claim to be discussed in this chapter. Programmers apply algorithms as they build soft structures or 'code'. This code is interpreted and executed by a computer. The product of programming is a *computer program*. Composers use schemes or styles to conceive sound structures encrypted as symbolic scores that are interpreted and played by performing musicians. The product of composing is a *composition*.

Programming creates *machine contexts*, while composing predominantly involves *human contexts*. This may indicate, somehow circumstantially, why engineering and the arts are perceived so antagonistic in our culture. But is there a common ground ? Let us for the sake of the argument switch terms:

> *Composers use algorithms to build code that is executed by performers.*
> *Programmers create structures that are played by machines.*

Both domains seem to have their particular terminology with specific associations that consistently reinforce the idea of two "Snow-worlds".[1] What makes these ideas so convincing is the underlying and well-cemented conception that places the 'Arts' firmly inside the tradition of 19th c. romanticist aesthetics and on the other hand associates engineering with 'scientistic' ideologies of the 20th c. Both views are disputable and we should be able to find logical as well as empirical space for other and less polarizing approaches.[2] Let us start with the roots and nature of 'algorithm' and then clarify the concept of 'composition' in the light of this perspective.

1.2. Algorithmic sources

'Algorithm' (**A**) is derived from the Persian/Arabic mathematician Al-Khwarizmi and originally referred exclusively to the arithmetic rules for manipulating Arabic numbers. Nowadays, algorithm means somehow:

> *a definite procedure for solving problems or performing tasks. An algorithm consists of instructions how to do a task by splitting it into subtasks and sub-operations describing their order in a process.*[3]

'Algorithm' is a *'primitive'* concept. Due to its generality an exhaustive and precise definition seems impossible. Examples are easier to find though. The constructors of the Egyptian pyramids had to plan and solve their overall task of building Cheops by splitting it into subtasks. Subtasks that were temporally and spatially ordered from bottom to roof; not the other way around. Similarly, the writing of a paper, including this one, involves planning and procedures for solving each of the chapters individually, in an order that ensures overall coherence and efficiency after conclusion.

In most programming (or instructing) of computers it is necessary to be rigidly precise and complete. Algorithms are rules and techniques (definite procedures) for writing such instructions or code. Flowcharts represent and explain algorithms. What is the connection between data (e.g. numbers) and algorithms? Algorithms are *involved* with data; they use and control them. Data may be of any type, even though numbers are often favored due to their being native in 'digital' machines. Computing today (especially in AI) uses predominantly other data types or compound data structures (as in symbolic computing). Even physical movements and perceptions for instance, are now defined and engineered algorithmically[4] in robotics and sensor technologies. Algorithmic thinking shares with number systems and logics the fundamental ideas of *precision* and *determinedness*. Algorithms and mathematical functions abide so to say to the "ideology" of determining a *unique* element or step for any input or previous step (related to the notion of abstract Turing machines; see below).

But real-world problems often require decisions without sufficient knowledge at hand. And machines by their very nature do not stand out in such open-ended environments. Neither will they adapt well to novel situations. Programmers must therefore construct *closed* or at least *well-defined worlds* to ensure that completely specified programs (algorithms) will help to avoid situations of "doubts and hesitations" that ultimately may result in machine malfunctions.

Algorithms are vehicles used to formalize processes, actions and conditions. Algorithmic thinking goes along with civilizational processes and increases the degrees of control along with the "closing of environments", i.e. algorithms contribute to more predictable worlds by regimenting the ruling forces and thereby reducing the informational content of situations.[5] Mathematical rules that manipulate arithmetic expressions are prototype models for algorithmic formalizations. A formal criterion for well-formed algorithms is their possible implementation on (completely-specified) Turing machines[6]. In a Turing machine model one can intuitively relate to the *'halting problem'*, i.e. whether a specific algorithm describes a *terminating procedure*. Non-terminating procedures, like infinite loops, prove unsuccessful as algorithms. Successful algorithms may then be tested for practical applicability on physical machines and real problems. Is a certain problem NP-complete? Does this problem take more than polynomial time to execute?[7] Engineering and technology maximize *control* and *predictability*. The question posed is what are realistic and efficient algorithms in information design (data/algorithms) and information machines (executing implementations)? To answer such questions theoretical studies of algorithms are required.

1.3 Analysis of algorithms (AA) – quantifying algorithmic power and limits

Algorithms are general approaches for finding efficient solutions to problems. The theoretical study of algorithms, the analysis of algorithms (**AA**), a special field of the computer sciences, consists of theoretical and empirical analyses of the properties of algorithms. **AA** is *about* algorithms i.e. meta-programming. It applies discrete mathematics and theoretical computing theory to these analyses.

Why should algorithms be analyzed at all, and not just used? Essentially for two reasons: to know which algorithms work and what resources they require. But algorithms are not programs themselves; programs are defined at code level of specific programming languages. Algorithms are meta-programs or *procedural plans* that describe in general terms the behavior of many possible implementations (or programs) in *any* programming language. An algorithm can be written out or realized as many *program code versions* in different programming languages. Analogously, implemented algorithms or program code can be realized with *machine code* versions for various machines types and platforms. Machine code or *byte code* is the most *concretely specified* instruction type for machines; Algorithms on the other hand are the most *general* description of these operations.[8]

Algorithms are communicated through a mixture of natural and mathematical languages. **AA** looks for the material requirements of algorithms in terms of *space* (memory) and *time* (instruction cycles). Once a measurement language and method is established, different algorithms may be *compared* in order to make predictions of time and space requirements for specific algorithms based on knowledge about comparative algorithm classes.

Another aim of **AA** is to achieve conceptual rigidity and clarity in algorithmic design. Algorithms are classifiable as different paradigms or types. Examples of paradigms are '*divide and conquer'*, '*dynamic programming'*, '*greedy method'* and '*backtracking'*.

A classic algorithmic problem is the *sorting* of data. Sorting of unordered sets requires the *finding* and *comparing* of items, and *placing* and *replacing* items into sorted sets. All these operations require place and time in computing devices. Standard *merge sort* uses e.g. the “divide and conquer” strategy, that amounts to dividing the whole set into subsets that are solved recursively. *Recurrence* or *recursivity* are essential ingredients in algorithmic complexity. Recurrence relations in **AA** correspond to iterative and recursive programs. The introduction of *quick-sort*-algorithms makes programs more efficient. Even though merge-sort is considered theoretically optimal, quick-sort does significantly reduce the running times for *typical* applications. This touches on premises of **AA** about the *quality* of data. What kind of practical problems can be anticipated? Allowing any kind of unsorted data set, merge-sort is the most *balanced* algorithm. Compared to quick-sort, it will use more resources with “standard” distributions of data, but less for somehow “odd” distributions. Empirical and pragmatic considerations must be made in analyzing algorithms, including the use of a data or *input model* in addition to an algorithmic model.[9] This type of considerations are the core of the different perspectives in **AA**. Briefly, some practitioners give priority to *safe algorithms* and analyze algorithms with focus at worst-case or “upper bound” performances (“computational complexity”); while others are more inclined to study *applied algorithms* in models that analyze average-case performance primarily (e.g. Knuth), using best-case and worst-case only as limiting conditions.

Another important concept in the **AA** is the *generating function*. Generating functions are data-describing combinatorial tools that generate enumerated combinatorial structures, also simply called *symbolic method*. Algorithmic objects can be formulated as ordinary, exponential or bivariate

generating functions. To understand structures in this way the symbolic method derives *functional equations* from generating functions that display the corresponding *structural definitions*.

Generating functions are tools that describe the relationships of recurrences (structural similarity) to amounts of *running time* and *occupied space* as a function of the *problem size*. Generating functions play a constructive role, but do also clarify complex algorithms that would otherwise be difficult to analyze. In these cases, qualified estimates of asymptotic behavior of coefficients lead to useful approximations of *computational complexity* (**CC**), employing classical mathematical functions:

> In computational complexity, the O-notation is used to suppress detail of all sorts: the statements that Mergesort requires O(N log N) comparisons hides everything but the most fundamental characteristics of the algorithm, implementation and computer. In the analysis of algorithms, asymptotic expansions provide us with a controlled way to suppress irrelevant details, while preserving the most important information, especially the constant factors involved.[10]

CC is really more about classes of *problems* than classes of algorithms themselves, and brings us ultimately to the questions about tractability or non-tractability of problems in computational terms. (see below).

Common to all methods in **AA** are[11]:

- Specification of realistic input models
- Derivation of a mathematical description of performance
- Development of concise and accurate solutions
- Comparing variants and alternative algorithms
- Adjusting values of algorithm parameters

More general models are associated with classes of *combinatorial* algorithms, typical of abstract and mathematical problems, as for instance in the ordering of number sets (sorting). Classes of combinatorial algorithms are studied as permutations in trees, strings, words and maps a.o. We will look at some examples:

A fundamental data-structure for algorithms are *trees* and *forests*. The more constraints we introduce the more regimented will the sorting of structures become. From the most weakly structured acyclic connected graphs, forests, unrooted trees, unordered trees, ordered trees to the most rigid or orderly binary trees. A binary tree has one top-node. An algorithm for traversing binary trees will be well-directed and economical because every node has exactly one parent-node and two child-nodes. This structure is studied a great deal because it is a naturally recursive structure, i.e. any part of the tree is itself a subtree (of the same type). Tree data-structures and their accompanying algorithms implicitly model the behavior of recursive programs. Many properties of important algorithms are revealed through the study of trees and especially by following their tree paths. Search problems in **AI** belong to an early discipline that was born out of similar motivations. Recursive tree structures lead to closed-form expressions or recursive formulations to their generative functions. Analysis of tree algorithms express information about *tree length*, *tree height* and the specific *path-lengths* necessary to find “tree-solutions” to tree-conceptualized problems. Atoms of data-structures can be of other types than numeric numbers. Characters (bytes) and sequences of characters (strings) from fixed alphabets can be atomic entities as well. Combinatorial properties of strings, words and maps are used pervasively in text-searching algorithms (string-searching), but are also useful in semantic search, e.g. on the internet. Studying sets of strings in algorithmic terms leads ultimately to the study of formal languages. There are connections between fundamental theoretical properties of formal language systems and the analytic properties of the generating functions associated with algorithms for numerical systems.

Other studies in **AA** are about issues of efficiency of algorithms. Generally, it is easier to measure relative differences of complexity *between* algorithms, e.g. variants of sorting algorithms, than to define terms like 'effective procedure' or even 'computation' in *absolute* terms. These basic concepts are, just like the concept of 'algorithm' itself, without formally satisfying definitions. They are primitive or intuitive concepts that have to be circumvented by analogies and examples.

Limits of algorithmic thinking and the incompleteness theorem of Kurt Gödel
Logical systems that require higher order formulations than first order logic, e.g. like the system of natural numbers[12], will always contain demonstrably true and yet undecidable statements from within the expressive repertoire of Turing machines. This is Gödel's theorem in a short approximation. Hence, the truth of those "difficult" statements (subset of all statements) *cannot be proven* by any *formal algorithm of this language itself*. This follows from the (generally accepted but not proven) Church-Turing thesis that states that *Turing machines* (**TM**) are *universal machines*, i.e. they are able to compute *any computable* problem.

This is somehow equivalent to the fact that there are functions, e.g. on the natural numbers domain (or any other data in systems that contain mathematical inductions), that cannot be computed on **TM**, or in other words that there cannot be effective procedures or algorithms for computing them mechanically. Those "difficult" functions will not return answers, but will run forever on a **TM**, even worse so, there cannot exist any general procedure that will tell us in advance what concrete functions actually belong to this subset.

Those limitations of algorithmic thinking are called *undecidability* (non-provability of true statements) and *non-computability* (halting problem in **TM**) respectively.

In contrast to these absolute and theoretical limits of computability, **AA** *quantifies* the obstacles to computing, known as *intractability* problems. **AA** classifies problems as a function of the assumed time needed to solve them. Problems that exhibit polynomial growth in the time needed relative to the size of the problem are easier to solve than those problems that exhibit exponential growth in time (relative to the size of the problem). This does therefore not measure the necessary (absolute) resources for solving single algorithms, single procedures or programs; it only *compares classes* of problems using generalized descriptions. All the same, small problems of exponential time growth are easier to solve than big problems of polynomial time growth, so the relevance of this classification of formal problems or algorithms grows with the size of a particular instance. In other words, undecidability and non-computability may be seen as part of a theoretical (or **TM**-model) "semantics" while intractibility is better understood as theoretical "pragmatics".

But when are problems untractable? We start with the class of *P-problems* that exhibit polynomial-time-growth (relative to the size of the problem). They are the easiest problems, as long as problems are not sized too big. But P-problems will be solved by the growing numbers of computing resources in the future. The bigger sized the problem, the bigger must the resources be to solve it (O(n); O(log n)) *Non-deterministic problems* (NP) on the other hand can be solved in polynomial time by testing guessed solutions. In contrast to P-problems that can be solved by "brute force", provided sufficient force, NP-problems can only be solved indirectly or *heuristically*. NP-problems exhibit exponential-time-growth (relative to problem size) and are therefore inherently *hard problems*. An assumed sub-class of NP are made up of the *NP-complete* problems with higher order time-growth exponentiality.[13]

AA is hence the theoretical study of practical algorithms and their relative efficiencies, costs and complexities. Logical and philosophical problems of computability, such as incompleteness,

undecidability, non-computability and intractability take issues of consistency and complexity into the mathematical and philosophical domain of formal systems.

1.4 Algorithmic solutions for Sudoku puzzles

Let us now construct an example algorithm for a well-known problem that applies to numbers or for that matter any other symbols: **SUDOKU (S)**. This originally Japanese game is a puzzle that comes in several sizes (problem size!) and typically consists of a varying number of *cells*, ordered in *rows* and *columns* of a square or *grid* (*figure 1*). In the following example there are 6*6 cells, i.e. a grid of 36 cells (Sudoku's standard size is 9*9/81). A "S problem" starts with a subset of the total 36 numbers already filled in, called *givens*. Rules are as followed: only numbers from 1 to 6 are allowed and all numbers must appear once only in rows, columns as well as sub-squares (or sub-grids)..

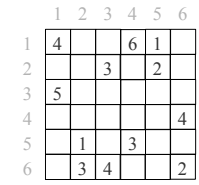


Figure 1: Sudoku grid (problem)

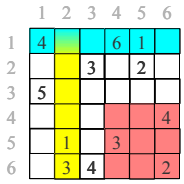


Figure 2: Sudoko row (blue), column (yellow) and subgrid (red)

It is easy to see structural recurrences (see generative functions; **1.3**). Besides the equal-sized sub-grids (*g*) of size 2*3, there are rows (*r*) and columns (*c*) of identical length. Rows and columns contain 6 cells, but in different constellations; *r* and *c* are intrinsically identical data-structures (***s***), rotated by 90° /270°.

A particular **S** (problem) always starts with certain cells filled in (pre-filled cells or *givens*) from the beginning. These *givens* function as facts. They act as premises for the filling of the remaining cells, i.e. empty cells, that must satisfy the Sudoku rules of consistency (**C**). Such rules state simply that any Sudoku-structure, i.e. column, row or sub-grid must contain a complete set of the possible values; in other words: no value can occur more than once, or by equivalence no value from the value domain remains unused. Valid values are in [1...6] for 6*6-grids, in [1...9] for 9*9-grids[14] and so on. Facts, valid values and rules of consistency collectively constitute the conditions for filling the empty cells. What are then the subtasks and strategies in the search for valid values? And in what order should those strategies be applied to ensure efficient algorithms?

Subtasks are defined as the incremental filling of structures *s* (rows, columns, sub-grids). There are several obvious algorithmic strategies:

> **A.** Scanning for determined cells first (*'cross-hatching'* and *'counting'*)
> Are there structures ***s*** ={r, c, g} with a single cell left empty? If so, the remaining unused value of the value set can be filled in. We deduce its last value directly by exclusion.
> If more than one cell is empty in a *s*, we check with intersecting *s* for additional constraints. That means we look at several *s* at once and apply C to all. This may or may not eliminate sufficient of the remaining choices to determine further cells. If it doesn't, other strategies must be followed.
>
> **B.** *Eliminating* candidate cells in a *s*
> This is basically the inverse strategy of above: it eliminates possible candidate cells through looking for contingencies, i.e. narrowing down the possible locations of unused numbers to subsets of cells in a ***s***. When subsets of cells for a specific value are also subsets of another ***s***,

these strategies may determine or at least narrow down the options for a single cell by *elimination*.

C. *Logical analysis* (deduction from multiple constraints)
When scanning (**A.** or/and **B.**) does not supply additional fills, logical analysis must be applied, i.e. hidden relations must point to transformations and hence solutions of cells. This amounts to the *collecting* of statements of constraints and continual analysis for detecting of "cell singularity". Humans do this typically by so-called sub-script or dots notations. It implies that even non-adjacent *s* may contribute to the elimination of candidate cells or numbers; such constraints are often opaque to the "human eye" but transparent to logical analysis.

D. *Analysis* (twins and triples)
Eliminating candidate cells by "unmatched candidate reduction or deletion".[15] Cells with identical candidate numbers (see **B.**) are matched *collectively* once those cells have identical sets of candidate numbers *and* the number of candidates equals the number of cells. A triple in a s for instance means that C_1:14[16], C_2:45 and C_3:15 exclude any other candidate numbers other than 1,4, and 5 from the set of cells $\{C_1, C_2, C_3\}$, hence restricting the possible placing of *other* remaining candidate numbers to the *other* remaining empty cells in s.

E. *Hypothetical placement* of non-determined cells and backtracking (generate and test)
When no direct (**A.**) or indirect placement (**B.**,**C.**,**D.**) is possible anymore, preliminary or tentative placement of an "unbound" value x in a cell that permits either x *or* y can be tried (induced from **C.**). The value x is hypothesized only. One chooses (arbitrarily?) x as one of two mutually exclusive candidates, and continues from there trying to generate an inconsistency (duplicate values in any *s* from the generated "paths"). If x turns out to be false, one "backtracks" or annuls the filled cells after placing x. Since x was demonstrably false, y must be true: therefore one can confidently fill the cell with y and continue from there. This strategy is also called "as-if-approach".

Strategies based on scanning and analysis solve different types of grid problems. Grid problems are easy if they require only scanning (**A.**) and difficult if they require hypothetical placements (**E.**). Grid problems are even more difficult when several analytic strategies must pull together to solve a candidate number or place. It would seem that the ratio of givens to non-givens is a measure of the difficulty of solving (filling) a grid. This is not the case. Even though this ratio is not irrelevant, the quantity of *information* in a constellation or pattern of givens is more important. Some givens are redundant relative to possible consequences or constraints for the filling of empty cells. Therefore some grids with few empty cells can be actually harder to solve (requiring more analysis) than certain grids with more empty cells but much information by their givens[17] (requiring mostly scanning). Let alone the non-trivial questions of which strategies to apply and in what order? And last but not least, in which order should the structures be examined using certain strategies?
The application order of the strategies is by no means arbitrary. The first scanning strategies are more visual or intuitive; the last strategies are increasingly opaque and logical in their nature. This is further reflected in the fact that **S** problems that are classified as easy problems can be solved using the first and basic strategies exclusively while progressively difficult problems require the application of all strategies in *specific orders*. Another observation is that all strategies are somehow analogous to logical or mathematical transformations of equations that deduce unique solutions of variables. The last strategy (**E.**) even relates to mathematical proofs and search problems with backtracking.
Now, should rows (**r**), columns (**c**) or sub-squares (**g**) be checked first? Left to right? Checking 1

upwards to 9? Such choices seem arbitrary. But they certainly are not without significance. From a higher level perspective some givens are more eliminating than others and some ***s*** are more determined or determining than others. Such heuristic criteria are working similar to how humans solve Sudoku problems. But could these problems not be solved by brute force alone, using backtracking and systematic generate-and-test exclusively? This means to start from any empty cell and trying any possible value, and backtrack whenever inconsistencies will be generated? If a Sudoku grid is small enough, this is possible by a mechanical procedure (on a computer). But if the grid is bigger than a certain threshold, it will just as well be impossible. The search tree or search space will become too big. Sudoku is a NP-complete problem type. In other words, if the problem size or grid size is big (enough), systematic backtracking (trial and error) will take more than polynomial time to execute. On the other hand, any problem solution of a whole task (whole grid) or subtask (candidate number in a *s)* is *testable* in polynomial time with a *specific* solution candidate. Therefore a successful algorithm will have to apply possible strategies in such a way that the supplied information by the givens and their *s* will be utilized *optimally.*
The algorithmic rules of heuristics include e.g. steps that have

- to scan, eliminate and analyze *first* **s** that seems *closer* to fills (in that order)
- to *recheck* all relevant strategies after a successful fill and
- to check the candidate numbers that *occur more often* than other numbers first.

Such rules are analogous to rules in search algorithms that find shorter (or shortest) paths through a search space to a goal state using heuristics (where pruning the tree amounts to removing *seemingly*[18] non-productive paths of the search tree).[19] These problems are typical and well-known in Artificial Intelligence (AI).

Sudoku-problems or grids with givens that are "easy sets", may be solved by scanning only. Difficult grids will come to a halt after scanning, but remain unsolved even after analysis. They will require "what-if-speculation" or hypothetical placements (**E.**). But even the generation of hypothetical candidates (x) may be guided by heuristics to identify best places in the grid to start from.

'Weak AI' tries to solve difficult problems, typically solved by humans. 'Strong AI' is more ambitious. It tries to solve such problems not in any conceivable and successful way but in the "exact" way that corresponds to the human solving process of that problem (i.e. *emulating* human intelligence).[20] But humans will use different approaches and strategies with different grids, and humans switch dynamically (perhaps even whimsically[21]) between strategies. This makes strong **AI**-solutions for **S** considerably harder. E.g. humans often find counting or scanning (**A**) of ***s*** boring (after some time), notation (**B, C**) time-consuming and forms of analyses (**C,D**) and "what-if-analysis" (**E**) obscure.

Different implementations of algorithms will lead to different steps, different order of steps, and different number of comparisons and inferences in the computer (due to memory and instruction requirements for a machine execution). As a general rule though, the more elaborate and well-formed an algorithm the shorter time will a solver (whether machine or human) need to fill the grid. Again, using only scanning strategies (**A.**) will solve only easy grids. Such grids may be solved by deterministic finite automatons or **TM** that potentially visit the whole game tree. Using only **E.** without heuristics will not lead to any solution at all for bigger sized grids (NP-completeness). The number of valid 9*9 solution grids has been calculated to *6.670.903.752.021.072.936.960* (where even the finding of this number alone was a challenge[22]). This number illustrates the reality of NP-completeness even in an innocent looking Sudoku example **S** as it also denotes the number of micrometers to the next star from the earth.

1.5 Programming and composing

We have looked at the concept of 'algorithm' and related issues from computer programming. Algorithms are guides to programming, may be like an architect's building plan controls the constructions of physical structures in space.[23] Algorithms are the *structure*, a fitting together of parts into a *con-struction* or building.
Composers con-struct musical objects from atoms into temporally ordered sequences in a process that is guided or determined by their ideas or/and rules. The organization of a musical work or com-position is also called its *musical form*. The concept of com-position basically stands for 'putting together' something and is a key concept for many, if not all, cultural domains (design, architecture etc.). Musical forms are created by applying compositional techniques and methods. Fundamentally we can sort them into categories of repetition, variation and contrast. Naturally, musical forms are not the product of accidental or random processes. Nonetheless "experimental" techniques that transcend deterministic procedures of formal composing[24] have been in use. But by definition, or at least convention, composing is basically a planned and deliberate process of creating sound structures with some degree of inherent order and organization. The compositional process must therefore be guided by mental and cognitive processes that build intentional content. Such processes are usually conscious, but even musical improvisation shows clearly elements of both semi-conscious and semi-planned structuring on the borderline to composition. We will look further into differences between various aspects and phases of musical activity and creation in later chapters **[ch2,ch4]**. At this point we only presume that composition *typically* involves some degree of planning and rule application.

The generality of Sudoku as puzzle

We suppose that composing is guided by rules and plans and that compositions are intentional structures at least to some extent. But those are the qualities we find in computer programming as well. Are programming *and* composing after all both algorithmic *and* creative activities?

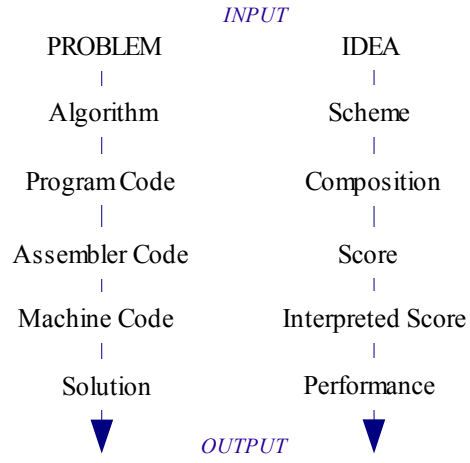


Figure 3: concepts and products

There are obvious similarities. First, both processes start out from somehow disorderly material (a situation that is viewed as a potential for a better situation), interpreted as a problem or idea. A problem can be *solved*; an idea can be *realized*. In both cases, an input (problem or idea) is transformed or developed into an output as solution or performance. Choosing of an algorithm, i.e. a systematic transformation plan, parallels the choice of a general scheme or musical form. Algorithms or schemes are then applied to generate the program code or composition. Different implementations or interpretations may lead to different assembler types or scores. The solution or

performance is in the end the result of a machine or performer interpreting code or scores. A solution or performance realizes finally the potential of the problem or idea. Solving a puzzle computationally presupposes systematic rules that transform puzzle pieces into a whole picture. “Performing an idea” amounts to coming up with meaningful ways of developing the idea into a whole composition that somehow expresses it. Using an algorithm involves the identification of subtasks and recurrences in the transition from problem to solution. Likewise, using a musical scheme/form involves the generation of repeated substructures, transformational functions and effective contrasts in a final composition that adequately realizes the original idea.

Let us now play with the Suduko board and use it as a musical composition problem. We assume that we operate in the material domain of a 'sextatonic'[25] scale, i.e. six equidistant half tone steps from 1 to 6. As a consequence, the **S** example from above works interchangeably in this new context. We may even substitute numbers with note names **(figure 4)**. It turns out to be perfectly in line with the definition of **S**, reminding ourselves that **S** is not intrinsically numeric but a general puzzle problem like the Cubric cube and many other variations on this theme.

	1	2	3	4	5	6
1	F#			A#	C	
2			E		D	
3	G#					
4						F#
5		C		E		
6		E	F#			D

Figure 4: Sextatonic Sudoku

	1	2	3	1	2	3	1	2	3	1	2	3
1	F#			A#	C							
2			E		D							
3	G#											
4						F#						
5		C		E								
6		E	4			2						

Figure 5: Four bars in 6-tone style

If our compositional idea is the “melodic material” of **figure 4** (essentially our **S** problem from figure 1 from above), we may choose a framework, scheme or musical form to develop it into a full-blown musical piece. We choose a smaller cousin of the twelve-tone-style (dodecaphony) by Schönberg[26], calling it 'sextaphony' or 'six-tone' style. This style requires that every tone of our scale of six must be used exactly once[27] in each voice (row) and chord (column) of the composition. A composition lasts for two bars of 3 beats (one row of **S** consists of “two bars”). We add the additional constraint that no tone is aloud to be repeated (i.e. all tones are used) in any bar's low, middle and high voice group (sub-grids corresponding to the soprano, alto/tenor and bass part of a choir). Voila! Our sextatonic style is equivalent to the rules of **S**. As a consequence our composition algorithm is now formally identical to solving Sudoku problems.

Extending and iterating the grid, we draw iteratively more bars and obtain something that essentially becomes a notational system with six voices.[28] Polyphonic music of the Renaissance period had mostly six voices that were developed according to strict laws, not too unlike the rules we have formulated for **S**. Certainly, they were much more liberal than our rule for columns. They permitted repeated notes, but banned certain short distances between notes (only fourths, fifths,and octaves were legitimate intervals). Instead of our **S** subgrid-rules, Renaissance-counterpoint asked for “harmony” (or balance) between the lower and higher voices and other necessary voicing conditions (especially about relations between rows 1 and 6 in **S**).

But most composition problems have more than one possible solution, countless actually. Which other constraints or rules should the composer “choose”? The **S** -solution expresses my idea best in this context, because I make this particular argument. In general, choosing of constraints is very much a *creative* process inherent to composition. But is not any process of finding valid solutions

creative to some degree, since they involve the finding (and selecting one of the valid alternatives) of a path from the idea to the consistent sextatonic composition?[29] In figure 5 we extend the structure or composition with an empty grid (empty bars). Should we transpose the "motives" (patterns) from grid 1, or vary them otherwise? Or should we introduce completely new patterns (counter-subjects) that will contrast with the givens of grid 1 (the starting or original idea)?

We may further liberalize our **S** style by permitting empty cells (pauses) and thereby increasing the compositional freedom, equivalent to a higher number of legal solutions or "style-abiding" compositions relative to the number of false solutions or "style-breaking" compositions. Generally, we conclude that the more givens there are in a grid and/or the more constraints or rules a problem must take care of, the less solutions are possible. This parallels the situation of a composer that starts with rather determining ideas and chooses a rather strict style for composing leaving him with little compositional freedom. Music composition teaching distinguishes between 'free composition' and 'style-based composition' (mostly in the tradition of the nineteenth century). But free composition is certainly not meant in the sense of free *of* style altogether, but captures the idea of being free from *specific* and *established* styles.[30]

In a general sense we may now see processes like composing and programming as *puzzles* that are to be solved in accordance with given rules of a *game*. Are music and other cultural conventions really so unlike puzzles, problems and Sudoku games?

Programming stands for the activities of planning and determining cultural expressions and constructions[31]. Often, programming is understood narrowly as the finding of solutions to problems. But programming understood in this way, is involved in the process of com-posing musical ideas into full-blown compositions. There may be single optimal solutions at times, but more often than not there will be many different but equally successful alternatives of solutions. Sometimes the optimality of one solution may mask its sub-optimality in terms of other criteria. Musical scores and performances can *appear* to be logical conclusions (necessary) to their premises (ideas). But most typically, musical compositions are the result of ideas that could have been realized in many other alternative ways and forms.

1.6 Programming styles and types

Computer programming is the craft of implementing interrelated abstract algorithms and concepts. The tools are programming languages that produce concrete programs as results (Wikipedia). Different programming languages support different styles or paradigms of programming (**PP**).[32]
A program or program code integrates algorithms with data. In some paradigms the distinction between program and data is fundamentally blurred (Lisp **[ch6]** or ANN **[ch2]**), i.e. data can be programs and vice versa.
A **PP** is a general view of how computational "things" (data, algorithms, code) should behave during execution in a computer. One may favor, loosely listed:

- distributed structure programming (unstructured programming:BASIC or ANN)
- structured programming based on either procedures (Pascal), objects (Smalltalk, C++), events (Perl), tables (databases), components (OLE), functions (Lisp)
- functional programming (Scheme, Haskell, Lisp)
- declarative and logic programming (Prolog, Expert systems, constraint programming)
- flow-driven or dataflow programming (Pipeline, message-oriented)

Programmers will often look at their programs as mostly concrete collections of interacting objects, sequences of commands or function evaluations. Many programming languages permit or even

recommend combinations of PP (CommonLisp, C++, Phyton). Syntax of programming languages reflect their defining paradigms.

In the broader context of the integrated software development process, we may distinguish on a more general level between structural and explorative approaches. This distinction is really more about the attitudes of programmers in their use of paradigms and languages.

Structural programming prefers hierarchies in a top-down fashion following clear plans and intentions. It is presupposed that the program specification is stable, i.e. the "mission" is known from the beginning. Structural programming realizes a more closed model with input, output and other explicit-made conditions. It often assumes that data representation can be build up from bottom to top in well-defined hierarchies that then are implemented from top to bottom.

In contrast, *explorative programming* starts with only approximate ideas about the desired goals. Goals will emerge during the construction in a process of clarification (trial and error). Data objects and representations are loosely defined from the outset. Typical for this approach is the use of abstraction barriers for postponing lower level implementation until higher strata are conceptualized and tested. Structured and explorative programming reflect the differences in teaching composition as free composition or style-based composition.

Another relevant distinction lies in the *interaction* between programming environment and programmer. *Compiled* languages separate compile time and runtime (execution). The compiler performs syntactical and semantical tests/checking for consistency and optimization before code is executed in runtime. As a consequence compiled programs can be tested and observed only as larger units and not during coding.

Interpreted programming languages (without seperations of compile time and runtime) on the other hand permit code to execute directly in runtime. This makes it possible to test smaller pieces of code in a matter of instants. Interpreted languages make incremental and tentative programming easier because testing of parts independent from the rest is straightforward. On the downside, runtime errors are harder to trace back to their dysfunctional sources. Dialects of Lisp and Java are typically interpreted *and* compiling languages. C, C++ and many of the older programming languages are predominantly working in compiling mode.

1.7 Algorithmic perspectives and music history

> Techniques of algorithmic composition have been applied in European concert music for centuries. Implemented in computer software, however, these techniques have expanded into systems of music representation and production.
> (Ariza, Christopher, *An open Design for Computer-Aided Algorithmic Music Composition,2005)*

Music history is very much the history of musical compositions.[33] Musicologists study to a large extent the structure and organization of sound in compositions of various styles and periods. The ancient Greek word 'tonos' stood for rope, cord and tension, referring to the physical string that vibrates and causes air waves in complex patterns of the so-called harmonic series or overtone series. These partials (overtones) along with their fundamental are as a whole perceived as steady or *complex tones*. Such periodic phenomena are typical examples of regularities in music. Any wave is basically a repetition or pattern of pressure variations, just like any tone is a "composition" of waves or partials that make up the harmonic series of its resulting complex tone.

Regularities are also a necessary part of pleasing and coherent compositions consisting of complex tones. Scales were among the first regularity-providing structures to be exploited in “pre-historical” music theory and compositional practice. Later there was an increasing interest in polyphony[34] and hence harmonies in chords (stacks of complex tones). At first, in medieval practice, polyphony was limited to the “celestial harmonies” that were based on the privileged intervals of octave, fifth and fourth. Even though particular rules and laws for good practices change with time and periods, there are some over-arching principles for music. Music must integrate repetition, variation and contrast to become a successful art.[35] One might be tempted to see these most intuitive form principles in the light of the logical connectives to see some superficial (?) “congruences”[36]:

repetition	AND (evt. =)
variation	OR
contrast	NOT

All three principles are really quite algorithm-friendly. In musical “proto-coding” (planning process) and analysis, a *repetition* is often shown by identical letters, eventually using repeat symbol **:||** with the number of repetitions indicated above. *Variation* is by convention an apostrophed letter of the original and so on. While *contrast* is written by a different letter from that used to symbolize the contrasted theme. For example:

A A A' A'' A' [B :||] A B B A A

would be a 12-part musical piece with two contrasting 'subjects' (A vs. B) and where A is varied (A',A'') and both are repeated in total as four instances. Most of music notational symbols are easy to use in algorithmic descriptions of musical works. Musical structures are recurring or regular, like measures, beats and mostly triad-based chords.

Examples of algorithm-friendly rules in the history of music are many. An early example, from late-medieval motets, is the technique of isorhythm where an order of durations, independent of pitch is the ordering principle ('talea', mostly in the tenor voice). In later counterpoint, the underlying form principles are made of complementary rhythms, a balance between melodic and harmonic order (combinations of lines that result in harmony) and effective devices like canons and imitations. Rule systems for counterpoint were continuously developed and at a point so successfully refined that the version of Fux (Gradus ad Parnassum, 1725) became itself the canon of counterpoint technique. Margaret L. Wilkins describes the “beauty” of Bach's counterpoint:

> Both vertical and horizontal aspects were **cleverly calculated**. All of Bach's contrapuntal lines could be **accounted for** within the **tonal harmonic system**, as if laid on a **grid**. Chromatic harmonies, accented passing notes, and appoggiaturas were all part of Bach's formidable **technical equipment.** Since the collapse of the formal tonal system, composers have a degree of freedom not available to composers of the “Common Practice era. Nevertheless, the most convincing contrapuntal writing takes account of some **logical harmonic outcomes**.[37] [accentuations added in citation]

The “calculations, grids, technical equipments, formal tonal system and logical harmonic outcomes” are all descriptions very compatible with algorithmic thinking. Historical composers often did not learn by rule books but learned by examples.[38] They rewrote pieces of prior composer generations and analyzed and internalized those rules that can be used to understand the inner logic of most works of the “Common Practice”. Composers like Domenico Scarlatti wrote over 500 sonatas for keyboard, but used not more than a handful of schemes or forms.[39] The most applied, refined and varied musical form of Western Music history is without doubt the sonata form. Like the harmonizing style of Bach, the narrative form of sthe onata emerged from practice and was not

made explicit before the next generation taught them to their pupils as “master form” for successful music. The effective quality can be explained by the psychological reasonableness, found in the balancing of old and new material. The following is denoting the structure of an early string quartet in sonata form (Haydn[40], op.2, no.4, IV.):

aa (F) 10
bcde 24 :||
||: a/b (G-F) 14
aabcde (F) 28 :||

The finale of this quartet has five different subjects, with keys (letters in parantheses) in accordance to functional harmonics (I-IV-V-I) and it consists of 10+24+24+14+28+14+28 bars; 142 bars in total. Here is how Griffiths describes the “formal laws” or narrative flow of sonata form:

> The first part of such a form, or 'exposition' as it has come to be called, moves from the tonic or home key to the dominant, conventionally a fifth above in the case of a major key, or else in a minor key to the relative major, i.e. the major key with the same key signature: the general term 'secondary key' will sometimes be used here to indicate the dominant in a major key or the relative major in a minor key. The move from tonic to secondary key is a basic facet of the major-minor system: simple tunes very often modulated from tonic to dominant in their first halves and then back to the tonic in their second. In sonata-form movements this move is simply lengthened and emphasized, often by having a second theme or group of themes in the dominant. Then the symmetry of return is delayed by a period of waiting, or 'development', where the composer can explore conceivable avenues of escape from the inescapable: the return of the 'exposition' and the tonic key. And this 'recapitulation' is customarily made more decisive by cutting out the earlier move to the dominant, so that all the 'exposition' material is now heard in the home key.[41]

A less dramatic or picturesque description is found in Wikipedia, but this citation shows how much formal considerations are integrated with harmonic considerations referring to counterpoint.[42]

Mozart's “Musikalische Würfelspiele” is an often cited example of algorithmic music. His composing game consisted in finding a set of melodic sequences that were interchangeable (in the sense that they all ended and started neutrally for this group) and in turn combining them non-deterministically using througing dices as decision method. John A. Maurer sums up:

> This very simple form of “algorithmic” composition leaves creative decisions in the hands of chance, letting the role of a dice to decide what notes are to be used.[43]

In the 20th c. algorithmic composing with elements of chance had regained attractiveness (under the names of aleatoric or stochastic music). Randomness is a “repeated pattern” in this thesis, without becoming a main focus in the discussion of the algorithmic nature of music composition. All in all, randomness has always been the expression of a minority in music history. The opposite “super-style”, e.g. serial or twelve-tone-method of the second Viennese school (the first was associated with the sonata form) consciously employs algorithmic processes in their rule-generated musical forms, and in some of these forms even ending up with total determinism. As we saw from our example of Sudoku-counterpoint above, fewer choices are left to the composer as constraints increase in number and significance.

In general, we may want to conclude that composing activities (constructing structures) depend more on rules, rule systems and constraining practices than is often admitted. The obvious equivalent to the programming algorithm is the composing *plan*. Wilkins writes:

> Contemporary composers, however, often need to undertake structural planning as part of the composing process [44]... To use another metaphor, plotting a composition is rather like planning a journey. The more ambitious the expedition (to the Antarctic, or the Amazon, for example) you will not be able to keep returning

> home for vital equipment that you have forgotten. Forethought and imagination are vital to a successful undertaking. Even a holiday to another country has to be planned and booked; however, a simple walk around the park can be pleasurably undertaken without much effort.[45]

This type of structural planning is not unlike programmers' effort to plan datastructures and recursive functions in *their* journey from the problem to the solution. Sceptics will often turn to musical styles of romanticism to make a case for the non-algorithmic nature of music. Let us therefore look at some of the supposedly hardest counter-examples.

At the zenith of musical romanticism there are two antithetical aesthetic schools: the first continues and expands classical forms and languages of tonality and rhythm (Bruckner) and the other looks for new ways to express *extra-musical* content in programmatic and tradition-transcending ways (Debussy, Wagner).

Debussy is known for his form-dissolving style and his intentional weakening of the tonal center or structure. Musicologist Reti (1958) calls this a “new concept of tonality in European music that is a synthesis of monophonic based melodic tonality that is different from the existing harmonic tonality”. Still today, many use categories of 'impressionism' (or even 'associativism') for the style of Debussy, characterizations that Debussy himself found “imbecile”.[46] Often this characterization makes it more difficult to see Debussy's extensive work of planning and structuring of the tone materials.[47] Howat in “Debussy, Ravel and Bartok: Towards some new concepts of form”[48], analyzed a significant amount of works of Debussy and came up with an unexpected many mathematical relations, such as the golden ratio (sectional lengths), recursive or fractal structures (with minor sections mirroring the proportions of the higher level structures), and fibonacci sequences in melodic material. In addition there is no doubt that Debussy uses an intricate logic of weaving together the many rather independent motivic parts. Howat writes about Debussy's '*Reflets dans l'eau*':

> ...the formal and more static aspect of the piece, although it uses Golden Ratios, balances them to form a symmetrical whole, whereas the dynamic arch shown on the top part of Fig.4 is quite asymmetrical, being divisible in no way except by Golden Ratio. By joining the two parts of Fig.4 it becomes apparent how neatly and harmoniously the two strands of structure, dynamic and formal respectively, are dovetailed together, so that the subsidiary points of the dynamic arch form the main pillars of the overall structure – the one growing organically out of the other. And as a concluding link between the end of the dynamic arch and the end of the whole piece, the 32 crotchet beats of the cadenza plus coda are divided exactly 16:16 by the final cadence to the E*b* tonic after 105 beats (beginning of bar 29) – an obvious equivalent to the symmetrical 26:26-bar placing of the final return to the tonic key in 'Reflets dans l'eau'.[49]

This analysis does not really picture a pointillistic or structure-weak style. We may therefore propose to reconstruct his five agreed on stylistic features[50] in an algorithmic perspective:

Wikipedia:	*Comments in an algorithmic perspective*
Glittering passages and webs of figurations which distract from occasional absence of tonality	Variation, local structure
Frequent uses of parallel chords which are “in essence not harmonies at all, but rather 'chordal melodies', enriched unisons”	Interchanging of rules from different dimensions (melodic, harmonic)
Bitonality, or at least bitonal chords	Composition of multiple structures
Use of the whole-tone and pentatonic scale	Basic structures
Unprepared modulations, “without any harmonic bridge”	Contrast

The purpose of this musicological-algorithmic conversion is to narrow the gap from the sometimes inspirational or rhapsodical perceived qualities in Debussy's music to the processes of formal planning and their algorithmical composing style. In fact, as Howat sums up:

> This sort of study can raise as many questions as it answers, ...: for example the question as to whether such planning is conscious or the result of an exceptionally refined instinct... This analysis of music from the proportional and numerological standpoints is as yet a little-explored field, but one which could yield a greater understanding of music in relation to other arts, particularly drama and litterature... Proportional exactitude of all sorts can be found in the music of practically all well-known, as well as some lesser-known composers.[51]

Howat's numero-logical standpoint is very much algorithm-friendly. The fact that many listeners of "impressionistic" music hear little structure in it but an abundance of sound colors (evoking rather subjective associations like moods and pictures), is intended by composers like Debussy, but nevertheless more likely a product of the "ear" of the beholder than that of its creator[52]. From this timbral and mathematical approach found to be essential in the music of Debussy there is no insurmountable distance to the 20th c.'s activities subsumed under "Computer music" where both symmetry, mathematics and chance became parts of a new union between algorithmic methods (computer programs) and modernist aesthetics (John Cage, Xenakis and Hiller a.o.). But since *their* use of algorithms are not only conscious but even materialized in algorithmic computers we probably don't have to persuade too many of the existence of algorithmic techniques in composing 'computer music'. Computer music will be the object of discussion in several other places.

More of a challenge lies in the analysis of the compositions of the "cathedral romantic" Anton Bruckner. He was an enigma to many people because of his awkward, childishly naïvity and simplicity, so much in contrast with the "greatness and sublimity" of his music. Here are some observations by the conductor Bruno Walter:

> Bruckner sang of his God and for his God...gigantic whole...
> His super-dimensions, his surrender to every fresh inspiration and new, interesting turning, sometimes not drawn with compelling musical logic from what has gone before, nor united to what follows, his abrupt pauses and resumptions: all this may just as well indicate a defect in constructive power as an individual concept of symphony. Even though he may not follow a strictly planned path to his goal, he takes us over ways strewn with abundant riches, affording us views of constantly varying delight.[53]

Walter is deeply entrenched in the romantic attitude and language, but nonetheless communicates what many listeners even today experience. Bruckners symphonies are almost all written in sonata form, a form that with Mahler ended its natural life in opulence[54]. Sections, movements and symphonies had grown considerably in length and complexities. Bruckner even included explicit pauses between sections in the movements. There was a general expansion of everything (size of orchestra, size of subjects, number of subjects, counterpoint complexity and so on). Functional harmonics were stretched to the breaking point, using unprepared chords, far travelling keys, and extended tonalities. All this made the cohesive forces somehow feebler than the gravitational ones. Bruckners symphonies may seem chaotic and fragmented, but the inner logic was certainly present in Bruckners head. Bruckner had rigorously studied counterpoint and instrumentation. His teacher Sechter taught him for six years under the condition that he would not compose anything of his own until he finished his disciplined studies[55]. This made him the ardent craftsman that meticulously worked on methods and plans. In fact, as we will see in the next section, Bruckner may have been the composer that revised his compositions more frequently than any other. What would be the reason for such excruciating efforts? The conventional answer is that his insecurity let critical comments by his students and conductors[56], as well as his chronic self-doubts, become the cause of rewritings and new versions. This may be so, but an additional explanation may lie in the relational

richness of his works that made it nearly impossible for him to find solutions that satisfied *all* of the many rules emanating from his elaborate and complex systems constituting what many people feel is well described by the notion of "sound cathedrals".[57] Brucker has also been found to have "suffered" from "numeromania" (compulsory counting, number obsession). In this account the algorithmic nature of musical composing is naturally manifested in the craftsman's struggle[58] to incrementally ameliorate the consistent logical structure without excessively compromising with his high-reaching aspirations to find a "religious voice" and its soul-gaining integration of chorales of the *cathedral* with the Ländler of the Austrian *small towns*.[59]

1.8 Computational compositions and Musical programming (a possible synthesis?)

The paradigm of computing and programming of von Neumann machines very much favored a numerical and Fortran-like approach **[1.6]**.[60] We can see the predilection for numeric analyses and extreme cases of datastructures (Analysis of algorithms, **[1.2]**) as confirmation for this focus on hardware or machine issues on expense of the programmers software and his psychologically founded creativity. In the last decade several movements or schools of thought have responded and questioned this engineering viewpoint of programming. Extreme programming, e.g. wants to give priority (or at least a balancing of the traditional considerations) to adaptibility and changing demands in problem situations. Its values are listed as communication, simplicity, feedback, courage and respect. Values we easily recognize as commensurable with artistic activities.[61] Projects should be flexible enough to aloud for incremental changes over time (embracing change). Extreme programming methodology is thought to fit best in the development of new and prototype-like technologies and research where unpredictable requirements typically turn up underway. Extreme programming nourishes the underlying scepsis towards concepts such as hierarchy and quality control. It leads them to a greater belief in horizontally organized teams and internal as well as user-involving communication.

While 'Extreme Programming' may seem more like a corrective idea to mainstream computing, the 'Feyerabend Project' is nothing less than upright revolutionary in spirit. It presents itself as a redefinition of computing and comes with a polemic critique of 'computing theory'

> ... as a branch of computability theory based on a particularly inexpressive set of mathematical constructs. The proof of their universality under certain assumptions and equivalences to other mathematical constructs have irreparably boxed us into a realm of almost blinding inexpressiveness.
> (Feyerabend project at www.dreamsongs.com)

Both movements derive their vision out of an integral view on society and technologies. Power structure is a part of their analyses when they claim that "software development methodologies evolved under a mythical belief in master planning" and that the acceptance of "high-octane capitalism" is basically irrational. Their alternatives are distributed systems of "different-minded people" that realize that "everything changes, every version is necessary, evolution happens". Therefore their positive vision is a computing culture that is "based on utility, performance, efficiency and cleverness. Where are beauty, compassion, humanity, morality, the human spirit and creativity?", they ask us challengingly.

The 'Feyerabend project' honours the 20th c. philosopher Paul Feyerabend. Feyerabend made drastic conclusions (for example in his anarchistic theory of knowledge,"Against Method") that stirred the circles of otherwise calm philosophy of science. He critizises modernity for causing barbarian effects and expressing an inhuman love for methods and normative systems (e.g. laws and professions).

> The liberal practice, I repeat, is not just a fact of the history of science. It is both reasonable and absolutely necessary for the growth of knowledge. More specifically, one can show the following : given any rule, however 'fundamental' or 'necessary' for science, there are always circumstances when it is advisable not to ignore the rule, but to adopt its opposite.[62]

Obviously, many of Feyerabends attitudes resound and resonate in the project named after him.

Another approach to dissolve the radical asymmetry that human and computer parts of life have developed into[63] is the ”search for patterns of creativity and composition in software development, music and film” by Flanagin and Holloway. These programmers propose to transfer knowledge about composing of large-scale productions (like Bruckner's symphonies) to the software developing process. “By treating software as composition, and considering that software developers may be composers, we revisit the age-old debate between art and science with a new focus”.[64] The authors study three composers[65] as guiding examples. In Bruckner they find the pattern of creativity as well as the potential for its obstruction through revision. Liszt's revisions were often made out of usability concerns[66], to make them playable by “mere mortals”. In Dvoraks works they see a “free spirit” that learned from mulitiple teachers and divergent models.

Flanagin and Holloway believe that creativity in composing and programming is, above all, a social phenomenon where “people are working cross functionally”. They define 'compositional activity' with this description: “the interrelation of the parts is more essential than the “material” from which any individual part is made”. They conclude

> We can no longer rely on the constancy of our technologies. By creating silos in the workplace, we have abandoned the learning of integration and composition. It is rare to find anyone who has the “big picture”.This strategy of divide and be conquered is contributing to the cost overruns and lowered quality that we have today. Careful mentoring in the compositional and creative processes, combined with an acceptance of the transitory nature of languages, operating systems, and hardware, will create more adept software composers. These composers will then provide better, more satisfying software systems.[67]

Are after all, 'algo-sitions' and 'compo-grams' just a natural way to see products of art and technology as inherently related? Or is there a sound basis for putting compositions on the one side and algorithmic programming on the other side of human reality? Both Bruno Walter and some analysts of algorithms **[1.3]** may think so, but another important field of research is practicing the middle ground for some time already: Artificial Intelligence (AI), the science and engineering of intelligent machines. Generally, AI builds bridges between hitherto man-only abilities and machine abilities. AI often solves man-machine-interface problems. AI can be thought of as the process of formalizing the expertise of human crafts and practices.[68] In the debate between strong and weak AI (see above), the question is whether man-made artifacts will be conscious **[9.5]**. Computational intelligence is today explored in conventional symbolic machines and in neural networks **[ch 2]**, but also in fuzzy logic systems, machine learning, evolutionary computation, and many other paradigms somehow manifested in later chapters.

In the next and last section, we ask whether creativity (and eventually consciousness) can be the product of automatization and eventually a property of machines. Creativity seems necessary to invent algorithms. But can algorithms be themselves creative? And can algorithmic composers be creative? Or is there a non-symmetric relation here? i.e. only creative humans make non-creative machines as their products? We move on.

1.9 Conclusion: what makes algorithms and compositions different?

What is the relation between algorithms and compositions? As we saw above in fig.3, algorithms are functionally equivalent to compositional schemes, not their realizations or instances (composition). Algorithms (A) and compositional schemes (B) both need rules and embody a plan or structure that is the bone or skeleton for compositions *and* programs. In both processes the creator strives for *consistent* systems or plans that do *not* generate (A) or do *minimize* (B) contradictions or inconsistencies. But programming in contrast to compositions, is often (but not always) directed towards solving specific problems. Compositions on the other hand, are instances of composition classes (styles), all logically acceptable instances derivable from the compositional scheme/plan (or better its initial design problem). But there is no difference of kind, only one of degree. Programs and algorithms, are always single representatives of the many other possible solutions as well, and they never really shine like the sun as *the* unique solution. Theoretical computability may help the programmer to find an algorithm (and program) that is tractable **[1.3]**, and one that doesn't fall into combinatorial explosive traps. Programmers try to find *the* most effective solution to a problem; so does the composer, only that in his case will effectiveness be measured in logico-formal *and* sensory-empirical terms (it will have to please or interest the potential listener). Both programmers and composers can "calculate" numbers or symbols, just as serialist logic could easily limit itself to number systems[69] only.
Both follow a temporal logic og progression. The tonal system and the integrated circuits of the microprocessors (on von Neumann machines) are executing in strictly linearly order.
There is though, one important difference between these two activities: Progammed machines may fail and stop completely. Composed works may be played no matter how faulty their rules are employed. Compositions seem to have more in common with connection machines than von Neumann computation. The latter is fragile and inflexible. Therefore programming must be absolutely perfect to avoid breakdowns. In exchange, machine solutions (once debugged and tested) are reliable, repeatable and predictable. And this seems to be the defining difference. Compositions may be faulty, but both performers and listeners are tolerant. They may perceive imbalances and awkward solutions in a composition, but they (or machines in their place) can still execute them. This opens up space for creative freedom, not like the skill of the programmer realized in his search for *the* best solution, but permitting him to do "personal" decisions like a composer.[70]
In programming contexts there is usually a collective frame for use and success, i.e. others will participate in selecting the good and fast solutions. This, in short, is what makes programs part of engineering (compositions are not solving public problems as a rule). Yet, sometimes will hackers and other "independent programmers" create games to their own pleasure, and in those cases programming and composition actually coalesce.

We permit ourselves a small sidestepping and ask what a world could look like with these roles switched, a thought already played with in the entry of this chapter. We might imagine public commissions for musical works that ask for solving stylistic problems in the musical domain. For example contemporanous composers should solve formal problems that Bruckner never managed to eliminate. And such works would be accepted for public use only if they followed logical lines of inner structural expansion and complexities from their earlier established template compositions, all checked and tested before their final launch. Any of these "useful" musical works will have to be both utterly transparent and consistent and will be expected to function as a material continuation of historical precedences in music history.
On the other hand, programmers might get premiered and prized for their funny and interesting experiments applying invented connectionist neural nets and genetic programming toys, created in artistic labs and put in use to entertain the masses with peculiarities and the educated circles with substance for debates[71]. This is not a world we soon will live in or even might want to inhabit, but this scenario straightens out some of the provocative potential of our initial verbal recombinations:

Composers use algorithms to build code that is executed by performers.
Programmers create structures that are played by machines. (repeated from 1.1)

The idea of uniqueness in "human algorithmic composing" stems from the cultural conviction that *man is creative, machine is not, hence machine algorithms are not*. But machines today beat the most excellent humans in chess, financial trading, pattern recognition, certain medical diagnosing etc. Machines certainly play Bach an awful much quicker than any virtuoso might hope for to attain in his own lifetime. But composing *like* Bach, or *like* Mozart? We will see in **ch5** that many styles, including that of 'Bach' and 'Mozart', are pretty much computationally solved already. They pass Turing tests. They stimulate thought and provoke debates **[ch5,ch9]**.
And for all that, there remains the question of creativity. What is its nature? Is it the final criterium for sorting computations and compositions into different piles?

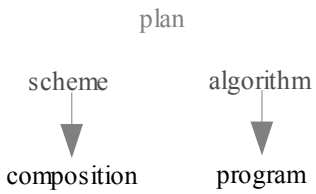


Fig.6: Relation between algorithms and schemes

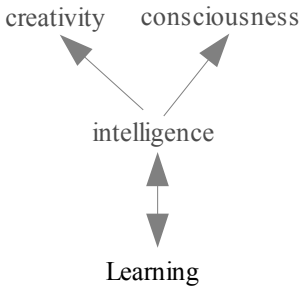


Fig.7: Disputable hierarchy of higher level phenomana of the mind

Leonardo da Vinci is generally viewed as *the* incarnation of that very general and difficult to circumvent-able concept of creativity. Da Vinci is cited[72] as the constructor of a mechanical device that generated canons, possibly the first known Machine Composition system (MCs) in history. What is it, that da Vinci obviously had, but his canon-generator evidently had not?

Definitions of creativity are many and multi-faceted. 'Divergent' ideas or thoughts are a good starting point. But creative acts (in both science and arts) must be both original *and* appropriate, in the sense of satisfying inventiveness as well as taking care of needs or problems. In other terms there must be novelty and usefulness involved. Novelty alone, is a too crude condition, since it would make e.g. the factory minting of coins a creative process. Minting, however factory-like executed, is certainly productive, but not innovative, even though values are produced in such unstoppable machines. Therefore, Margaret Boden defines psychologically creative ideas as those that cannot be made from generative rules that make our other familiar ideas. Studies in the cognitive sciences generalize creativity in a new field of 'Conceptual blending', that makes use of theoretical concepts from linguistics, such as metaphor, analogy[73], structure mapping and geometrical mapping**[2.5]**.[74] Embarrassingly, creativity has a long history of being tied to mental illnesses[75]. It seems like individual autonomy or originality[76] (detached from collective truths and habits) as well as intelligence are necessary but not sufficient conditions for significant expressions of creativity. The informal uses, and sometimes metaphorical misuses, of the concept of 'creativity' in fields like economy,[77] organizational theory a.o. have made many serious researchers avoid this concept in their scientific studies. Nevertheless, research in AI is studying innovation and creative processes in computational terms, and hence often radically deviant terms of understanding. Among the approaches taken are case-based reasoning of symbolic AI, but also new forms of Neural Networks (ANN) and machine learning, and not the least the recent evolutionary paradigms in computing **[ch5,7,8,9]**.
'Artificial Creativity' will have to answer how algorithms must look like to be able to learn and change themselves? This question is often related, but really not substantially connected, to the idea

of "creatively breaking rules". Rule-breaking artists are by many seen as the irreducable counterexamples to 'artificial creativity' and even artificial intelligence as a whole. But during our short excursion to composers in **1.7** we found something that looked a lot more like rule-inventing processes. In fact, whenever Bach and Bruckner etc.[78] implemented their compositional schemes they notoriously stumbled over compositional dilemmas or inconsistencies deriving out of their adopted rule systems.[79] They typically answered to these "indeterminate situatations"[80] by choosing or inventing ad hoc rules, a process that may *look like* breaking rules in a certain perspective. But actually such ad hoc rules inflated rule systems with additional rules and more discriminating algorithms responding more adequately to the exigencies of these masters. In doing so, they built more resistant and compatible integrations of *personal* schemes (ideas) with *collective* rule systems (tonal tradition e.g.). Composers may change their compositional schemes (or ideas) to adapt better to the existing possibilties, thereby avoiding the "breaking of any rules", *but that* would certainly have hindered their "artistic freedom" as well.

A generous and positive view on 'Artificial creativity' lies in pursuing computational and recursively algorithmic tools that *recreate the creative* or a self from non-creative physical stuff. This was the subject of the classic "Gödel, Escher Bach" (GEB). Hofstadter's GEB has been extensively cited in order to "prove" the *sameness* of arts and sciences. This is far from the intentions of Hoftstadter. His famous and mind-boggling examples of recursive realities in mathematics (Gödel), arts (Escher,[81] Bach) and the sciences was meant as a theory *about consciousness* and more accurate in his own words "

> how is it that animate beings can come out of inanimate matter. What is a self, and how can a self come out of stuff that is as selfless as a stone or a puddle?"[82]

GEB is therefore about creativitiy as well, to the extent that consciousness is a related phenomenon of creativity **(fig.7)**.

Roger Penrose, mathematician and physicist believes that the algorithmical nature of machines in AI disqualifies them from exhibiting high-level phenomena of the mind (consciousness). His arguments range from non-algorithmic tiles[83] problems (mathematics), physical quantum gravity (excursions) to Gödel's theorems of limitations of computability **[1.3]**. His underlying feeling is that Turing tests do fail to discern between clever "copies" of intelligence and their biological originals.[84] AI, he argues, is only *mimicking* intelligence, i.e. only "taking care" of the *behavior* of an agent. But does Penrose here overstate his feeling about the resistability of "mathematical intuition" and does his argument rest too much on these beliefs in "deep mathematical truths"?[85] As a matter of fact, the majority of scientists and philosophers does not follow him in his views[86] on AI (Wikipedia:Penrose) and hence the consequences for Machine Composition.

At this point, we have to admit a rather important omission so far. In relation to Knuth's definition of algorithms, compositional schemes fail due to the requirement of '*definiteness*'. Algorithmic steps of composing are often non-definite decisions without unique actions. Ambiguity results, that is taken care of by the composer himself, that "chooses" (or decides on) the right percourse of action. But since subjective choice, i.e. non-deterministic methods are involved, compositional algorithms do *not* meet the *formal criteria* of the *'hard' algorithms* (finiteness/tractable; definitness; input, output, effectiveness/same output for same input). Therefore we may classify those composing algorithms as algorithm-like or '*soft algorithms'* (thereby distinguishing them from Knuth's supposed 'hard algorithms' for machines). But, Knuth mentions aesthetic criteria as well, such as simplicity, elegance and parisimonity (getting straight to the point). These are certainly required of artistic algorithmic activities as well.

Differences between algorithms/programs and schemes/compositions

Algorithmic	***Schemes***
Local logic (step by step)	*Global* logic (generative rules, constraints)
Search for (a) *best* solution	Search for an *excellent* solution
Deterministic methods (objective choices) (Knuth's requirement of definiteness)	Local choices non-definite or non-deterministic, but not random or subjective (or miraculous either). Deterministic methods for global structure
Adapted to *convergent* and truth-based application (e.g. science)	Adapted to *divergent* and coherence-based application (e.g. arts)
Hard or technological definition of algorithm	*Soft or cultural definition of algorithm*

We might by now presume that our examples of algorithms and compositional schemes (and their compositions) illustrate *more likenesses* than differences between machine programs and music compositions. Therefore we may proceed with a “lightened heart” and open eyes to converging phenomena. In evolutionary theories and terms we look for earth-bound answers to such investigations. Algorithms in machines as well as schemes or styles are all subjected to external tests for their usefulness and fitness. Where does music, musicality, musical intelligence and creativity itself arise? And is there an algorithmic part to play in this unfolding drama?

What I have tried to do in this first chapter is to unrest and loosen some of our very cherished beliefs to prepare the ground for a restructuring of this conceptual garden or territory. Do we thereby set the background for composing machines and install with Machine Composition a license to create (as well as work)?
Now that we have established the roots of algorithms in mathematics and engineering as well as having demonstrated their relevance to music and the arts in general, and that we have detected common sources of the creative nature of all these activities, we may continue with an account of the natural origin of humans, their musicality, music and last but certain not least their creativity in the following chapter.[87]

Chapter 2
Processes and Objects:
Musical intelligence and representation
Naturalizing musicality and natural tools for constructing musical worlds

1. Evolutionary accounts of musicality and musicianship
2. Music naturalized? Consequences for musicology (cognitive and behavioral models)
3. Representation: Computational models
4. Levels of musical representation
5. Problems and prospects of discretization and formalisms in music
6. Conclusion: Ontic engineering of musical ojects, processes and worlds

The idea, that all the fruits of evolution can be explained as the
products of an algorithmic process is Darwin's dangerous idea. (Daniel Dennett, Dangerous Idea,1995[88])

This chapter is about the sources and conditions of musicality and music. An evolutionary theory of the natural *processes* leading to the phenomena around music and specifically the composing by humans will hopefully contribute to a better understanding of *representational* issues in music and composing.

2.1 Evolutionary accounts of musicality and music

> *What Darwin discovered was not really* one *algorithm but, rather, a large class of related algorithms that he had no clear way to distinguish. We can now reformulate his fundamental idea as follows: Life on Earth has been generated over billions of years in a single branching tree -the tree of Life- by one algorithmic process or another (Dennett,1995[89])*

The sound making of birds has been an object of fascination in many cultures,[90] if not humanity at large, and has often been compared to both speech and music. Earlier beliefs focused on the invariance and therefore used hereditary explanations of "bird songs". Nowadays, most research is more concerned with the invariances[91] and studies learnability in bird singing and bird calling.[92] Some species' acquire repertoires many human singers might dream of, e.g. the 3000 distinct songs of the Brown Thrasher[93], and they apparently learn them by practicing, involving feedback from their teachers and error-correction. Earlier it was believed that birds essentially picked up elements from their parents only, but newer evidence shows inclusions of "ideas" by neighboring birds as well. This allows for still greater variation in individuals and opens for cultural aspects and interpretations. Additionally, recombinability, of in themselves meaningless acoustic elements, (phonocoding[94]) is found in bird's songs as well as in the music by humans. But the line between communicating calls (e.g. giving tongue to danger, or "here I am, a strong male looking for a mate") and non-communicating bird songs (mostly meaning not much more than "here I am, and this is my place") is thin and knowledge about the meanings of bird songs far from complete.

Notwithstanding the obvious discontinuities between animal communication systems through auditory means and human language and music cultures, there cannot be denied a certain familiar basis to all of these activities. Stephen Wolfram finds it plausable that:

> Famous motifs from human music are heard in bird songs probably more often than would be expected by chance. It may be that some common neural mechanism makes the motifs seem pleasing to both birds and humans.[95]

Mithen cites musicologist (and composer) François Bernard Mâche that finds many direct parallels:

> "Other birds not only enumerate the tones of their scales, but also build 'melodic motives as elaborate as many human achievements and even sounding so close to them that one might be mistaken' (Mâche, 2000,477). More generally, bird song includes many of the processes of repetition that are so critical to human music, such as refrains, rhymes, symmetry and reprises.[96]

When it comes to explaining the origin and functionality of bird's singing, Darwins[97] theory of birds "courtship display" as part of sexual selection principles is still widely accepted today.[98] Evidence for it is rather compelling, since only males are singing, and especially actively in mating periods.[99] Another known function is the expressing of territorial claims and we don't know whether other functions will be added to this list in the future (e.g. the pleasure to express or produce in aesthetically ways).

Interestingly, the list of other species that display music-like behavior has recently become increasingly longer. Dolphins, whales, bats, mice and gibbons, as well as insects[100], are all known examples of animal specieses that researchers study in this direction.[101] Male whales during breeding season e.g. produce long lasting songs in a choir with other "group mates" of astonishing complexities. The songs in different groups of whales are similar in structure but often very different in content. Research by Katharina Payne points again to certain likenesses with human musical practices like improvisation, summed up by Mithen, 2005:

> Each individual whale is constantly altering its song by modifying the phrases and themes it uses, and the manner in which these are combined. Nevertheless, all whales within a single group appear to agree as to which themes are stable and which are changing. No single whale appears to be driving this change; each seems to change its songs about as much as any other whale in the population. But listening and learning must be essential to the evolution of their songs; Payne suggests the process is best understood as being analogous to improvisation in human music. It certainly occurs more quickly than change in human language; whithin one decade a population 's song may have undergone so much change that one can no longer recognize its relation to the earlier version.[102]

A recent discovery by Holy[103] shows that mice males sing in response to pheromones, a scent hormone linked to mating processses. Whether their singing, that occurs two octaves above human hearable range, is contributing to fitness seems probable, but is still unproven. Holy is ranking mice just below birds and whales with regards to their songs' diversity of pattern and rhythm.

In perspective, does this indicate an evolutionary line between the singing birds (or insects) to the singing of the "Pavarottis" of our own species? The animal communication expert Marc Hauser doesn't think so: he bases his opinion on the fact that our nearest relatives, the great apes, do not exhibit singing or music behavior[104]. He thinks therefore that convergent evolutionary trends shaped the vocal and neural functionalities allowing for learnability and recombinability (or phonocoding).[105] This makes the questions as to why and how musical abilities have evolved more of a mystery, not less.[106]. If chimpanzees and other ape species' do *not* have rhythmic capabilities on the level with humans,[107] the question that Peter Gärdenfors superscribes his book-section about the origin of music, "why chimps do not play in the circus orchestra",[108] will have to hold our breath for some moments. In other words, why are humans singing and clapping while chimps (presumably[109]) cannot. Gärdenfors believes the explanation lies in the human practice of rituals, and coordinated and cooperated work activities; activities we know well from so-called ethnic human cultures today.[110] A peculiar theory, proposed by Björn Merker, tries to explain the fact that some chimpanzee cultures are female-exogenetical (females moving out to other groups) and that therefore males in the competing "tribes" are required to promote their fitness by hooting in synchrony[111] to reach further out. Such "power display" may attract 'wandering females' from longer distances. Gärdenfors (2003) continues this line of argument and mentions the fact that hominids

lived in savannah terrain, making them more visible (compared to forest-inhabiting apes). A consequence of this higher visibility would make “hominid dancing parties”[112] effective in attracting females from other groups:

> If a group made a big kill, they could spread the message about this by shouting in time, but also by moving rhythmically. Dancing goes together with singing. The combination of plenty of food and singing and dancing is an efficient way of making a band of hominids attractive, in particular for young females. Apparently, the party has an evolutionary value. In all societies dance, music, and song are important for meeting a partner. Nobody becomes an idol with the young because you can speak or walk, but because you can sing or dance.[113]

Gärdenfors discusses the implications of hypothetical male-exogeny as well, but his theory remains in all cases inside the paradigm of competition and power models; that is in full accordance with Darwins selection of the “strongest” singer regarding a “musical evolution”.

Another explanation of music and its origins has been put forth by evolutionary linguist Steve Pinker.[114] He describes his theory of music with the term 'auditory cheesecake', to contrast the status of the *adaptive* (therefore evolutionary necessary and important) *language* with the position of *music* as mere evolutionary *off-spin* or technology for entertainment. Pinker thinks that language or proto-versions of it aloud enhanced prosodic forms of meaningless sounds become a derivative phenomonon that we like to call 'music'. According to Pinker, music does not have any central place in the mind (in the evolutionary sense of survival fitness); Pinker thinks it is derivative from other evolved propensities or invented from these for entertaining use only. Pinker writes (cited by Mithen) that “as far as biological cause and effect are concerned, music is useless... music is quite different from language... it is a technology, not an adaptation”.[115]

Prehistorian and archeologist Steven Mithen presents us with a very different explanation in “The Singing Neanderthal” (2005).[116] Mithens position is the antipode to Pinker's 'auditory cheesecake' theory. Pinker's view of music has provoked many musicologists to rebut his “devaluation” of the role of music in human culture. It could, in fact, seem, like a battle between language and music theoreticians about the master role in the evolutionary play. Mithen[117] presents the hitherto most serious defense on behalf of music, where music has its own and autonomous evolutionary road, and thus turns Pinker's order of sequence quasi around, when he “vindicates” the “nonverbal, prelinguistic, “musical” mode of thought and action”.[118]

According to Mithen, music and language share a common ancestor in a “musilanguage” or 'Hmmmmm' communication. Mithen sums up:

> Music emerged from the remnants of 'Hmmmmm' [communication] after language evolved. Compositional, referential language took over the role of information exchange so completely that 'Hmmmmm' became a communication system almost entirely concerned with the expression of emotion and the forging of group identities, tasks at which language is relatively ineffective. Indeed, having been relieved of the need to transmit and manipulate information, 'Hmmmmm' could specialize in these roles and was free to evolve into the communication system that we now call music.[119]

The 'Hmmmmm' stands for **H**olistic, **m**ulti-**m**odal, **m**anipulative, **m**usical and **m**imetic. This type of communication, Mithen thinks, is functionally tied to the transition to bipedalism and shaped by selective pressures for enhanced communication coming from foraging, mate competition, parenthood and group activity.[120] It is Homo ergaster[121] that in this version of genesis “invents” a new communication, unknown in apes and other early hominids. 'Hmmmmm' is

- holistic because their utterances are non-specific,
- multi-modal because several modes are used (e.g. body language, facial expressions, gestures, sounds),
- functioning *manipulative*, such as requesting, appeasing, greeting, warning,
- musical in using songs to attract females,[122]
- mimetic in e.g. holding one's heart is thought to come from imitating grieving members of the group, and eventually becoming generalized as mimicking in other similar situations.[123]

'Hmmmmm' communication was emotional as well as practical for group adhesion and trust, and it contributed therefore directly to survival.[124] But, in addition, it had other advantages: singing and dancing together was creating a group awareness and attenuating the tendencies of self-interested behavior, enabling cooperative contexts with a certain stability.[125] At about 500.000 years, Homo ergaster split in two major directions: the Homo neanderthalensis in Europe and the african Homo lineage of homo Heidelbergensis. Mithen's theory states that Neanderthalensis developed the 'Hmmmmm' further into an advanced version, while Heidelbergensis 200.000 years ago introduced *segmentation*[126] of the holistic phrases into language and specialized the emotive part of 'Hmmmmm' into music, how we essentially know it today. The new language communication was now changing from holistic and manipulative to a symbolic mode of compositionality and referentiality, like we essentially know from linguistics today (Chomsky grammars, lexical referring). This change happened while Neanderthalensis developed 'Hmmmmm' into more advanced, but still fundamentally holistic and manipulate forms of communication. The reason for an instrumental and flexible language occurrence may have been adverse conditions in the ice ages, since language permitted more effective adaptation and hence survival.[127] These different scenarios for Neanderthalensis and Sapiens may have been accompanied by different models of mind. Neanderthalensis' intelligence or mentality was mainly *domain-specific,*[128] while Sapiens was *cognitively fluid*[129] and domain-transgressing. Such a quality of transgressivity may be another way of speaking about instrumentalizing culture and creativity[130] (see below 'conceptual spaces' of Gärdenfors). In other words, Neanderthalensis conserved and developed social culture and cooperation, while Sapiens went instrumental and individualizing.[131] As Neanderthalensis sang and danced to make “altruistic”[132] culture thriving, Sapiens built progressing civilisations in incremental processes (agriculture, pyramids, tools etc.).

The implications of this theory are wide ranging and we can unfortunately only scratch the surface. Just some indications of the arguments for Mithen's theory. Mithen looks at the relationship between language and music, at studies about aphasia, amusia, musical savants and brain damaged patients. He deducts a modular brain model of music processing.[133] This model is neutral relative to its input (acoustic), and outputs singing, tapping and speaking according to the flow of information through the modules (partly musical, partly linguistic). The fascinating consequence of this model is that many of the recorded dysfunctional states[134] are explained through this one and only model. For example, Mithen hypothesizes that absolute pitch hearing[135] is something all humans have at birth (as certainly had Neanderthals as well). With enculturing into discrete “linguistic terms”, this adaptive advantage in Neanderthals becomes then an interference with the exigencies of language learning. Hence, this capacity in human children is suppressed and vanishing, if not otherwise preserved through unusual musical training in early ages.[136] Is it possible then, that humans could choose to become less “logical” and more emotive through developing our innate 'Hmmmmm'-ness? Another very convincing argument derives from the mother-baby talk (IDS[137]). Studies show that IDS is universal[138] (independent of languages) and is caracterized by hyperarticulation of vowels, gestures and generally more music-like utterances.[139]
Other evidence used to build a case for the 'Hmmmmm' theory are drawn from 'music therapy', laughter[140], medicine, psychological illnesses, “happyness hormones” (oxytocine), and the so-called 'Mozart effect'.[141] All in all, the breadth and depth of arguments, deriving from a plenty of sources from paleoarcheology, neurology, psychology, medicine, biology, genetics, musicology to mention only a few of them, is pretty much astonishing.

Where does this leave us with respect to our understanding of music, musicality, and musical representation? We have first to choose between two types of theories about the origins of music: the Darwinian[142] and the Mithenian. The Darwinian stresses competition (between individuals), and rational, instrumental language as major adaptational vehicle for survival. Mithen on the other hand

emphasizes cooperation (of groups), emotions and holism. He assigns musical communication an adaptational and necessary role in survival in all species' except Sapiens who reverses the balance between music and language. If Mithen is right, much of our musicality is still in ourselves. It opens debate about gender perceptions (masculinity and emotions) as well as the future of of our species'. A simple argument might point at the Neanderthalensis demise to prove Sapiens' superior essence. But, if we look at our history in perspective (*Homonoid phylogeny figure*[143]), the history of Homo Sapiens is miniscule compared to our predecessors. Therefore, could it be that we are a dead end only, due to a debilitated emotional architecture?[144] We cannot follow this thread trough here.

We recapitulate the evolutionary accounts of musicality in informal terms :

Darwin: "here I am ♂, a strong male looking for a mate "

Gärdenfors: "here we are, we are a strong group of able people that you may become a member of ☻ ☺ ☻"

Pinker: "here I am, and how do you like it, by the way? © ♥"

Mithen: "you and me[145] ☼, and actually the others as well, belong to each other, we are the same and pull in the same directions as a 'we' ♥☼☺"

In the next section, we will look at some of the possible effects and consequences a Mithenian view on music and musicality may have, before we go one step further and look, very much in "Sapiens-manner" at 'representation' and 'abstractions' that will enable machines to do what Neanderthals did to their higher positioned larynx, resulting in the tenor-voiced[146] emotional songs.

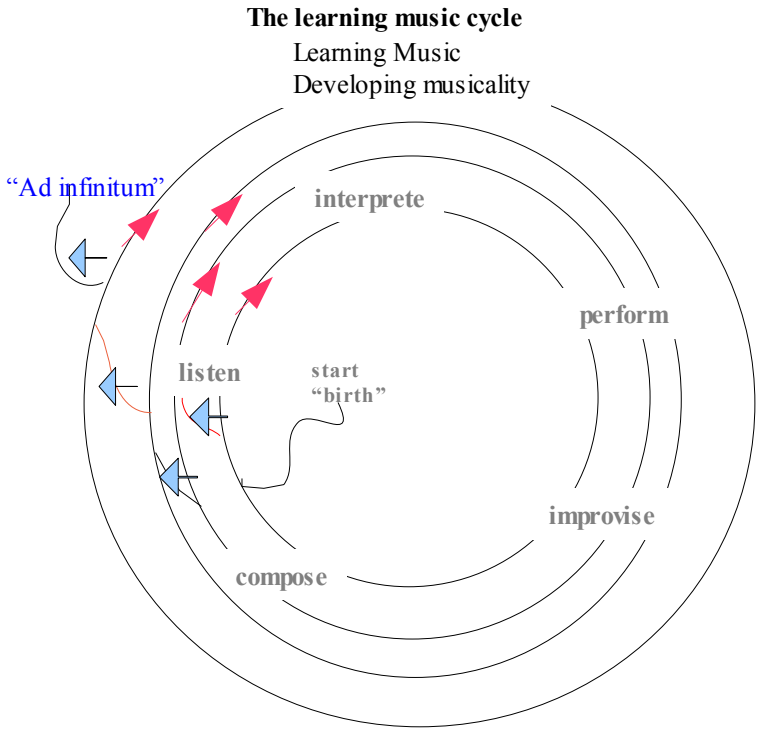
2.2 Music naturalized? Consequences for musicology and music psychology?

We have visited Darwinian, Gärdenforsian, Pinkerian and Mithenian accounts of the origin of music. Whatever the scientific consensus will be, we have to do with evolutionary sound theories that place music in a continuity[147] with other neuronal and anatomical functions and developments, specifically in the context of communicational tools. Music is then the outcome of a series of reproductive variations selected by environmental pressures. Music is the effect of gene recombinations relatively benefitting either competitional (Darwin, Gärdenfors) or cooperational strategies in evolution. The major difference herein lies in the weight and importance[148] giving to the social and emotive functions of music in prehistoric "societies". In the following I choose to follow Mithen's theory and its consequences for musicology and as well as musical practice. The more social[149] its function is supposed to be, the more importance should be given to the learning process in music, and specially the *social* aspects in learning.

I like now to propose a *cycle of musical learning* that is emphasizing the dynamical and social aspects:

A newborn begins her life by hearing sounds in general and "noises"of music in particular and will respond to them. The baby is immediately interpreting sounds in some "Kantian"[150] categories and eventually responding to them in ways that will seem gratifying to her mother and others around her (also IDS above). This is eventually making the child confident and leads to variational responses (improvising) on the established materials. Growing mature, "composing" (in the sense of proto-composing really) would mean that the baby/infant sometimes uses longer time before expressing itself ("proto-reflecting?) and that it responds in more intentional ways than by mere improvising or repeating (mimicking[151]). Finally the infant is awarded or "frustrated" by others around her acting as feedback agents.[152]
In older children these modular learning stages will have grown more crystalline and ultimately stiffen and solidified, in Western musical cultures, into rather different activities of kind.

While older cultures still leave these aspects or stages in musicality intact in fluid and intertwined wholes[153], much of western music teaching is fairly obsessed with specializing and building "walls" between these stages. The fairly strict treatment of adult Anton Bruckner during counterpoint instruction[154] stands in very much contrast to the disciplined, but auditory rich method of notation-free Suzuki learning classes[155] in Japan today. As musicality in the child evolves, the assumed transitions between the theoretical stages in learning are actually rather plastic.



The difference between them is gradual and manifests itself collectively in a continuum of adapting, correcting and expressing tendencies, that we may look into systematically:

- **Listening:**
  Listening is the active activity of *making sense*[156] of input. It makes sense out of structure, using discriminating and generalizing methods. Composition is the opposite, namely the *making structure sensible*. These processes are interwined: the composer uses itself and listeners to test appropriateness, while listeners use compositions to test and develop their listening abilities.
- **Interpreting**
  Interpretation is a more active and "manipulating" process. It is more individualized and constructive than the rather phenomenologically induced listening. It may be enhanced with parallel or analog structures from other domains (body movements, pictures, synesthesia etc.) and alouds choices that may even be presented publicly. On the other side, we see easily how interpreting is the necessary preparation or input to performing practice. Performing is essentially the expression of an active regrouping of features and nuances during prior interpreting at various degrees of deliberate and conscious activity. 'Interpreting' may be viewed as proto-activity for any cultural process, declared by hermeneutics, existentialisms, phenomenologies etc.
- **Performing**
  Performing is the body-and-mind involving integration of understanding and expressing. It is the coordination of intentional interpretation with muscular responses that is used for "manipulative" ('Hmmmmm' communication) purposes. It can be any place between the quasi-passive repetition after a model (like a cousin of a Neanderthal, guru or teacher) and the quasi-active formation of new formulaic variations "invented" during improvisation. Performing does enliven structures.
- **Improvisation**
  Improvisation is the adaptive and creative process where variations and alternatives are intuitively planned and actively proposed during a "real-time composition". We may see improvisation as primarily group directed activity, where several improvising members find a balance between foreground-background alternations and novelty-familiarity proposals of materials and variations.

Still, a single singer/player may quite similar respond to his own output in such adaptive and creative ways. While performing does very much the same as improvisation at the tone-level of sound (see micro-sound below, **2.3**), improvisation works on higher levels of melody and harmony.

- **Com-position**
  Composition, at last, is the highest-level activity of a planned and creative process that not only recombines single notes, but whole structures in carefully designed[157] ways, deliberately chosen and tested by improvisational (see generate-and-test, in **ch1** and **ch7**) means and selected by refined levels of listening at varying levels of conscious control. Listening and improvisation therefore encircle composing activities, very much like all the other stages/activities in this learning model must be seen in context with their neigboring stages/activities. Arranging is between improvisation and composition and is defined as scoring and choosing instrumentation and sometimes harmonization as well.

This proposed cycle has several advantages. It both includes and is compatible with the Mithenian theory, and thus somehow ameliorates assumed contrasts between Sapiens and Neanderthals[158], and in the same time avoids the many artificialities that have arisen in modern musical practice and theory (musicology). We may remind ourselves of the obvious musicality-friendly upbringing of many historical composers, thought of as geniuses[159] today. Among the many examples, Mozart's is the most folkloric and exploited to this end in current folk psychology. But in the light of the Mithenian theory of musicality and the learning cycle, Mozart's existence is no mystery anymore. His very early starting music-learning cycle was intensely specialized, caused by his father's wish to make him a foreground musician. We now may say, that his behavioral idiosyncracies are in full line with 'Hmmmmm' communicating Neanderthals, and his natural instruction of every aspect in the cycle of learning was most effective and as well as spectacular: he had high abilities as listener, interpreter, performer, improviser and finally composer. What Mozart lacked presumably was the strategic and cynic power of others (Salieri e.g.) to survive in the intricate social hierarchies of modern Sapiens. This might be somehow speculating, but in my opinion is this strongly supported by many other examples of musicianship in earlier centuries. The fact that todays music learning is heavily compartimented and isolated into stages and orders of teaching (Bruckner, **[ch1]**) may explain the degree of specializing diversity of expressions of different types of "musicality", such as "I am a violinist", or "I am a baroque trumpetist" or "I am a composer of computer music" etc.

When looking at musical learning today we may call the activities in other terms than above: listening becomes analysis or sub-conscious digesting, interpreting becomes musicological theories, performing becomes virtuoso display and perfection (repetiton/mimesis[160]), improvisation becomes experimental studies of style, and finally composition becomes intellectual exercises in civilisational conversation with many other knowledge fields like mathematics, sociology etc. It is no wonder then, that these stages of professional music life (originally derived from aspects of the holistic learning cycle) have been split up into separate teaching subjects in music education.[161]

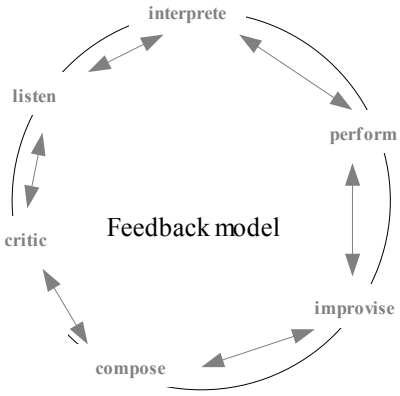
Basically we may now restate the cycle as the integrated and intertwining activities that are to varying degrees active and spontanous. We see for example that improvising is the real-time composing, listening the real-time interpreting, performing the real-time improvisation. The basic generative processes may be giving some more time or less, they are still recombinative and constructive all the same. The by far "longest form of listening" is composing, a far-out corollary of this theory (the inverse of it: "the shortest expression of composing" lies in listening). This analysis of listening makes it a thoroughly dynamic view on communication (bottom up).[162] Let us therefore expand the model a little, before presenting some explicating definitions of fundamental notions:

Question: what will happens when external parts "interface" with our until now harmonious learning context? A typical case is an authorative critic in a newspaper, or a public (on loci musicæ). They are no longer participating, but can be described as professional observers (critic) or listening publics (audience). This idea would probably seem estrangious to a Neandertal group engaging in

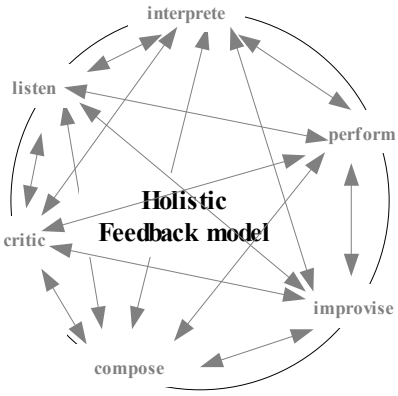
advanced 'Hmmmmm', but is today most natural in our musical practices and cultures.[163] Some music critics today still somehow rest entangled in the manners (and status) of earlier days' of Hanslickean tradition where a strict and important "superlistener" feels that his mission lies in telling the crowd, using flowering and cliché-laden praising-and-blaming-lingua, what to think of (and may be even how to listen to) their concert experiences.

More generally, we may admit, that the transitions between the aspects/stages of the cycle are actually feedback loops of different shades (but of equal kind). Therefore we update our model to version *II*, that reflects the increasing specialization of Sapiens' musical culture. Cycle *II'* , on the other hand is based on the more Neandertal-like communication (advancded 'Hmmmmm'). It describes to some degree older cultures of today as well. This holistic cycle (*II'*) is the *non-instructional* version, and hence *adaptional* type of learning that was already mentioned in various places above.

The learning music cycle II



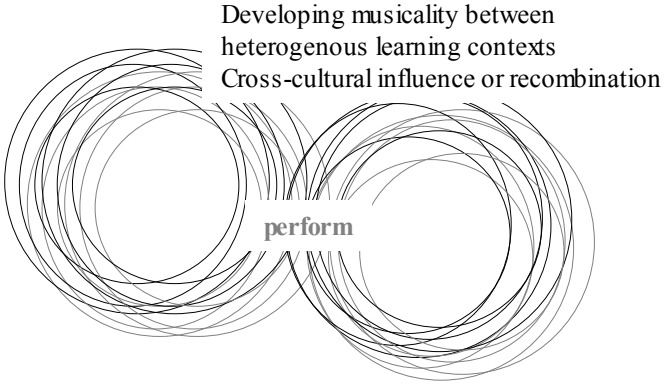
The learning music cycle II'



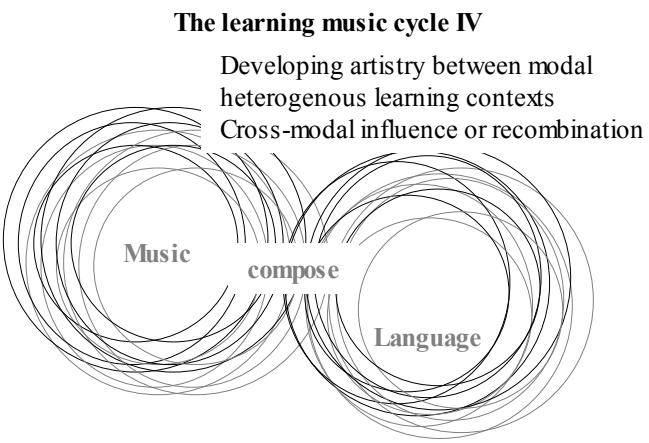
What happens when differently molded parties, i.e. coming from diffent learning environments or learning cycles, take part in common activities? This applies really to all of the above described aspects of the cycle: two performers or interpreters from different teachers, two improvisers from different styles, and two composers of different "genres" cooperating on a sonic project.

In model *III* we see an illustration of what might happen when two different traditions, coming from teachers, styles, genres etc. engage in common musical activities. This will lead to an exchange of musical material or possible new recombinations, leading eventually to new forms of results. An example is our conception of 'World music', where different music cultures are friendly finding new contexts for interaction. Other examples are ensembles cooperating having little or no prior history in common. When pianist Andsnes for example chooses new chamber musicians for new musical projects (cd, concert tours), they will have to cycle around their frictioning

The learning music cycle III



intersection points until sufficient adaptations have taken place.



A last example of divergent learning cycles (*IV*) is made of even more different qualities or modalities. Operas for instance, are art works where different learning arenas participate in a common of communication, believed to be effective. The same argument is valid for singing ensembles, choirs and songs. In an opera, language-expressions may be integrated with concerted body action (ballet) and 3D-model rendering of a virtual world behind the curtains, accompanied by the sound-painting orchestra that underlines moods and emotions. In such multiple cross-modal learning cycles, many forms of human and artistic experience are entangled into one unitary 'Gesamtkunstwerk' (Wagner's opera circles are proto-typical illustrations.)

With this framwork of dynamic understanding of music, I propose definitions for the following related terms:

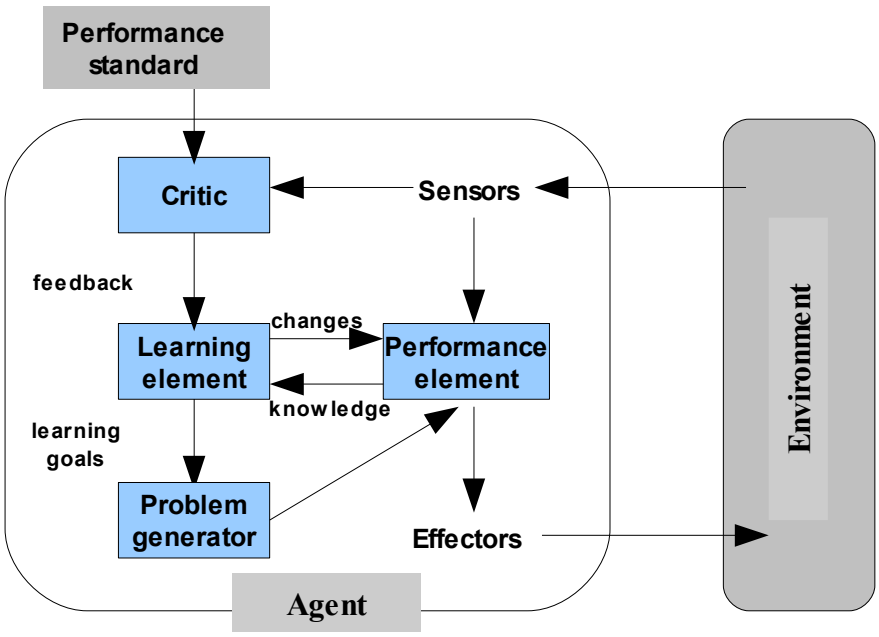
- **Musicality**: the inborn and universal capacity (even for absolute hearing according to Mithen a.o.) in hominids from Heidelbergensis (evt. Ergaster) and above able to participate in musical learning cycles that include mimicking, imitation, variation and contrast leading to ever higher levels of reciprocal adaption.
- **Musicianship:** the degree of attained and sustained level of qualified musical experience in relation to music learning circles and cycles. A cycle consists of sustained and repeating social and interactive learning processes that results in an extended musical repertoire of *flexible* abilities.
- **Musical sector proficiency:** the competence in certain domains or the acquisition of *specific* musical skills that is trained in mostly *instructional* modes and therefore derived from *partial circles* only. Such proficiency or skillfulness is assumed to make only partial use of the cognitive or neuronal model of music processing[164] (Examples[165]).
- **Music:** to define music 'in generis' remains a hard task. Bruno Nettl[166] is particularly non-comitting in his “human sound communication outside the scope of language”-definition. We try a little harder and propose the following in the light of the anterior thoughts: Music are the activities where combinations and recombinations come out of consistent and integrative learning cycles. Should we say something about the aesthetic nature? I think it is implied in our learning cycle idea. The cycle is inherently *social*, *emotive*, *creative* and *hence artful or aesthetic*.
- **Musical intelligence:** Intelligence is in general tied to the capacity to solve new problems. A typical test of intelligence contains sequence or pattern prediction[167], e.g. expecting 15 after occurring 1..3..6..9..12....[168] One of the methods that may be used is statistical learning, that seems to be the source of both music and language learning in very small infants.[169] We may therefore speak of the instrumental capacity to manipulate others by recombining sounds, learned in musical learning cycles. Even without compartimentalization of

competence (see sector proficiency) we may better speak of individual and collective forms of musical intelligence**s** to avoid problems usually encountered in definitional projects for this term.

With these conceptual tools we may look at the *state-of-art model for machine learning*.(AIMA[170]) Surprisingly, there are quite som overlapping areas and modules. Here is a tentative overview:

Musical learning cycle	***Machine learning model***	***Comments***
Musical motivation or creative drive + inborn limitations	**Performance standard**	*This is the critic's background goals for the whole activity*
Listening	**Sensors**	*Information input, ears/brain*
Interpreting	**Performance element, Critic**	*Matching patterns, understanding input*
Performing	**Learning element, Performance element, Effectors**	*Execution of Model of correct patterns and practice through information output, brains/muscles*
Improvising	**Problem Generator**	*Informed by learning element (learning goals) to generate new "variations over the themes" in performance element*
Composing	**Performance element**	*Model of correct patterns and practice; has to be updated by learning element*
Internal Critic	**Critic, Learning element**	*Analyzes input and output to make learning element*
Other co-learning individuals, evt. External critic (public e.g), other musical stimuli from the outside of the learning cycle	**Environment**	*Environments are vital for learning and quality. e.g. a non-discriminating and little-inspirable public may not help artists much in learning and developing*

Any of these learning modules can be represented by any represention system of AI. All seven components can be described by quantifiable functions in a computational model, where the goal of the whole learning system can be seen as learning the representation of a specific function[171]



The performance element can for example be represented as a decision tree with yes-no outputs for every node or represented by any other AI and machine learning representation scheme.[172] As we saw in **1.3**., there is a trade-off between efficiency and expressiveness. The more reductionist our representation of atoms, the more heuristical must the methods be to compensate and assure tractability. This is in analogy to the musical cycles incremental learning exigencies. If too difficult pattern problems and resolutions of representation, the learning elements will not be able to find the "solution". On the other hand this implicates background and prior knowledge in the learning cycle, translated to incremental learning. It is also a good way to assure the success for a machine, i.e. the distance between existing knowledge in performance and learning elements (+ critic) must be sufficient to allow a solution to be found as an NP-complete computation.

Composition projects will be enhanced by *speedup learning* that exploit analogous situations for effectiveness.[173] The available type of feedback from the environment is also relevant. In improvising, and helpful interaction, other musicians will indicate the solution in a qualified way (e.g. play the right line). This is called *supervised learning*. In *reinforcement learning*, publics will just grunt or nod(*no!/yes!!*) to give away dissent or consent that is exploited to change course of action in both cycle and model. The hardest learning environment would be a public that doesn't show any sign of response, such that the machine or musician cannot get information about the appropriateness of its musical actions for possible change.[174]

A very much oversimplified example of choir singing: there is only one goal, namely be in tune with the others. Singing in a choir is very much like using the learning agents circuitry. It means continually monitoring the environment to fit in as much as possible. Such a *tuning process*, where one uses a positive feedback to adapt to, is a very straigtforward learning example. Eventually, one could have an expectancy module as well, to learn which pitch is coming next, but this is actually done very elegantly by notational sheets, making it easier for the choir to be in line and tune with the rest of these singing Neanderthals.[175]

We might conclude this short comparison with the observation of the importance of the *environment*. The quality or expertise of both co-players or external critic (public!) is very much determining quality of the learning process as well. As Mithen observes correctly, where would Mozart or Beethoven have been without their demanding and expertly discriminating publics?[176] As we see, this applies equally to machine learning.

We have learned that music is a dynamic and socially learned activity. And we may at least conclude that machine learning and learning cycles are not incompatible. Next step is to discuss the representational possibilities for machine composition, both in computational **[2.3]** and musical terms.

2.3 Representation: Computational models (algorithms need data!)

What is representation? It is the symbolic method of treating one thing (A) as something else (B).[177] This is what Sapiens, according to Mithen, specializes in language communication, namely effective referentiality.[178] We may say that Homo Sapiens invented 200.000 years ago the first symbolic machine, namely language.[179] Language *produces* multiple and infinite expressions from a few atomic symbolic words and sounds. This feature of language is 'compositionality' (generative principle). In **ch1** we established common ground for com-posing and com-puting (algorithms and schemes). But algorithms need data, just like minds need perceptions. Without data or perceptions, computers/alorithms and minds are empty and powerless. What are data? Data is "stuff" that represents objects we want to manipulate.[180] 'Data' in computation are numbers that represent words and objects in the world.[181] How are 'data' implemented in physical machines? They are as digital

bits of ones and zeros electronically “put” into memory locations or registers of von Neumann machines. Almost all machines today follow this 'von Neumann architecture' or Turing-machine-model. Programming languages will use representing *primitive expressions* to build compound expressions by *means of combination* and manipulated further using *means of abstractions.*[182] Procedures and algorithms are descriptions of the rules for manipulating the data (expressions of various levels of abstraction). In **1.4**, Sudoku problems were “solved” by a representation of primitives (given, s, etc) and their manipulation through the proposed procedural steps of an algorithm. More difficult problems need levels of abstraction (building “logical walls” or 'abstraction barriers',[183] and hierarchic 'data structures', like trees and forests presented in **1.3**.)

The choice of data (representing the real thing) and the representation *of* data in computing devices and programming languages is determining the success and appropriateness of a solution. E.g. if we need to solve a difficult problem and decide to use imaginative numbers as primitive expressions, we may end up with memory related intractability. When we choose to represent a human agent based on physical atoms, we will probably end up with combinatorial explosive results, while a representation on the intentional level[184] can project a “comfortable” machine model that makes problems solvable.

But there are also different ways to represent the abstract data (representing the real thing, “stuff”, information) on different machine architectures. In addition to v.Neumann, we can choose connection machines. These are modeled, non-symbolically, as networks. Such networks are “copying”[185] the biological neural networks functionality of brains and neurons. Brains are representing “stuff” distributed over many *connected* neurons (networks) and respond to “holistic” input. 'Connection machines' or artificial neural networks (ANN) do computational processing in similar (but certainly far from imitating!) ways. This has consequences for representational issues. Neither primitive expressions (words e.g), nor compound expressions (sentences, melodies) are traceable, i.e. located or anchored precisely in the brain. Even though, certain mental functions are found to be seated in particular areas or regions of the brain (“language in the left part” e.g.), much of this knowledge remains unresolved.[186] In general, neural networks are distributed systems, opaque to our understanding, and this applies equally to their technological off-spin (ANN's). Why should one use such non-representional machines at all. ANNs have these advantages (after AIMA):

- Networks work in parallel; (v.N.m. Process sequentially in Turing-like fashion)
- Generalizing and complex tasks may be possible; (v.N.m: intractability problems;**1.3**)
- Fault-tolerant or “graceful degrading” performance; (v.N.m : “break downs, loops; **1.3**)
- Programming by examples (learning) or “automated programming”, i.e. ANNs are trained in various ways, delegating the actual “programming” to the adapting net itself. (v.N.m.: see learning model in **2.2**, with symbolic paradigm)

The reason that neural computing has not replaced symbolic computing are the problems and difficulties of control (updating of multiple weights) and transparence (contribution of hidden units). The mathematical descriptions of ANNs are rather complex and non-intuitive relative to the symbolic languages inheriting from logic. Still, in many areas of pattern-based computing, like perceptions (including sound), ANNs have proven useful and superior computing systems. The consensus in AI is caracterized by a pragmatic attitude regarding the use of these paradigms.[187]

Summing up, there are established representation paradigms for logically specified domains (symbolic machines) and mathematically derived mechanisms for complex input-output functions (connection machines). Applications in the cognitive sciences depend on representations of concepts, concept formation and conceptual change processes. In short, these have been insufficiently handled by symbolic means and are difficult to upscale by connectionist means. Conceptual phenomena, such as metaphorical meanings are notoriously difficult to model in conventional paradigms. Examples of successful applications based on symbolic computations are

expert systems, where the knowledge domain is as solid as metaphors are fluid.[188] Examples of successful connectionist applications are perceptual discrimination problems of very limited problem size.

With this background Peter Gärdenfors[189] looks after the middle ground between these two forms of representation, calling it the conceptual representation. It is nearly as transparent as symbolic representation and almost as adaptive as connectionist or subconceptual[190] architectures. Gärdenfors' idea is to represent multi-dimensional contexts as conceptual *spaces*, building geometrical structures on 'quality dimension*s*'. Cognitive sciences will be able to profit from this approach, believes Gärdenfors, since concept formation processes as well as analogy and similarity, to mention a few only, seem to be difficult to model inside conventional platforms. But to incorporate compound data structures and multiple arguments/dimensions seems rather feasible in symbolic terms too. The novelty lies therefore in the treatment *of* the resulting geometrical structures. In metrical defined conceptual spaces, it becomes possible to measure geometrical properties in metrical *terms* such as distance, equi-distance, in betweenness, connectedness and so on. These, in turn, are used to explain "sharp" notions like similarity, sameness etc. As results, Gärdenfors finds that :

> ... possible objects are represented as points in conceptual spaces, a categorization will generate a partitioning of the space and a concept will correspond to a region (or a set of regions from separable domains) of the space. [about properties]).[191]

The notion of 'property', that becomes in conventional "intensional semantics" an abstractum and leads to well-known riddles of induction (Goodman's 'grue' and 'bleen' and his projectibility paradoxes). Not so in the 'conceptual spaces' of properties: Gärdenfors defines a criterion P that says that *convex* regions of a domain in a conceptual space mean that they represent a *natural property.* In other terms, "strangely arranged"[192] properties in geometical terms are *non-natural* and as such conventional/arbitrary in this sense. Criterion P presumes technical concepts for 'betweenness' and 'connectedness' of a region ("no jump necessary"), in addition to metrical domains where distances can be measured relative to the theory of conceptual spaces.[193]

> Constraints like connectedness, star-shapedness and convexity can be used to impose ontological structure on the categorization of the conceptual space, i.e. not any old region can serve as a category. In fact, there is compelling evidence that natural properties correspond to convex regions in conceptual space, and using the idea of a natural property in this way Gärdenfors [2000] is able to sidestep the enigmatic problems associated with induction.[194]

But it goes a lot further. The theory applies *topological*[195] notions and geometrical rules in their theoretical exploitation of 'proto-types' and 'anti-types'. When measuring distances and applying '*Voronoi tessalations'* to conceptual spaces, they can be "split" or "tessalate" into "categorical regions" with boundaries that are calculated from a prototype and similarity thresholds.

> A Voronai tessalation divides a conceptual space according to the *nearest-neighbor rule* which says each point/object in the space is associated with the prototype closest to it. This results in prototypes being centrally located in their category.[196]

The surprising applicability to a variety of domains and even some of the most problematic ones, such as non-monotonic reasoning (concepts), intensional semantics, induction, and as we saw above, the 'concept acquisition process',[197] gives this theory of representation and meaning the status of a whole new 'framework for cognitive representation'.[198] A framework that is meant both for explaining empirical cognition as well as constructing artificial cognitive intelligence. Conceptual spaces can be simulated on symbolic machines, but Gärdenfors thinks a more specialized computational architecture will be a lot more effective.

Until now, we explored the two existing computational representing machines as well as one important prospective framework in the making. Because of the relevance for future music

composition systems we now look at a recently emerging research area in AI: *affective* or *emotive computing*.

Affective computing arises from or/and deliberately influences emotion or affective phenomena (affective cpmputation includes “lesser levels” of emotion like 'attention' and 'interest'). It requires affective representations to be fundamentally integrated with procedural modules (with inseperable states perhaps). This machine approach involves new ways to 'sense' affective-cognitive states in users for example through physical and wearable sensors of various kinds. These may collect information about emotions (stress, frustration, moods) and respond to them as an emotional intelligent machine agent, i.e. seeking to reduce negative feelings, improving interest, attention, self-awareness and other *productive feelings* in human[199] users. In general, it contributes to “pushing of the boundaries of what can be achieved to improve human affective experience with technology.” [200]

Rosalind Picard[201] argues for the importance of their mission:

> Emotions are steadily at work within us, biasing goals and motivations, and ultimately what we think and do. And it's good we can't turn these emotions off.
> [Without emotions]... we would spend lots of time wandering around acting rather irrationally [incapable to choose between some actions][202]

We recall Mithen's very similar view on emotions and music[203].
And Picard goes further and doesn't exclude the design of morals and even dignity into machines in some future time. About creativity and emotions, she sees an obvious relation:

> Some have thought that the entrance to the secret garden of creativity was randomness; others quantum mechanics; others emotion. In my book[204] I talk about some links between emotion and creativity, and could contribute to a new kind of machine creativity. The evidence seems to indicate that creativity involves emotions. Psychologists have even shown that being in a good mood can facilitate creativity, making it more likely you can solve a problem that requires a creative solution. However, emotions are not enough. It is not clear what provides the extra spark.[205]

In the next chapters we look how others have tried to find “artificial matches” to generate such “extra sparks”, somehow like hominids had to find their methods to induce fire and ultimately culture. Affective computing has been explored for use in musical systems already. E.g. “The Affective Remixer: Personalized Music Arranging”,[206] targets to induce affective states in listeners by selecting and re-arranging musical segments (controlled by a probabilistic state transition model with musical dimensions), based on real-time affective cues from a listener.

We have presented four very different representational paradigms: the symbolic, the connectionist (sub-conceptual/ANN), the conceptual (conceptual spaces) and the affective (emotive). In principle, all those representational strategies can be applied to musical representing. But representing musical “stuff” or “sound information”, as we will see soon, is a non-trivial task, but one we have to engage in before representing actual music.

2.4 Levels of musical representation

All of the representation paradigms or strategies must be filled with content or actual data from a domain. A database model e.g. will split a “phenomenon” into records and relations, before filling them with data. In a similar way, 'sound' as phenomenon has to be split, segmented or discretizised. The easy answer is to *depict* the world of sounds just as it *is*. But this idea of pure perceptional information is an illusion. Anything perceptual *is* interpreted and molded by our brain's continuous filtering and representing. Unfortunately, there is no stream of auditory *information*[207] or raw data. Actual “sounds” *are* representations!

Some examples of representational choice are

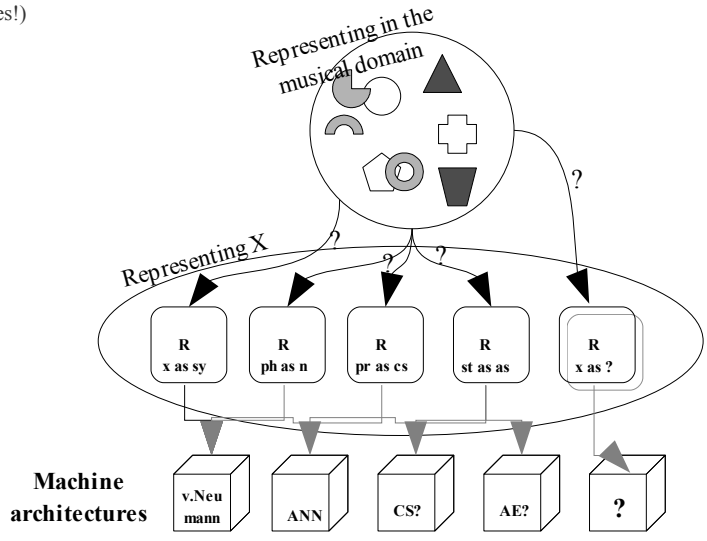
- Should a pause be part of the preceding, following or be an autonomous event?
- What do we make of virtual fundamentals? Virtual pitches in a chord? Or virtual beat in a rhythm?[208]

These questions are trivial for some tasks and non-trivial for others. They problematize that some things are there in some sense, while non-existant in other senses. In general, many “things” or “stuff”[2.3] are there, but not concretely perceived in auditory terms. In AIMA[209], knowledge has an epistemological, logical and implementational level. The issues in this and the next section are exactly the epistemological and logical levels of musical representing.

There are at least three alternative modes for representing musical sounds: the physical (A), the symbolic/notational (B) and the experiental (C). We move systematically through some increasing abstracted levels of representing musical sounds. Naturally, this can be neither exclusive nor authorative, but it tries to exemplify musical representing hierarchies. We remind ourselves of the fact that such hierarchical systems are built differently, according to aims and choices of the representational scheme.

The process of representing 'continuously variable factor information' in terms of discrete categories is called 'categorical perception' in psychology.

Representing in the musical domain: anything musical!
Representing X: B is A! (anything on both sides!)
R(x)=**sy**mbol,
R(**ph**enomenon)=network,
R(**pr**ocess)=**c**onceptual **s**pace,
R(**st**ate)=**a**ffective **s**tate,
ANN = Artificial Neural Network
CS= Conceptual space machine?
AE= Affective machine?
? = Any other machine architechture



A: Physical representing (looking into an oscilloscope)

This way to literally look at sound is how we can circumvent the dilemma of objective interpretation[210] from above. Sound energy is converted directly, ie. without interference of any brain, into pictures that display the properties of sound waves visually to our brains. The observer is now able to seperate the real sound structure on-screen from his interpretation in-cerebrum.This makes this level intersubjective[211] (physics). But can we expect objects? Not really, sound is continuous.[212] Sound is defined as the periodic pressure changes in a medium (air, water etc). Its periodicity or wave is an “organized traveling disturbance in the medium”[213] which transmits energy without transmitting matter.[214] Some “sounds” end with silence. In these cases we know that *a* sound has ended and therefore that the next one is *another* sound. In other cases, an observer will define criteria for 'object-ifying' the scenerio. At this level, nearly all objects are representing sound *and* the theory (including the definitions of compartimentalization decided upon[215]). Still, the picture on the monitor is certainly a representation *of* sound (not sound itself), since it *resembles* or “mirrors” the oscillating waves on the scope”. The “moving pictures” or “sound film” on oscilloscopes display sound properties of higher levels as tiny discontinuities or at scaled resolutions. Other sound measuring and representing devices display sound energy in different “formats”, such as the soundmeter etc.

B: Symbolic or notational representing

Higher cerebrally organized organisms, like bats, mice, apes and humans make sense out of unorganized sound (probably any organism with auditory sense organs) by *interpreting* it. Loy writes the following about this process:

> Our ears are continuously bombarded with a stream of pressure fluctuations from the surrounding air, not unlike the way ocean waves ceaselessly beat upon the shore. Nonetheless, our ears discern discrete events in this continuous flow of sound and assign them meaning, such as footsteps, a baby's cry, or a musical tone. Just as the geometrical point is a mental construct that helps us navigate the underlying continuity of space, so the musical tone is a free creation of the human mind that we apply to the unbroken ocean of sound to help us organize and make sense of what we hear. Though its definiton has been stretched to the breaking point by recent musical trend, tone is still the fundamental unit of musical experience.[216]

'Notes' are the basic unit for almost all notational systems in Western music history. Loy calls our standard notation system 'Western common music notation system' (CMN). A 'note' is the symbolic expression for a tone, i.e. a point where sonic qualities of pitch, loudness, timbre and duration are pinpointed.[217] Proficient readers of the “CMN format” get higher order meanings of melody, harmony etc. from note texts or scores. Scores make use of many special symbols like staff, clef, time signatures and measures or bars etc. Its discrete nature is well-fit for algorithmic and rule-based treatment of problems. The functional[218] representation of CMN is optimalized for human performers (not machines!). Ancient cultures[219] (and old music cultures today[220]) chose other notational schemes, giving priority to different aspects or features of musical events (notes). Ancient Greek notation, for instance, did not convey information about harmonies or even simultanous sounds. Nor did the 'neum' notation system from 9th c. that consisted of dots and strokes written over the mostly liturgical texts. It served a mnestic function providing gestural information, but without more precise information about pitches and durations. With arising polyphonic and instrumental practices, notations had to specialize into what essentially started with Guido d'Arezzo's four-line stave systems and eventually matured into what we now may call CMN. Having said that, special instruments, like percussion, still need specialized notations to represent their sonic characteristics more adeptly.

Similar arguments apply to contemporary practiced notational systems by other cultures. The indonesian “orchestra” Gamelan, for instance, uses 'kepatihan' notation[221] consisting of 6 or 7-pitch pélog tuning systems that makes them often difficult to translate to CMN's staff notation; pitches

are only approximately overlapping. Integration of pitch and tuning systems illustrates how symbol systems are determined by cultural factors, but show also the fact that symbolic schemes necessarily specialize on certain aspects at the expense of others. A universal notation system that would be equally perfect for any thinkable sound event is therefore an illusion.[222] Many of the numerous notational traditions (e.g. Chinese) reflect the needs and decisions to put certain aspects *on* stage or *into* the consicousness of a symbol reader (e.g. singer).

The CMN system is optimized for diatonic scales and it is most precise for equal-tempered ones, see below). It says almost nothing about the timbral qualities.[223] Figured bass notations on the other hand, prescribe only the functional harmonic structure without reference to singular notes. This is therefore a notational level above the 'note' level, more practical in the hands of learned musicians for improvising in real time. In the second half of the 20th c., composers tried new and often more graphical oriented notations, to make continuous and less event-based descriptions of sound structures possible. Nevertheless, the fact that CMN is the standard teaching “language” makes many performers feel alienated to innovative notations and composers gravitate therefore towards CMN none the same.

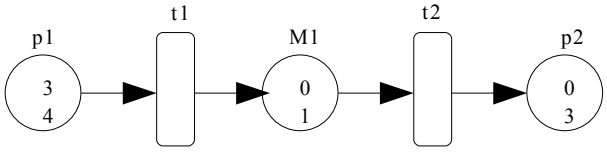
The notable exception is notation of music where computers are involved. Music-performing computers ask few questions about input formats. A group around Goffredo Haus and Alberto Sametti[224] introduced an interesting and computation-friendly notation. Sametti adapted Petri's generalized net theory for arbitrary morphology and abstraction (often causal graphs) to musical time-directed unfolding events. Their software tool, *ScoreSynth*, analyzes and synthesizes musical scores and generates Petri nets with directed graphs consisting of 'places' and 'transitions' (connecting archs).
The open architecture and flexible time notion inherent in the firing rules allow for countless forms of representations and structures. In Petri nets may multiple 'tokens' flow simultanously:

> Transitions in the network state can trigger other actions, such as causing transitions to occur in subnets, nested hierarchically. The flow of time can be made explicit in Petri nets. They can handle deterministic and non-deterministic operations. Representations of music structure with Petri nets is compact and expressive. The elements of a Petri net can refer to musical objects such as notes, phrases, motives, sections and the like, or they can refer to nonmusical objects that manage and control the compositorial process. [225]

Antoni and Haus made an analysis of J.S.Bach's “Canon Perpetuus” from 'Das Musikalische Opfer', and described a temporal logic of this piece. Similar to a particular sonata form scheme (e.g. **1.7**), it shows repetitions only inherently, where CMN writes out most of its repeating structures. Its object-action paradigm (places-transitons) permits hierarchic refining or vertical incrementing structures:

> Petri nets can be developed through a process of refinement, where a place or transformation can act as a placeholder to be given a more detailed description later. “Refinements can define very complex petri net models by means of simple petri nets and hierarchical structures, i.e. allowing models to be designed by either a top-down or a bottom-up approach.[226]

This example of a very simple network plays three notes in M1:



Petri nets show effectively the time directionality and modularity (repetitions) of music in a representation model that is algorithm-friendly, i.e. easy to implement.

Many other music representational systems have been proposed, each one specializing on their own set of musical aspects or qualities depending upon one's aim. Examples are GUIDO, Kyma, DMIX, Score **[ch6]** and finally MAX **[ch3]**. But the by far most used and widely accepted standard of music representations for computers is the Musical Instrument Digital Interface (MIDI). It describes notes with pitch and loudness in numerical values between 0-127 and has a somehow rudimentary timing mechanism. We will discuss MIDI more in **2.6**.

More non-specific symbolic representing of music is taking place by using natural language. To describe perceptual experience in concepts of natural language is a tricky affair. We might say: Oh, I heard something.[227] Or, "I see some sound with the frequency of 412 Hertz on the screen". If we want be more precise, we need the less naïve ways of auditory concepts *derived* from physical studies of the sound (acoustical) as well as psychological, cultural, historical and musicological-auditory knowledge. All those empirical concepts are still abstracting notions from 'continuous sound', and will therefore to some extent always be of theoretical nature as well.

C: Experiential and theoretical representing

Sonic qualities of the tone (pitch, musical loudness, duration, timbre)

We distinct between the non-identical phenomena of acoustic and auditory events.
Frequency e.g. is the physical measure of vibrations per second.
Its auditory epiphenomenon, *pitch***,** is "the auditory attribute of sound according to which sounds can be ordered on *a* scale from low to high.[228] Musical engineering uses more the former concept, while musical activities adopt the latter. This is out of pragmatism, not unlike why one prefers partituras for conducting and voice sheets for playing an instrument. Frequency values occur in non-practical ranges (and resolutions) and change in different ways from what our perceptional experience tells us.[229]

Acoustical	***Relation***	***Auditory***
Frequency	Equal steps of pitch are equivalent to doubling of the frequency	Pitch
Intensity	Roughly proportional, but subject to context conditions with other musical dimensions	Musical loudness
Tone length	Seconds vs. beats (per second)	Tone
Timbre (related chroma)	Spectra vs. instrument name/family	Tone quality or color

Cultures choose privileged tone systems (defined as scales/scale degrees) for the human hearing domain of pitch space in order to allow for possible identification by human brains. The "Western orchestra" uses around 90 pitches in their repertoire in total.
Loudness can be called the response pattern to intensity[230], but it varies substantially with frequency and intensity range. Auditory classification follows often CMN with its qualitative scale from ppp to fff (pinisissimo to fortississimo).
Duration is either the length of a tone event (*absolute* duration) or the time period from the start of one tone event to the start of the next event (*onset* duration).
Timbre is the quality of complex tones (tones with overtones), that we use to differentiate between instrument types. For example, the tone quality of a piano is different from that of a harp. Acoustically, this difference in timbre (or tone quality) is quantifiable as the relative sound energy distributions of all partials (fundamental and overtones). Common instruments have successfully been classified as three-dimensionally spatially and hierarchically clustered to show their timbre similarities.[231]

Sonic qualities between tones (intervals, scales, chords)

When two tones take place simultaneosly (*chord*) or in sequence (*interval*) their pitch distance can be measured acoustically by comparing the ratio of their frequencies. Intervals are tied to the notion of a scale or an ordered set of pitches. Most scales are recursive with respect to the prime interval of an octave, i.e. they have equal steps in all octaves. Western music scales are mostly diatonic, with seven scale degrees (also called pitch classes) in each octave using two *interval sizes*, the half step and the whole step. From the many possible diatonic scales (different orderings of these interval sizes; also called Church scales), Western music uses only two of them as their privileged and defining diatonic scales: *major* and *minor*.
Major scales divide octaves in [1 1 ½ 1 1 1 ½]; minor scales exist in several variants.[232] Substituting whole-tone steps with two half-tone steps, octaves are divided into 12 semi-tones. As a consequence, there is only *one* chromatic scale in contrast to the numerous diatonic scales with variations.
Chords are stacked groups of tones played simultanously. Their respective intervals (relative to the scale context) define *chord type*. In Western music tradition the various forms of *triads* (including their extensions) have a defining role due to their context-building effects in the so-called tonal system. "Tonal music" is well-known for its directedness, i.e. listeners feel a propulsory effect, known as *progressions*. This quality of tonal systems, seemingly derived from the assymetric quality of diatonic scales, makes listeners expect following events in relation to previous events (not very unlike Markov chain formalisms of some types, **[ch7]**).

Qualities of tone groups

Melodies and *rhythms* are multiple tone events in temporal successions. Melodies are about both pitch and duration, while rhythm is about durations only relative to a grid of beats. Melodies and rhythms are meaningful and recurrent structures of musical works. Subjects or themes are melodic and rhythmic shapes with significance to the overall form of the piece, notably. their logical structure.

Qualities of local and global contexts

Progressions, cadences and modulations are multiple chords in sequence that have structural meaning in the tonal play or harmonic structure of the whole piece. Such chord sequences are said to have a principal chord or "representing key". Such a key is said to be the functional tonic or center key. Interesting, in notational systems, naming a key can be relevant for either local/vertical structure alone or classify a local horizontal contexts as well **[ch4]**.
Works with several movements are analyzed for their home key and analyzed thereafter locally relative to this key.

Qualities of work collections (style, genre)

Work collections are categorized as compositional *type* (sonata form e.g.), *style* (impressionistic e.g), *period* (classicism e.g.) or *genre* (entertainment or church etc.), to mention some of the more central categories.

Universals in Music

Musicology and Music Psychology are continously looking for absolute properties in musical features. Examples are culturally neutral characterizations of musical harmonies (or melodies) as pleasing or non-pleasing, leading to categorical studies of *consonance* and *dissonance*. All in all, there seems to be only partial success in pinpointing such musical universals. Mithen[233] refers to recent studies that correlated mostly "superficie attributes" of volume, tempo, rhythm and pitch with specific emotions[234] Such emotion inducing variables may well be of interest to researchers of affective computing **[2.3]**.

The higher the level of these three modes of representation (A,B,C), the more specialized are their "users". Listeners, dependent on their developed musicality or musicianship prefer lower or/and higher levels. Composers, musicologists and other professionals will look at both low and higher levels in conjunction.

Curtis Roads[235] distinguishes between 'microsound', sound properties inside *a* tone event, and 'macrosound', referring to the resulting relations and properties of combinations of tones.

Choices of representational modes and levels are determined by interest (goals), culture and theory one is working from within. In addition, general attitudes towards music, technology and philosophy are entering in such "decision processes". We will present a couple of arguments about these issues in the next subsection.

2.5 Problems and prospects for discretization and formalisms in music

From the flux of Protagoras, to the ideal reflections of Platon; from the the mirrors of the mind that represent the world of Descartes[236] to the *primitive, compound* and *complex ideas* in Locke; 'representation' has been an issue to Western philosophy ever since. No wonder the questions we have posed so far as to *how* musical entities should be represented are not obvious ones to answer. According to Mithen, we recall from the beginning of this chapter, language in Sapiens was the beginning of referentiality (and compositonality). But even though 'Hmmmmm'[237] **[2.1]** was holistic, music is certainly not. We understand the various modes and levels of musical representation quite intuitively**[2.4]**. But which representations of music are natural (or innate) and which are conventional? At which level of representation should we object-ize the perceptual flux of sounds? Is the tone or its sub-tonal attack part[238], the phrase or its reduction in a Schenkerian analysis, the style or the formal structure of one of its instances an adequate and correct way to represent musical meaning? It depends all on a series of factors. It is for instance relative to whether we want categorize pitch or timbre, melody or harmony, pieces or forms etc. Musicologist Nicholas Cook[239], finds "simple verbal descriptions of the musical experience a practical starting point for analysis", after having explored at length the intricate and in his view problematic alternatives of various score-analyzing formalisms. But formal analyses with representations of e.g. 'Ursatz' or set-theoretical terms may be of little importance to listening or evalution (according to Cook), they are definitely fertilizing composing. Cooke describes 'relativity of musical representation' in these terms:

> ... the value of an analysis consists in what it does for the analyst, then it is plain that what would be a bad analysis under one set of circumstances can be precisely what is wanted under another.[240]

Formalisms are systems that place chosen elements into defined relational contexts. Typical formal systems are logical languages with explicit syntactical and semantical rules. Many formalisms for musical descriptions have been built with numerical elements and arithmetical rules of manipulations. Such systems have what Rolf I. Godøy calls 'monstrative' capabilities[241], i.e. they demonstrate or show rich relational schemes in comprehensive ways. But such tempting clarity is often confused with a sense of reality or rationality, that may lead activities go astray. Many compositional projects in the classical realm of 'computer music', roughly between 1950-1980, have been somehow self-centered in the sense that they did not care so much about listeners, that were left alone in their struggling attempts to draw meaning from these sound-complexities. It is very much easier for a Phytagoraen experimentalist to appreciate "sound-calculus" than for its rather bypassing spectators. We may say that many of those compositions were more machine-music for machines than music by machines for humans. The monstrative lucidities may also have been reinforced by positivist and reductionist flavours of modernism at the time **[3.2]**.

A more practical and concrete formalism uses the tone as primary level of representation: MIDI[242] is

a highly standardized and widespread sound language. Its protocol defines 7-bit data bytes encoding information about time (*Note-on, Note-off*), pitch (*key number*), loudness (*velocity*) and channel, in addition to the so-called "continuous controllers" conveying rather crude gestural information. This means any MIDI-message or byte (pitch, note-on etc.) is restricted to values from the range between 0-127. MIDI is much used for real-time controlling multiple digital MIDI-instruments and computers. Its interface-protocol is now nearly universally supplied. In **ch3** MIDI will be "showcased" in relation to MAX. Just so much for now, MIDI has inherent limitations, its handling of time information happens to be problematic (two messages for a single tone[243]), it includes only very limited information about timbre and finally it has very rudimentary gestural[244] controlling capabilities. Still, it demonstrates the monstrative capabilities quiete well. Composing or sequencing software is able to do complex and transparent manipulations of MIDI-events, very much contributing to its success in both commercial use and research. There have been recent initiatives to create new and/or expanded MIDI protocols (MIDI2), but MIDI is, as we speak, still *the* protocol in town.

Godøy in "Formalization and Epistemology"[245] calls the non-represented, i.e. uninterpreted sound 'musical substance' and believes that substance to have a "distributed nature". His stated "aporia of the continuous and the discrete" lies:

> ...between the infinitely varied and complex on the one hand and the relatively simple and coherent on the other...the focus is a discretization of the continuous nature of the distributed substrate...Ignoring the distributed nature of musical substance may lead to the kinds of category mistakes and constriction of perspective... Ignoring the need for a focus of some sort will make the musical substance both incomprehensible and unmanageable.[246]

Godøy's phenomenological and hermeneutic perspectives lets him pursue 'morphodynamism' as methodology in his research program, with bodily and musical gestures studied in integration.[247]

> Discretization is that which opens up the possibility of scrutinizing the musical substance, as *the flux is discretisized into an object which is overviewable as an image.* This is one of the basic maxims of morphodynamism to which I always return.[248]

The pictorial/spatial and topological characters of these techniques capture music-related gestural information as geometrical rather than arithmetrical information and involve carrying content from one quality dimension to another and vice versa. Two observations arise here. First, the tight coupling of body movement and music is in fact what Mithen's theory on the origin of music is predicting (or explaining).[249] Second, the "geometrical turn" of computing with 'conceptual spaces' (Gärdenfors) is by and large compatible with Godøy's approach. Actually, the concept of *integral dimensions* is described as:

> For example, pitch and volume constitute the integral dimensions of sounds discernible by the human auditory perception system. Integral dimensions are such that they cannot be seperated in the perceptual sense. The ability to bundle up integral dimensions as a domain is an important part of the conceptual spaces framwork. Domains facilitate the sharping and inheritance of integral dimensions across conceptual spaces.[250]

These "integrationalists" advocate prudence in relation to *discretization of musical information into objects*. When too much of the "living" structures are abstracted or sliced away in using low level and low resolutioned representations, we risk to loose some of the relational realities and richness beneath, somehow like "througing the child out of the bathing tub".

Alternatives to the somehow "sanctifying" of phenomenological experience are recently under making. Daniel Dennett has made a proposal of new scientific programmes under the umbrella of *heterophenomenalism*. In contrast to what Dennett calls autophenomenalism (of the traditional Husserlian kind), heterophenomenalism is the objective study of subjective experiences in a third person. One accepts here fully the subject's self-reports and supplies them with behavioral and situational data to model experienced mental states in a combined first and third person perspective. It seems to bridge over subjective and objective stances or experiences, believed by

phenomenologist theories (from Descartes or even Plato) to rest on an abyss. In practical terms this means to accept the fallability of the subjective stance. For example a person feeling not nervous or hearing a missing fundamental, can be told to err without making the subjective stance irrational or non-authorative[251]:

> The total set of details of heterophenomenology, plus all the data we can gather about concurrrent events in the brains of subjects and in the surrounding environment, comprise the total data set for a theory of human consciousness. It leaves out no objective phenomena and no subjective phenomena of consciousness.[252]

In relation to subjective auditory experience and its representational pauperisms, 'qualia' are often considered the holy spirit of consciousness (in addition to creativity, **1.9**) and believed to be private, inaccessable (see heterophenomenalism), as well as atomistic. Joseph Goguen has in an article. "Musical Qualia, Context, Time, and Emotion", targeted this conventional views about qualia. He thinks to have shown that qualia is the expression of multiple and dynamic parts in a salience-based hierarchic model.

> The experiments shows that qualia have structure and exhibit lawful behavior; they are *not* an unfathomable residual category. They also show it is far from adequate to say that the quality of experience depends on context. ... Using music as a concrete example, we argued that consciousness has more structure than generally supposed, including a hierarchical decomposition where each part has its own saliency and emotional tone, we also sketched an evolutionary basis for this, and showed how notions like context, foreground and background arise naturally. [253]

His findings are again in harmony with the theories of Mithen, Gärdenfors' conceptual spaces (blending e.g.) and musicologist Meyer. The anticipatory effects of music, their evolutionary basis, the emotion-involving structure of all qualia and contextual and time-related parts of his theory are very much of interest to the questions arousing around representation of musical elements.

The irreducibility of consciousness-related phenomena and states have been subject already in relation to creativity **[1.9]**. In this area too, brain research in alliance with philosophy seems pushing borderlines, thought to be untouchable (or at least non-negotiable) until now. For instance, eliminative materialism (Churchlands) had for some time now, accepted a non-reductionistic nature of the mental in relation to brain events. John Bickle's "new wave reductionism"[254] has started to collect research from molecular and cellular cognition into a "ruthlessly reductive account" of the mind-body relation. His psycho-neural reductionism has been said to open the doors to new theories that may solve many confusing properties of consciousness. Only time will prove the relevance of such research to musical representation modes and levels, but it should nevertheless remind us of the astonishing record of scientific progress in matters of the human nature, including musicality and musical intelligences.

In any case, the considerations presented until now are relevant to formalist ambitions in general. But we have to consider the different *kinds* of formalist programmes as well. The *empirical* and musicological formalisms, for instance, seek out models for their apparent "truthfulness" with existing organismic and cultural realities. Formalisms for *artistic* uses, on the other hand, and especially the ones designed and engineered for implementation on computational machines, will therefore be subjected to different requirements.[255] Otto Laske calls such constructivist activities 'action sciences', concerned not so much with 'competences' but 'performances' and taking evolutionary views in his proposals for a "new musicology".

In addition to symbolic logical systems, there are many alternative formal systems that may be more adequate for musical discretization problems. Systems of 'Fuzzy logic', 'Modal logic' are examples here. An interesting combination of spatial and logical expressivity is the formal theory of *mereotopology*. It specializes in relationships between parts, and parts and wholes. Achille C. Varzi[256] finds that mereology permits not only ontologically non-commiting formalisms (being compatible with nominalism, presupposing less than set-theory[257]), but that its applicability for

ontological analysis [and production] seems wider than that of many other formalist paradigms. Mereotopology requires a single primitive relation only, like the dyadic Parthood (*Pxy*). Adding morphological or "qualitative geometrical" expressivity, kinematical and dynamical sound worlds may be tractable with this formal-ontological apparatus.

In relation to normative questions about formalisms, representations and discretization, we might finally take a look at the 'principle of tolerance" formulated by philosopher Rudolf Carnap .[258] Carnap proposed several "definite" formal language systems"[259] in the logical and linguistic domain, before he concluded that pluralism and pragmatism (hence tolerance) seemed to be the only "truth" to this matter. Carnap wished to make scientific practice more precise and clarified and recommended to this end the use of calculus-like syntax'es and extensionalist semantics. If tolerance was the consequence for scientific formalisms, tolerance regarding artistic formalisms should be considered an even more adequate principle. Gareth Loy exemplifies tolerance and relativity of representation for the musical domain:

> Each view has advantages and disadvantages. The functional view provides a great deal of information about how a particular performer realized (performed) the note, allowing us to analyze the physical vibration of the instrument.[260]

Another argument for freedom in formalistic expression comes from composers themselves, when looking at music history in their own, rather ideosynchratic, but elucidating ways. Cook writes:

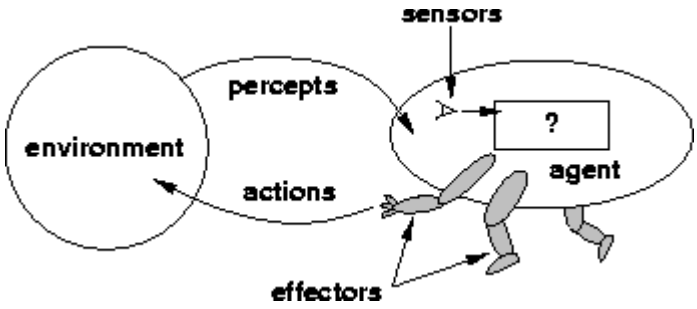
> [about the analyses serial composers of post-war Europe]. By and large these analyses were speculative to the point of irresponsability and, in comparison with any remotely competent Schenkerian analysis, they were frequently downright unmusical. But at that particular time a and place conventional musical sensibilities were not what was needed. Their analyses were good, not because they had any generally applicable validity, but because they stimulated an outburst of creative innovation in musical style.[261]

Compositional practicians might learn from the examples of serialism and 'computer music' in order to avoid falling into the familiar traps of hypercomplexities' irrelevances caused by abstract and non-adequate formalisms. However, composition as a creative activity should be able to use their privilege to execute experiments in the spirit of 'free composition' style,[262] without pressures to comply with truthfulness. Such independent-minded projects should be able to use the full resources of ontic engineered tool boxes, and they should hopefully extend to machine composition as well.

2.6 Conclusion: Ontic engineering of musical objects, processes and worlds

In **ch2** we naturalized the sources of our inborn musicality **[2.1]**, restrained the powers of sector-building proficiencies and recommended "full-cycling" musicianship **[2.2]**. Finally, we looked at representational issues in both machines **[2.3]** and humans **[2.4, 2.5]**. It is now time to bundle the threads from **ch1** about algorithms with the objects from representations. Algorithm-like schemes of composers will become full-blown computational algorithms in a *compositional agent* that uses percepts from environments to act on its environment by responding to and emitting musically meaningful structures, structures we ultimately may call compositions (AIMA, general agent model, **fig.2.1, 32**; below).

Such an agent will have both goal-based and utility-based to facilitate an optimal blend between autonomous ("happiness") and heteronomous goals of artistic scope.



To achieve these ends we must engineer an ontology of objects and processes for the virtual worlds compositional agents will have to reside in. Objects may be MIDI-messages or affective topological atoms; processes may be the generative principles, reverse engineered from the generative theory of the tonal music by Lerdahl/Jackendoff, or any inductive or learning algorithms meant to be executed on one of the in **2.3** mentioned machine architectures. Environments may be open or closed.

The "psychological model" for our agent should be based on constructivism for its innovative or creative power (constructing new world perspectives), on cognitivism for its variational reason (rational and planning potentials), and finally behaviorism for its reproductive potentials (learning and pattern based logic). These three models are reflected in the three "master modes" of musical creativity and learning (learning cycle!, **2.2**), namely contrast, variation and repetition.[263] This *trinity of expressional laws* will be the driving motor for algorithmical logic in Machine Composition.

Otherwise, there should not be 'a priori' frameworks to delimit, inadvertantly or conventionally, the constructive approaches of Machine compositional agents. It will be their success and not there strategic choices under the building processes that decide their survival and hence truthfulness to their world model. This is in alignment with Dennett's *generalized concept of design*, where:

> ...my examples in this chapter have wandered back and forth between the domain of organisms or biological design, on the other hand, and the domain of human artifacts – books, problems solved engineering triumphs on the other ... There is only one Design Space, and everything actual in it is united with everything else.[264]

We may add 'musical compositions' to the list of human artifacts, that are proposed in the *General Design Space*, in analogy to the search spaces of **1.3**, and that alouds "authorized configurations" to emerge as solutions after the necessary number of trials have perished in this "space" of designs.

Such systems will also have to modularize into hierarchic and non-hierarchic societies of composers (Society of Minds, Marvin Minsky), e.g. modeling the evolutionary strategies of Peretz modular model from **1.1**. But equally relevant are modules of reflective functionalities describing the three master modes of expressing, mentioned several times already: contrast, variation and repetition. An important bundle of problems related to design, will be the various challenges in mapping sound features to higher features, and sound features to control parameters.

Bio-inspired systems[265] with non-neglectable unpredictableness[266], are the horizon of Machine Composition. In addition, there will be rational assistents to composers that will use such tools, that are built on tools for building them in the first place. We will have to build on optimism too, in order to to aloud for some degree of the Sourcerer's apprenticeship in our new tools of compositional creation. In **ch7**, we will even look at what was called Frankensteinian models of Composition. We might with time see the growth of affective and other higher-level properties inside MC systems as well.

As to the question of which epistemological choices one should proceed with, we may look for

naturalized epistemologies, i.e. expecting answers from theoretical psychology[267] or in the present case from evolutionary musicology. For constructive purposes, we might content ourselves with a *procedural epistemology*, a non-commiting and pragmatic stance for the ontic design and engineering. Such an attitude asks how-questions instead of why-questions during the development of musical objects, algorithmic processes and environmental "worlds". Abelson and Sussman:

> Computer revolution is a revolution in the way we think and the way we express what we think. The essence of this change is the emergence ... of the structure of knowledge from an imperative point of view, as opposed to the more declarative point of view taken by classical mathematical subjects.[268]

The first two chapters attempted to provide

- clarifying concepts and a procedural sharpening of our tool boxes **[ch1]** and
- a view of musical composition in integrative perspectives (both in the evolutionary time scales as well as learning time scales, **ch2**) and
- a more balanced and unified approach to human activities as a whole (i.e. viewing creative activities as normal and naturalized) incorporating both instrumental and expressive/artistic cultures as recombining and meaningful parts of life.

Hopefully, this may contribute to the liberation of the powers of "engineering imagination" and new man-machine contexts that satisfy our inherited senses and understandings of musical composition, as well as they may build new levels of culturally significant Machine Composition. In **ch9** we will revisit and somehow revision the intuitions inspired by these thoughts. The next six chapters will overview some of the most influential examples and applications of Machine Composition in found in academic research today.

Ex nihilo deus → ex nihilo genius → ex nihilo machina
ex dubitatio nihil crescit → ex nihilo nihil fit!

Chapter 3
Objects and Messages : Machine composition with MAX
Patches with Max/MSP, Pd and jMax

1. Introduction
2. Early computer music
3. Basics of MAX
4. Types of objects
5. Types of messages
6. Programming methods and approaches
7. Composer objects and examples
8. Max/MSP, Pd, and jMax: dialects of MAX
9. Conclusion: MAX and machine composition

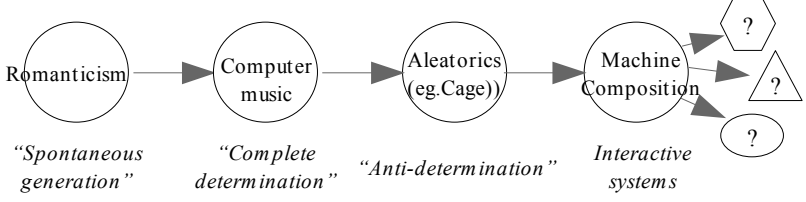
The computer should ideally feel in the musician's hands
like a musical instrument, needing only to be tuned up and then played.
(Miller Puckette, 2002[269])

3.1. Introduction

In **ch1** we looked at common ground of algorithmic thinking in computer programming and music composition. In **ch2** we discussed problems related to choosing a representational language for machine composition. In the following I concentrate on MAX as a programming tool for MC. MAX uses MIDI as its primary representation. It was developed at IRCAM (Paris) around 1986 and was in 1991 commercialized by OpCode (since 1999 Cycling74). But first we make an excursion into the historical background of MAX: early 'computer music'.

3.2 Early computer music

Some composers showed early an interest in electronic technology and used tape recorders to manipulate recorded material. Computer composition is generally assumed to have started around 1960 with Mathew's general-purpose music programs at Bell Laboratories. Computers worked extremely slow from our perspective, and it therefore required both time and patience to get results. Nevertheless, some composers enjoyed the precise control over many aspects of the sound result. Still, by substituting human performers on a stage with tape players one impoverished the experiential content of concerts. Audiences often got “disconnected” and failed to see a human dimension. A tape *as* performer may satisfy the composers intentions more than the expectations of an audience. Live electronic systems seem to close this gap. Typical set-up consisted of modules that delayed tapes or processed sound from acoustic instruments adding thereby effects, such as reverberation, distortion etc. Many early experiments were done in a spirit of control and determinacy. John Cage provoked aesthetic discourse about indeterminacy and improvisation contributing to the growing and ongoing interest in interactivity in musical composition.



Voltage-controlled synthesizers soon became the standard in computer music. Similar to MIDI becoming a de-facto standard for control, any synthesizer module now responded in the same basic way to varying voltages in changing the parameters of emitted sounds. (see also Subotnick[270]). In the 1970's CM mostly meant sound synthesis and sound processing on mainframe computers. This changed when two important developments fused around 1984: digital technology and MIDI language (International CM conference, 1984). Digital technology made computers and synthesizers (e.g. Yamaha's DX7) affordable for people outside research labs. The MIDI-protocol (Musical Instrument Digital Interface) became the universally employed method for sending and receiving musical information digitally. When music hardware became digital as well, these technologies furthered flexibility and modularity of systems. Timing problems for parallel streams of notes were now possible to control. Once instrument manufactures had agreed on MIDI, practitioners and researchers established common software grounds for CM systems. Research at IRCAM (Paris), STEIM (Amsterdam), MIT and Carnegie Mellon was necessary before todays consumer music programs such as sequencers, sound editors, notational programs and interactive software could be developed and standardized (M and Jam Factory, 1986 , **ch4**).

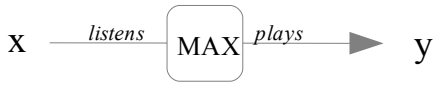
In the last 15 years many CM-friendly programming environments or systems have been developed. We shall take a closer look at Cypher, EMI, CommonMusic and others in **ch4,5,6,7**. For now we look at the perhaps most used platform for MC: MAX.

3.3 Basics of MAX

Miller Puckette created MAX originally as native code control system for IRCAMS 4X synthesizers. Later it was generalized and adapted to Macintosh by David Zicarelli. Puckette made a version for IRCAMs new Signal Processing Workstation (ISPW). In this MAX-version for NeXT computers audio signal production and processing was now possible as well. The audio-part of the software (FTS) evolved later into the 'creative commons' version of MAX, called "Pure Data" or PD. PD contains numerous extensions contributed by the user community of MAX, among them interfaces for video programming, 3D-graphics (GEM) and light systems for multimedia applications.

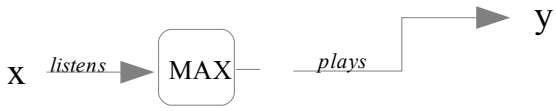
MAX is an object-oriented Fourth-generation language (4GL). Building MAX-patches means building collections of interacting objects (“objects are algorithms that precipitate action”[271]). Each object has a function and depends on its neighboring objects through communication of messages. Messages are information passing through the objects, received only, stored, answered or passed on unchanged or processed. Messages can be instructions for start or control of processes. MAX is a specific purpose language and does therefore not offer the full flexibility of other fully object-oriented-languages. MAX makes modular programming very easy. Small subroutines (objects) build up bigger subroutines (more objects) and so on. MAX programming is therefore intuitive, bug resistant, reusable, modifiable and as a consequence rather easy to learn.

MAX is an algorithmic and graphical programming environment that is widely used in machine composition circles today. It incorporates a MIDI-interface to facilitate programming with MIDI-sounds. MAX both *listens* to streams of MIDI-messages (input) and *plays* MIDI-messages (output) to a synthesizer (software or hardware) on the same computer or an external sound module.



What can we do with MAX? We can make a little game with MAX. MAX listens to small melodies and answers them by playing them back transposed (like an “echo spirituoso”). This is not exactly

impressing MC, but it illustrates the basic parts of such a system. It involves both transformation and decision-making. How do we solve such a task in MAX? We will fill in the unknowns throughout this chapter.



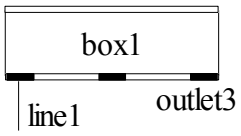
MAX is an object-oriented system i.e. data are represented as objects that connect in a network (called *patch* in MAX-lingo referring to the physical wiring and rewiring in analog synthesizer systems). We can group MAX-objects into functional categories. Some objects parse and process input-streams of MIDI-messages (*listening objects*); others generate or compose sound and send output-streams of MIDI-messages. Other MAX-objects work with notational or graphical representations (*score objects*) while performance directed objects increase interactive and interface functionality. Recent versions of MAX add multi-media objects for use with pictures, videos and lights. These extensions transform MAX increasingly into a general-purpose *control programming language*. But its history and basic approach (with MIDI as fundamental message system) classifies as music composition tool.

In its first versions MAX was limited to handling MIDI-messages. Today, MAX-programmers have tools for working with raw data of sound, permitting manipulation of physical sound structures (at a lower level than MIDI). Using these tools, machine composition can include sound synthesis, mixing and signal processing. In this chapter we limit ourselves to programming of MIDI-messages;

MAX typically operates in an interactive or interpreter-type programming style. The resulting MAX code is compilable, but the graphical interface with objects/messages and connecting lines invite to continuously test the audible results of patches (in RUN-mode).

Interactivity is a property we typically find in performance and improvisation (even solo improvising is a kind of answering ones own questions). But, as we saw in **ch2**, composition feeds on learned material. More importantly, most serious and interesting works seem to evolve from doubts and reflections similar to generate-and-test principles in creative computer programming **[ch1]**.

In Marvin Minsky's “Society of mind” the complexities of mind are broken down into smaller functions. Using MAX we may break up musicality into constituent parts or functions (objects). Objects output messages in response to receiving messages. What happens inside objects can be hidden (encapsulated) on the screen, so users can concentrate on what goes on without distraction by how it is implemented. MAX objects process data i.e. manipulate, generate and store data. Objects are algorithms that precipitate actions and they listen (observe) and play (act) according to messages.



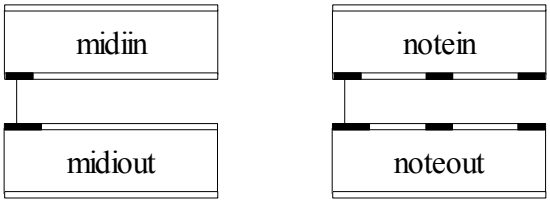
Graphically, objects and messages (i.e. patch) are displayed in a window by boxes (objects) and lines (connections), messages are not always visible, but can for various reasons be displayed by indicator boxes. Messages originate either spontaneously from inside objects (random), or come from a user (keyboard/mouse or MIDI). Messages then flow downwards in the patch (along lines) depending on the definition of the objects and their connections. Sub patches can be hidden in a sub-window. Programming or editing of patches is done in *«Edit mode»*. Execution of code/objects occurs in locked *«Run mode»*. Easy toggling between modes, makes MAX a highly interpretive

environment. Finished projects can be compiled for efficiency. Programs can be read from graphical representation or textual print of patch definitions.

A first patch: zero-patch

Knowing the different types of objects and messages is important for understanding MAX-patches. Objects have different numbers of input and output, called *inlets* and *outlets*. Some inlets and outlets must be connected, others are optional. Objects without any connections are inactive and hence without meaning. We may call a patch *zero-patch* when it is the simplest possible or atomic program with information entering and leaving unprocessed (e.g. input-object *midiin* and output-object *midiout*). It corresponds to an identity function in mathematics: f(x)=x. Between those two objects there must be at least one line or connection, sending everything from the input-port of the first object to the output-port of the second object.

Using *notein* and *noteout* instead we can process pitch and volume separately since the objects split note information into its constituent aspects. The left outlet of *notein* sends the MIDI-number for its pitch of the notes passing through. The middle and right outlets sends MIDI-numbers for volume and duration. Since the inlets of noteout-objects reflect this layout, we can make a slightly more advanced zero-patch or neutral program, that does not change anything, by connecting the three outlets of *notein* to corresponding inlets of *noteout*. As a result, information about notes are now split into three aspects and sent further through its three connections. For each note, *noteout* now receives three messages, not one as in *midiout*.



The objects used so far receive and send/transmit messages. Sometimes we need objects for displaying messages and stopping messages. We think of three “empty” 'indicator boxes' between *notein* and *noteout* lines. We now have a window into the programs message flow. Any time a note plays (travels through), indicator boxes display from left to right the value for pitch,volume and duration. Values are only visible/stored until the next incoming note initializes these values. Our patch is still neutral, doing nothing other than displaying input, so here is what we got so far: boxes, lines and visible messages in a working and transparent patch.

Processing patch (“note-amplifier”)

We were able to confirm that patches change nothing; zero-patches receive, display and transmit everything entering it at the top. We expand this neutral patch into something that does a little bit more. If we connect empty boxes under the display-boxes and fill '+1' inside them, the patch increases pitch, volume and duration with one step corresponding to one integer in MIDI-values. That means in practice incoming notes will be played one semi-tone higher, one MIDI-value louder and longer than before. We can again add three indicator-boxes to demonstrate it. Any other arithmetic operator and value can be used to process input, i.e.change values passing through the patch. We can compare the contents of indicator boxes before and after the processing objects (“+ 1”) to see that all incoming note values increase by one. We might call this patch a *“note-amplifier”* patch, since each received note is transmitted a little higher, louder and longer. We now comment the patch: comment boxes are objects that show literal and static text explaining the intention of the patch creator.

It is easy to see how more complex arithmetic processing will lead to programs of increasing complexity. Our echo game example from above should involve processing objects that control time delay and decide on the distance of transposition. Another example is a patch that increases loudness negative proportional to pitch values, i.e. low notes (midi values for pitch) means high volume (midi values for volume). Or any other logical relations between loudness and duration of notes, or any other variables of input. A *zero-patch* just accepts input and passes on. A *processing patch* analyzes input or listens and changes/manipulates output according to the intentions in the patch. Generally we may divide patches into *negative* and *positive patches*, i.e. patches that filter/scale or subtract information and patches that construct or add information.

Positive patches use incoming information as triggering data, building blocks or variables used for more complex output. **Negative** patches use incoming information as raw data for more ordered structures. Patches may use non-human data as input for meaningful constructs. This is typical for "computer music" . MC employs both types of patches, so did and does classical "computer music".

3.4 Types of objects

MAX objects are graphically represented as boxes in MAX windows displaying patches. In addition to predefined or standard built-in classes of objects, indicator objects and comments, there are a number of interface objects. They give the user direct control during run-time through mouse or keyboard. Examples are toggle-switches, sliders and bangs (emitting «do-it»-messages)

Predefined objects are the heart of MAX. In addition to arithmetic objects, we find objects for storing or recording consecutive data (*seq*), objects that generate random numbers (*random*), objects that generate regular streams (*metro*), objects that delay messages with a variable or fixed time (*delay* and *pipe*), filtering objects such as *stripnote*, *unpack* and constructing objects such as *makenote*, *pack* as well many others. Input/output-objects for other types of MIDI-information deal with continuous information in musical streams or sequences (*ctlin*, *bendin*, *touchin*, *pgmin*).

Objects have defined and limited numbers of parameters determining their behavior. They are graphically represented as inlets (input) and outlets (output) of boxes. Example: a delay object receives data at its left inlet and the amount of time of delay at its right inlet.[272]

Parameters are constants when their values are fixed and variables when their values can be changed during execution (run-mode). Parameters can be initialized at certain points in the patch (e.g. by a bang object when clicked by a conducting or performing user).

MAX has an active user community. Their code contributions built a huge library of user-defined objects. New objects can be written in C to further expand its predefined object library. We will go into more detail later about integration and import of Lisp-code into MAX-patches (MaxLisp v.0.8,**ch8**).

Predefined objects have been categorized by common high-level functionality: 'Listening objects', 'Composing objects', 'Timbre objects', 'Signal processing objects', 'Score objects', 'Multimedia

objects'. Lists of MAX-objects and types are found in MAX literature and programming guides, e.g. “Composing interactive music” by Todd Winkler (1998).

3.5 Types of messages

What is a message in MAX? Any information passed from one object to another. Messages are the data that flow through a patch and are temporarily stored inside objects of that patch (similar to Unix pipelines[273]). Messages can initiate inside an object or respond to other incoming messages from whatever source. These are the types of messages in MAX:

- Numerical messages: integers (*int* and *float*)
- Symbolic messages: words (*symbol*)
- List messages: lists (*list*)
- Special messages: *bang* (“do it!”)

MAX data flow consists of either musical domain data (mostly numerical MIDI values) and control data (symbolic or numerical values for algorithmic control).

Numerical messages may denote a description of an incoming event or outgoing event, but are also used for algorithmic processes inside the patch.
Symbolic messages are e.g. used to control *delay, metro* and *seq* objects using '*start'* and/or '*stop'.*
Lists can consist of numerical or/and symbolic data [274] that describe domain data, e.g. a list may consist of note values for pitch, volume and duration, i.e. one list item describes one note [60,39,10]. Or one list item may describe the individual notes of a chord .
Special messages like *bang* are used for interface objects where a user starts a process or data flow. If a *bang*-box (i.e. message box containing “bang” as text) is clicked, a “do-it!” message or bang will flow downwards to the next objects inlets.
Messages may be stored in objects before they are processed and transmitted further.

Still more complex data structures are *table*s and collections (*coll)* and the specialized array-type objects like *funbuff* and *bag.* Common for these objects are array-like structures with index and data-pairs. Some (*coll*) do sorts, swaps and inserts on their elements, others (*funbuff / bag)* include min/max element extraction. We will see some of these structures exemplified below **[3.7]**.

Message arguments
Messages may have variables. A $ followed by a place number (e.g. $4) will be set by a corresponding incoming list item (e.g. 4 if list=[1 2 3 4 5]). Messages are manipulated by objects like *pack* and *unpack,* items added to messages with objects *prepend* and *append*. List-separating *iter,* list-constructing *thresh,* and list-sorting *sort* also change message's contents. As a general rule MAX objects make it possible to *decrease/increase the size* of or *reorder* i.e. change the *contents of data structures* that circulate in a patch. Message flow is defined as *logical routing (space)* in a specifiable *order of operation (time)* of information passing through the pipes and pathways of MAX-patches (not unlike the task scheduling in Unix pipe-lines; Rowe, *IMS*, 2001)

Data routing is either defined by conditional statements with relational operators (<,>, ==, if) [275] or by predefined data routing objects that select or identify specific values or ranges of values (e.g. *split* and *select)*. Most data routing is controlled by musical message values, but special objects like *gate* and *switch* have extra inlets that are used for explicitly opening and closing messages from decision making objects in the patch or user controlled objects. Messages generally follow pipes (connecting chords or lines) through objects (or boxes). An exception to the rule are the so-called remote messages (*send* and *receive*) that are “beemed” messages without any “physical” connecting chord in the patch, i.e. s*end*-objects transmit messages wireless to *receive*-objects[276].

Ideally all events or messages should execute with no time delay. In a real-time-environment like MAX, order of execution on "machine level" often becomes essential. As a matter of convention, multiple events generated by objects are *scheduled right-to-left* (bottom-to-top if they are graphically placed perfectly vertical).

3.6 Programming methods and approaches

Computational problems should generally be solved as simple and effective as possible. But how do we find good solutions, in our case a "good MAX-patch". We will encounter problems that are common to all types of programming languages and paradigms, such as design, structure, modularisation, interface questions, debugging and others.

Since MAX is an interactive programming environment, programmers may experiment continuously and spontaneously during programming a patch. Starting with drawing boxes and lines, then writing essential definitions of objects into boxes and using functional inlets for their connected chords. Once parts of a patch seem workable, one switches from Edit mode to Run mode for inspecting the messages coming in and out looking at the values in display boxes in between. Such a process of continuously creating, testing and refining is most natural in MAX programming. It makes debugging focus local and easy. A more structured approach is Bottom-up programming defining first lower levels of data (raw data) before continuously increasing the levels of abstraction in message processing. The opposite approach, Top-down programming, is the most structured one, starting with essential, logical descriptions of processes and increasing step-by-step the level of written-out detail (ie. reducing abstraction). This rather architectural approach may concentrate on columns and roofs before adding nuts and bolts and only ultimately supplying colors and superficial design. Top-down programming with MAX means to start with objects that define an overall musical idea or high level functions such as "Find a matching chord" or "Construct a contrasting melody" hidden as sub-patches inside referring objects. Top-down design is best suited for larger scale projects that require considerable thought and planning. Sketch books of Beethoven illustrate typical Top-down thinking.

> A top-down design helps give the finished piece a viable structure with parts integrated into a coherent whole. (CIM[277])

Well structured programs have some degree of hierarchy built into their structure. In MAX patches we have implicit temporal structures, since there is a temporal *line* from input objects via processing objects to output objects. Objects with common functionality favor modular structure, i.e. calling for sub-patches or super-objects that may be hidden away within a referring object opening the sub-patch in a separate window when needed. Practical hiding of detail is called *encapsulation*. It makes the patch/program easier to read and look at, but also easier to reuse elsewhere in the patch. Modularity and encapsulation simplify debugging too. For debugging purposes there are special objects like *trace* and *step,* very much like in other programming languages.

Programming methods for MAX should follow quite naturally from its quasi-OOP-approach. In CIM[278] we find a list of suggestions for good MAX methodology:

- Keep it simple
- Make it modular
- Make it readable
- Encapsulate and hide programming complexities
- Make it generalized and reusable
- Keep it consistent
- Make it foolproof (complete)

The Max/Opcode manual specifies recommendations for efficient programming especially, regarding memory efficiency. These low level considerations center around the choices of implementation methods and data structures in particular, most specifically the mentioned problems of timing and order of execution.

Good programming methods depend on programming style or general approach chosen (top-down/bottom-up/experimental e.g.). Considerations here are valid for most other programming idioms and paradigms, especially object-oriented programming languages, such as Smalltalk and others.

3.7 Composer objects and examples

We now turn again to descriptions of particular objects in MAX syntax. In CIM MAX objects are sorted and presented along categories of interface, listening, composing, sound design and score functionality. For machine composing purposes objects from all categories are needed, but composer objects constitute quite logically the core of compositional tools. We need interface objects for interactive functions, listening objects for analytic comprehension of input that composing patches rely on, sound design objects for expressive power of timbre (sound color) and score objects to make compositions conductable and understandable (transparent) for humans during and after performances. Compositions that make use of extra-musical information and expression (such as multimedia and lights) need specialized objects for those domains as well.

We looked above **[3.4]** at zero-patches, patches that let data flow through without intervention. Similarly, we can look at patches where something happens inside (processing patches) but where there is no interaction with a user. Such *autonomous composing* patches must use random generators for varying output, otherwise they will produce identical output (i.e. the same musical piece) every time. Autonomous patches are not really composing but rather playing or performing ready composed patches.[279] We will see that the most interesting results come from MC systems where humans and machines *share* or *intertwine* in decision making either in real time performances as partners or in compositional processes of *interactive composing* systems.[280]

Most objects of MAX are *doing* something, they constitute well-defined tasks, with parameters (inlets). They process incoming messages and their variables. Some objects are more data-oriented, others are more control-oriented, just as some messages may consist of only content (data), while others are instructions for control.

What objects are composer objects? Let us look at fugues as typical compositional forms in music. They consist of a main subject and a number of other (and sometimes) contrasting subjects that are repeated in time and space in varied versions. If a subject is repeated in another voice it may be identical to the first instance and is therefore a temporal variation or repetition. If a subject comes in a different key or slightly different shape or contour we have a transformed variation or repetition. We may use basic objects like *delay* and + for these simple operations. But fugues and most known musical forms are more elaborate schemes. Many formal constraints must be respected to satisfy formal definitions. Such constraints could be objects in MAX that qualify as composer objects. Delaying and transposing, as illustrated earlier, are elementary plans to manipulate data in a specified way.

In an interactive system a performer-composer influences a musical process arising from a conditioning patch with specific parameters controlling the levels and shapes of the musical outcome within defined ranges for scales, chords and time frames.

Composer objects or interactive systems as a whole respond according to Robert Rowe **[ch4]** basically in three ways to data or messages:

- Generative methods
- Sequenced techniques
- Transforming methods

Generative techniques
Using fundamental input material, *generative objects* or patches generate self-sustained and complete compositions. Using mathematical formulas or musical rules, these objects create structure and complexity from simple data. Examples are the *random* object of MAX and its derivatives like *RandomMelody*[281] that generates melodic structures inside a specified pitch range between min and max values (*RandomPitchRange).* Similar objects are *urn* and *drunk* that add further constraints producing melodies without duplicate pitches and limited melodic ranges ("*random walk*"[282]). Instead of blind or non-directed random operations one can also use e.g. fractal, chaotic (objects made of parts similar to the whole in some respect)[283], golden mean ratio and other mathematical ordered structures and schemes for extrapolation of new data from limited input material.[284]

Sequenced techniques
Sequenced techniques are based on prerecorded material and change aspects in response to real-time input. Melodic, rhythmic, dynamic or temporal variations are applied. The material is a determining factor. What lies in *rules* for generative techniques, lies in the *sequence* for sequenced techniques.

Transformative methods
Transformative methods are used to change musical material from any source. Transformative composer objects are variation generators. Arpeggio generators, pitch shifting, panning, filtering, limiting, scaling, quantizing and thinning are examples of transforming operations in MAX objects. An interesting transformative method is the application of *mapping* schemes where performer's actions are linked to unusual composer object parameters (Cypher, **ch4**). An example is a patch where volume values of notes are controlled by the duration of silence between preceding notes, i.e. longer pauses or staccato performance causes higher sounding notes and vice versa.

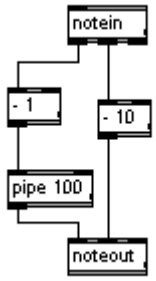
Many basic objects like *delay* and + are built-in class objects in MAX syntax. Other objects in shared libraries can be reused. But often require new musical ideas the definition and construction of new composer objects built from existing smaller task objects.

Some of the most complex standard composer objects are the sequencing objects *seq* and *detonate*. Seq objects record and trigger play back of entire streams of MIDI input. *Detonate* records on multiple tracks simultanously and displays the recorded material in a graphical editing window.

Some transformative objects may not change the structure of the outcome but add fluctuating or humanizing randomness to the superficial layer (expressive enrichment). This so-called "jitter" of continuous micro movement of parameters that are associated with organic qualities, typical of a human performance. Using constrained random values in very restricted scales "add life" to a sequence or composition. In a similar way MAX objects may follow human movements and translate or map them to musical structures applying a gestural interpretation. This seems to supply naturally expressive qualities as well (see *CaptureGesture* and *seq*[285]).

Example: From “note-amplifier” to the “spiritual echo-game”
We return to the echo-game mentioned earlier. It is implemented as MAX-patch that stores incoming note groups before sending them back again. Between reception and transmission notes are processed. Processing means in this case to changes pitches (transpose notes) and to delay the transmission. This involves several things: we must find note groups (listening), store note groups and make decisions about the method of transformation of notes and length of delay. A usual or natural given echo wouldn't be much of a game here because identical shape and delay of response to a question would mean the same thing every time (*Echo0*).Here are some more interesting variants of Echo with increasing processing load:
In **3.3** we looked at simple processing of note messages leading to a patch that amplified multiple aspects of incoming notes. In practical terms this was nothing more than adding '1' to each MIDI-value entering the “note-amplifier”-patch. **Add-one** is not really an impressing transformation, but it exemplifies how small changes in control functions can cause material changes. Adding one to MIDI-pitch-numbers makes an audible difference e.g. This is due to the relatively high abstraction level of MIDI representing high level structures (e.g.) by natural numbers. With MAX we construct functions or objects at even higher levels of abstraction (see *RandomMelody* above e.g). Let us continue with an echo **Subtract-one** (*Echo1*; **fig.3.8**) that transpose pitch one step or semi-tone down and in order to make the loudness perceptual relevant we multiply -1 with 10. Delaying numbers needs object *pipe.*

Figure 3.8: Echo1



We can now imagine all kind of combinations of different transposition and delay schemes. We can use a fixed transposition distance, fixed delay times, or varied distances and types for transposition and varied delay times. Variations may be random or semi-random using random ranges. Transpositions may start from the last-note's pitch or may reverse the whole note group (mirroring). As long as we output the notes with equal durations and pauses, we might speak of a *melotonic* echo (melodic implies rhythm). Here is an example called *echo2* or Melotonic echo that records tones played by the “game-player” inside 3 seconds and replays or echoes it 4 times, each time processed in the following way:

- random transposition value (from 0 to 10: *random 10*)
- loudness (or *velocity)* values decreasing by 20 each time *(100,80,60,40)*
- straight or reversed order user-defined by yellow button in the middle of the screen
- changing timbre (sound quality) each time

These definitions of the changing four echoes are both quasi-natural (decreasing loudness, number of times) and non-natural (transposing randomly and changing tone color).

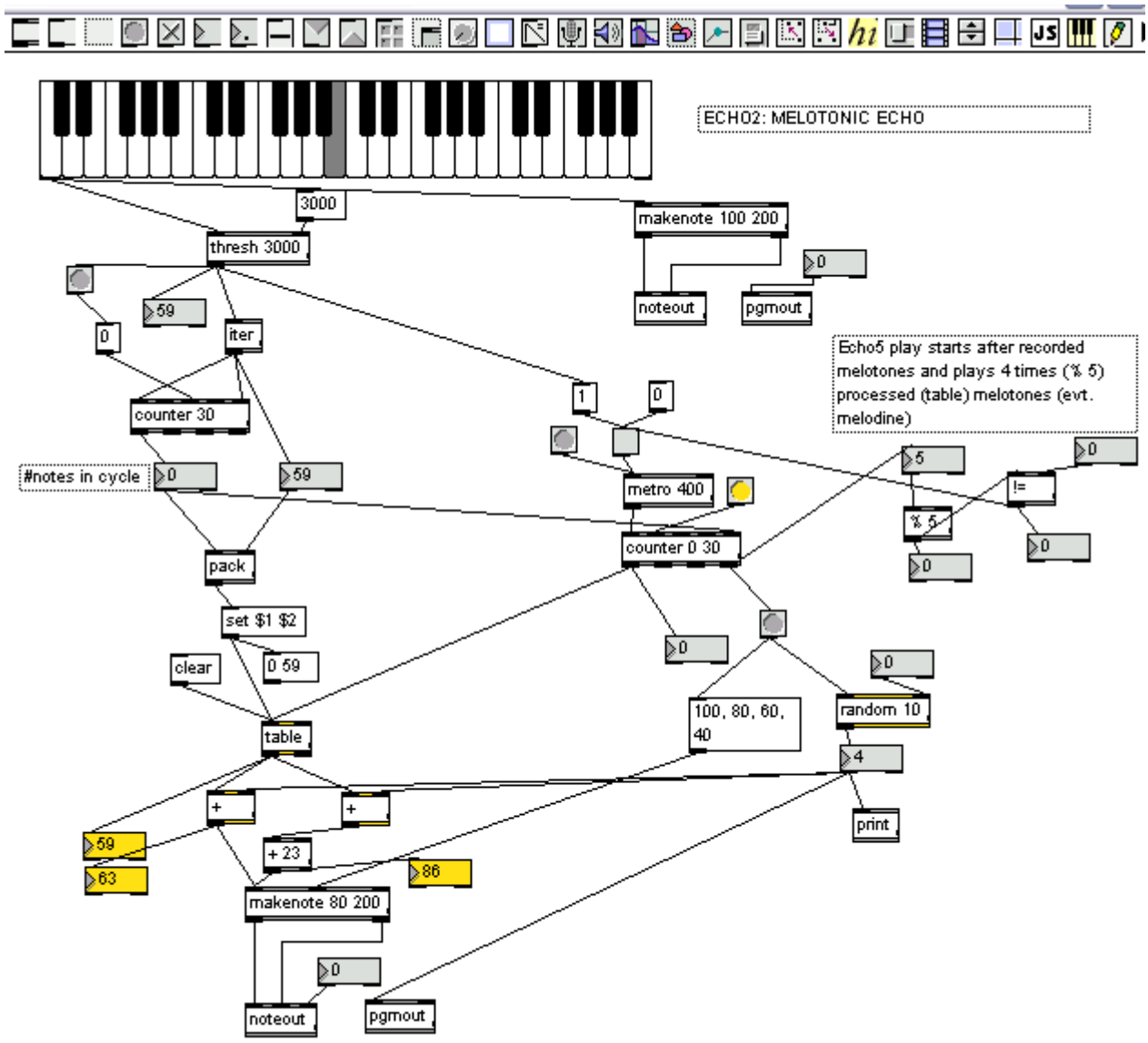


Figure 3.9: Echo2 or “Melotonic echo2”

The power of algorithmic thinking, applied in these examples of MAX, shows us how we can generate complex data out of rather simple input material and simple rules. The expressive power is applied very economically, i.e. small data combined with small algorithms result in surprisingly complex structures **[ch1]**.

Complex behavior generated from a handful variables is also demonstrated in the amazing experiments with *vehicles* of Braitenbergs “synthetic psychology”, where it is rather difficult to discover (or “reverse engineer”) the functional mechanisms in vehicles. A listener not made familiar with the components of Echo2/MAX will similarly have difficulties to find the essential working of echoes in MAX. Somehow analogously to the series of generated numbers by functions. It is usually difficult or impossible to discover the generating mathematical function,[286] even when listening to long streams of output. We will see how such things might be done in a more tentative way in **ch5**.

But Echo2 illustrates another interesting fact. We used composer objects with higher complexity than the pure arithmetic objects used in Echo1. Still, Echo2 does not make use of any musical

features of the input. We better use objects that find underlying musical structure (scale, rhythm) and transpose and delay accordingly. This would presuppose finding a referential rhythmic grid and fundamental melodic scheme or scale. Such echoes would be easier to trace back to their computational source. That is because scale- and rhythm-preserving transformation has *musical meaning*. As human listeners we understand the musical logic built into such composer objects. Composer objects that model human musicality or conventions are therefore decoded by an intentional logic.

Rhythm and scale are high-level concepts that presuppose contextual analysis and a certain musical expertise or musicality. This will be further explored in **ch4** and **ch5**.

3.8 Max/MSP, PD, and jMax: dialects of MAX

MAX – the paradigm

The creator of the original Patcher (early Max), Miller Puckette describes in "Max at Seventeen" (2002) the MAX paradigm and its history and development into current MAX-based languages or implementations of the MAX-paradigm (Max/MSP, Jmax, Pd). The paradigm and key ideas were the result of problems with limited processing power and non-mature synthesizer technology. MAX is mostly a continuation of software from RTSKED by Max Mathews that controlled polyphonic synthesizers with real-time scheduling, using wait functions and triggering. While earlier systems constrained users or crypto-composers to trigger whole sequences of actions, users of RTSKED/Max could now act more freely within an instrument or keyboard approach. This presupposed multitasking, "depicted" by boxes displaying parallel processes on screen. Even if Max was not conceived with the later developed MIDI-protocols in mind, the fusion of these two key technologies of the '80s was perfectly logical. Also other influences were present, such as the work of Barry Vercoe with MusicN/Csound, doing e.g. real-time score-following on 4X machines at IRCAM. MAX can therefore be seen as a necessary evolutionary step towards more flexible and integrated systems of controlling and synthesizing technologies at the time.[287]
All MAX implementations of today use GUIs. Back in the 80s this was no matter-of-course. But whether this property should be part of the paradigm is not obvious. All implementations permit saving patches as text-files, so GUI seems not an essential feature after all. But when Puckette in 1987 integrated a visual patch language in IRCAMs MAX versions it was also part of a general trend in user-friendly computing (e.g. Macintosh). But because MAX is meant to make music programming more musically we may include this feature in the MAX paradigm. Visually patching was modeled after physical patching, the very much non-symbolic programming of synthesizer modules in the '60s and '70s.

Max/FTS to jMax

As we see from **figure 3.10**, Max/FTS originates from the Patcher, just as the commercial Max/Opcode. Opcode and Zicaralli were concerned about the general public applicability of Max, while researchers at IRCAM had all resources for maximum performance on state-of-the-art hardware. Interest for both control part and synthesizing part led to the use of separate platforms for control + GUI (NeXT) and real-time sound generation on the fly/FTS:Faster Than Sound on a special purpose signal processing machine (ISPW). When NeXT inc. failed, IRCAM had to transfer software again to a Unix/X system (Dechelle). This version, called jMax, is programmed in JAVA and distributed under open source.

Max/IRCAM to Max/MSP and Jitter

Hardware-specific control-applications (4X, ISPW, NeXT) were finally made platform-independent using the inherent abstraction in MIDI. IRCAM distributed versions of "Max 1.0" in 1987 for

internal use only. David Zicarelli brought the academic version (Patcher, 1986) to the commercial domain (*opcode.com/cycling74.com*). Max/Opcode was strictly a control language only, without any signal synthesis or processing objects. Ten years later did Max/MSP catched up. But Max/Opcode increased numbers of objects and tools with every new version.
Recent versions of Max/MSP supply sound synthesizing and processing functionality comparable to FTS/jMax. With multimedia becoming commercially interesting, add-on modules like Jitter provide real-time-video, 3-D and matrix-processing. Max/MSP is now a full-blown control language for sounds, light staging, picture processing, video performances, dance performances. Earlier Max versions were Macintosh-only. Recent versions work on Windows-platforms as well.

PureData (Pd)
Puckette worked from 1994 (USCD) with a new MAX dialect, built from scratch to optimize and improve with the background of earlier experience. Pd was finally bringing MAX to the public domain. Pd was open-source from the beginning and runs on MacOS, Windows and Linux. It includes objects for audio signals.

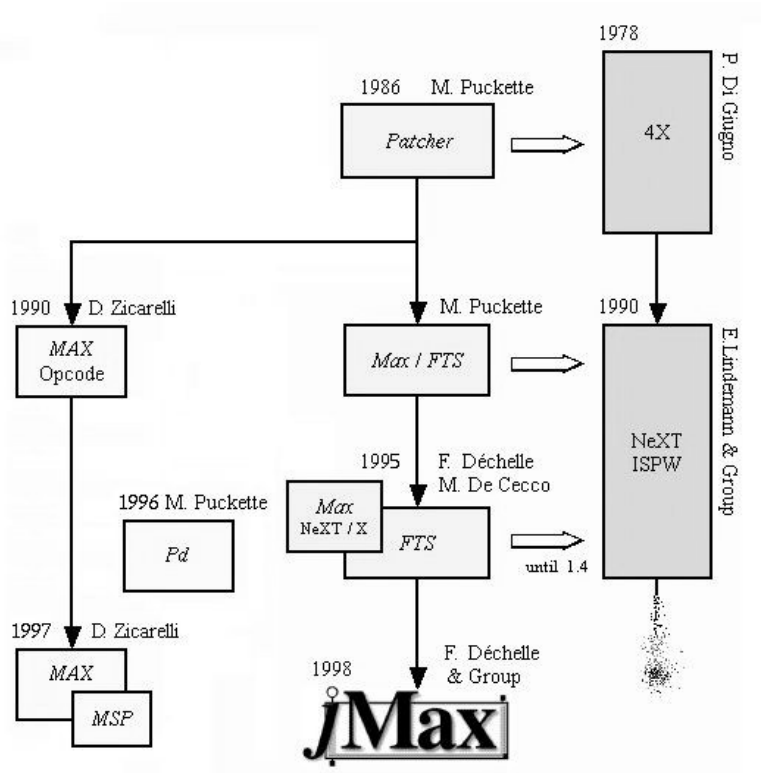


Figure 3.10: "The history of MAX"
*(*Puckette,*Max at Seventeen,* 2002*)*

Why several MAX-like languages?
There are several reasons for the divergent directions of the at least three MAX-like applications from the Patcher in 1986. First of all, there is the driving factor of hardware (discontinuity and multiplying powers). Developers bet on different scenarios. Another factor is obviously the cultural difference or the social fabric of academic research (at IRCAM and other laboratories, especially in the '80s and '90s) and the commercial market realities. While Max/FTS gave priority to high functionality and flexibility, Max/Opcode waited with audio processing until machines in the mass marked reached satisfying levels of performance. Last but not least, the personalities of the involved pioneers were little compatible with standardizing and compromising on behalf of their rather small sized user groups. Perhaps an integration between jMax and Pd would have been possible (both versions are open source), but jMax is modular designed with individually loadable packages, while

Pd is not. Music instrument corporations agreed in 1984 on a common protocol for synthesizer control messages (MIDI 1.0 to 2.0). This did not happen for the MAX paradigm. Nonetheless, user communities in all three camps continuously supply new modules and libraries as extensions to their respective core systems. Many libraries are doubled in the several MAX dialects. Therefore one can often rewrite special code in another MAX dialect. Since the MAX paradigm is common to all, patches or applications exported as text files are also readable and understandable for others. Some design issues, though, are solved differently, e.g. scheduling problems treated below are solved "graphically" in Max/MSP, i.e. actual screen positions of objects define order of execution, while Pd and jMax use syntactical methods (explicit order specifying trigger objects) to avoid ambiguity.

3.9 Conclusion: MAX and machine composition

We looked at converging and diverging forces in the evolution of MAX. The paradigm of MAX is a stabilizing idea (perhaps somehow like a species). Different MAX implementations or dialects push in opposite directions (maybe like mutants). The grand idea of MAX is to facilitate composing by machines. MAX users *define* patches. Are MAX-patches *compositions* or *instruments*? Is MAX a composer, an instrument maker or a programming language? There are no obvious answers here. MAX can be used in various ways ranging from programming, scheduling, synthesizing, sound-editing, instrument-defining, composing and perhaps programming of composers. This flexibility is sometimes a vice and other times a virtue. We may look at MAX as being like a "swiss knife" of musical tools. The same, arguments about whether MAX should be considered a "normal" programming language shed some lights about this issue and the character of MAX.

Real-time scheduling and task communication or "real programming language"?
Since designs/implementations of MAX disrespect many of the rules of computer science text books, some have claimed that MAX is not really a programming language. Puckette himself partially accommodates this view in "Max at Seventeen"[288] and calls MAX more a system for *scheduling and communication of real-time tasks*. Arguments for this view come from the following properties of MAX:

- it is more oriented towards processes than data.
- it doesn't enforce hierarchical system building, favoring instead formal plasticity (e.g. is there no built-in notion of a musical score or similar static levels of representation).
- it lacks a uniformity of approach in many ways, e.g. objects are free to store data in their own defined way or they have more behaviors/tasks than inlets (intransparent black box). All messages start internally with symbols/selectors, but in many boxes these headers are omitted; so called hidden selectors ([3 5 3 2] => [list 3 5 3 2]).
- it doesn't provide systematic (bigger size) data handling facilities, even if later systems aloud for viewing and editing in special purpose sub-windows of data structures like *table, qlists* etc.
- it combines heterogeneous strategies and data species (control and signal/audio) inside a common framework without clearly separating them (patching audio signals between boxes departs radically from the control scheme, i.e. conversion from audio signals to messages is problematic and schedulers have to treat these processes basically different leading to scheduling problems. When timing precision matters, data size of messages will be a factor. The two worlds of control and audio processing coexist somehow in patches, what offends rules of computer science.[289]
  On the other end of this spectrum: messages can be pure triggers, i.e. completely empty of data (bang).
- its limited use of GUI functionalities, e.g. different object types are not viewed as different icons, but users must enter names of boxes as text.

- its limited scoping and namespaces of varibles, symbols and their bindings are non-hierarchical or flat space. Objects have non-persistent states, i.e. whenever patches are reopened values are defaulted.
- its limited control flow syntax lacking conditionals, loops and subroutine structures, mostly as a consequence of graphical interface.
- its limited computational determinism: results of patches are not always easy to predict, e.g. when decision making in a patch depends on audio processing (of samples), there may arise computing-time depending competition with other triggering messages (from control objects). Such contexts are sometimes hard to control and predict.
  Scheduling problems generally increase with graphics and video objects involved.

Between data flow and OOP semantics, a non-complying style in between?

MAX does not follow OOP standards as exemplified in OOP-prototypical Smalltalk, even if messages are passed through patches. There is first of all no inheritance (and hence hierarchy) of data and functions as we know it from Simula and other OOP-languages[290]. Neither is MAX a data flow programming language even though large portions of data (sound and control) "flow" through patches in defined and logical ways. That MAX (as paradigm) does not belong to a computational school has multiple reasons. The fact that MAX was mainly developed during the1980s explains some ad hoc hacking mentality, also in relation to idiomatic hardware realities, that were operating in MAX developments. More importantly, the creators of MAX deliberately favored the perspective and needs of their "creative users" rather than that of "structured engineers". The focus on musical constructs, as opposed to more logical constructs known from computer music circles, led to a certain structural fluidity; thereby quite consciously avoiding stylistic bias in the users' products.

> "Style is important in software: not so much the internal style of programming, but the style with which the software engages the user. We welcome software if its external style pleases us. Well-designed software enhances the workspace in the same way that well-designed furniture does, not only in functionality but also in the stylistic choices that enhance, and don't depress, the quality of our environment."[291]

Actually, structural shortcomings and non-uniformity does according to Puckette (perhaps proven by the huge user community) turn out to be a virtue after all:

- process-oriented approach encourages dynamic and creative structures.
- non-hierarchical structures and multi-principled thinking aloud an atmosphere of experimental and open mentality (interpretive environment).
- non-systematic syntactic resources and possibilities speed up programming of contexts and processes without planning whole structures from the outset.
- non-complete representation of processes in patch windows permits display of more complex contexts in one window that should provide full documentation of what's happening. Using hidden selectors, incomplete view of task repertoir of objects, idiosyncratic ways of storing data in objects and similar shortcuts open up for more paths for realizing ideas fast and tentatively. It contributes to ease inter-connections by reducing the "amount of glue needed".[292]
- text-based object-boxes lower the threshold between system-defined objects and user-defined objects, thereby inspiring to a higher degree of user adaptation of MAX in use. MAX windows are usually opened as blank windows, infusing ideas of starting from scratch, what amounts to creativity-friendly surroundings.
- unification of data flow model with message passing model is suggested by an instrument metaphor, for example with a piano information about timing/triggering (on-off) is intrinsically paired with information about sound quality (like timbre, velocity and velocity curve/envelope). Since each model has strengths and weaknesses (data flow excelling in description/message passing in precision of timing information, a combination seems both natural for interactive music composition and problematic from a principled perspective. An instrument metaphor means that objects do not

> "know" what they wait for, message decisions are always made by the player or sending object exclusively. Decisions and their vehicles of propagation (messages) pass downwards only, i.e. objects are acting actively and listening passively. This idea is fits nicely in our context and is well represented by graphical patches. The order of message execution may not always be perfectly clear though.[293]

- even if MAX may not be a programming environment in the sense of allowing strong structural definitions, classical structured algorithms can still be added from within objects that import entire interpreters such as C, C++, Scheme or Lisp (maxlisp,**ch8**).

In his "anniversary article" ("Max at Seventeen"), Puckette is widening the perspective and includes arguments for democratization and cultural autonomy in his wish for future MAX to become a tool ever more neutral, both stylistically and culturally. His own contribution for open-source Pd underlines this interpretation of his.[294]

Puckette defends autonomy of MAX in regards to certain dissonances with computer science **[1.8]** and rules for software design and implementation in a rather confronting style:

> "... sometimes annoy the computer science crowd, the failing is a lack of understanding of the importance of style and even aesthetics in software design and implementation. Computer science has never found a metric for determining whether or not a computer program is fun to use."[295]

After all, the future of MAX seems bright today, but only prodigious use in the future will confirm the significance of the presented arguments and dialog.

We will see in the following chapters how different approaches of adding and reducing content induce different types of composing patches. In **ch4** we look at *positive patches* adding content. In inductive MC of **ch5** *negative patches* are favored in the extraction or search for structure in referent sets of stylistic examples.

In the next chapter we will look at how musical compositions can derive from machines using computational algorithms guided by an implementation of high level musical knowledge in an integrated system: informed machine composition.

Chapter 4:
Knowledge and laws: informed Machine Composition

Prototypical Cypher and other Machine Composition systems

1. Introduction
2. Types of Machine Composition systems
3. What is music knowledge? How does it relate and lead to musical intelligence?
4. Cyphers parents: Music Mouse, Jam Factory and M
5. Cypher: an overview
6. Other examples of informed systems in Machine Composition
7. Representation and methods of AI and Machine composition systems (similarities and idiosyncrasies)
8. Conclusion: knowledge and laws in informed Machine Composition

Machine Musicianship is the technology of implementing
music concepts in computer programs (Robert Rowe, 2001[296])

Machine Composition fleshes out computer music
through the informed nature of its algorithms (page 4)

4.1. Introduction

In **ch2** we started up with some observations about general phenomena around natural intelligence in relation to music or simply musical intelligence. Humans call musicality the state or capacity of being musical receptive and responsive in a musically qualified way. Music teachers support learning processes, developing musicality by a combination of method-guided instruction and immersing teaching practices.[297] We now address the subject of teaching machines musicianship or how to design and program musicality in machines.

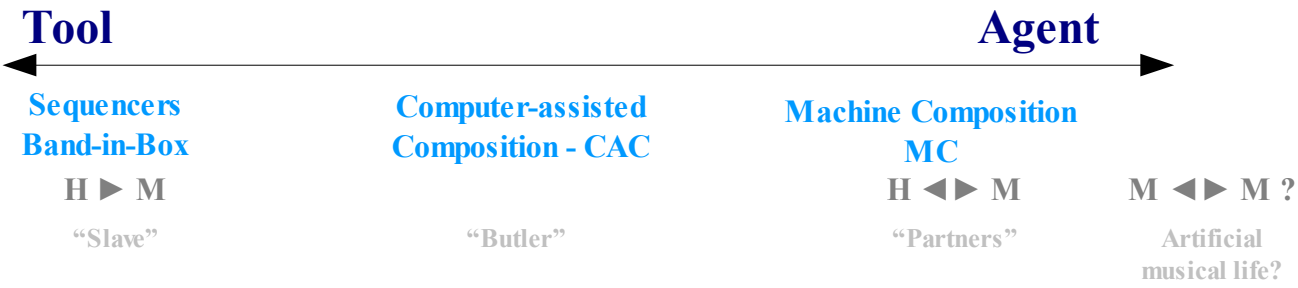
We need to build musicality or musicianship into the "soft-ware" of machines which demonstrate their musical intelligence by doing musical tasks in a musically meaningful way. How do we implement musicianship in machines? And what computational approaches should be endorsed in respect to music analysis, performance, improvisation and composition to achieve such ends?

We know from AI and its applications in non-musical domains that many methodological directions are pursued in parallel. A main division goes between the so-called *symbolic* AI *(*LS*)* and sub-symbolic and connectionist i.e. parallel distributed systems (in the following subsumed as *artificial neural network* approaches in AI, ANN[298]).

Even if the idea of ANN or more abstract distributed systems goes back to at least the 1950s, systems of ANN were mostly experimental and not as productive as LS until around 1990. As we briefly mentioned in the last chapter, the difference lies basically in the *level* at which a system learns, conceptualizes and stores information. LS works with formulations on the level of symbols, such as natural language concepts and uses knowledge that is transferred to computers through a logical apparatus. ANN do by definition not understand human concepts at all. ANN build its own subsymbolic "concepts" to relate (typical problem) input to (typical solution) output, and learn them by association of situations. Human concepts are therefore not localized in specific neurons, but *irreducibly distributed* over many neurons or even the whole network. To describe ANN in detail we have to use mathematical models. Learning in ANN is a low-level activity with situations fed into the network. Actions are sanctioned during a process of *tuning* and *training*. In LS systems we

would rather speak of *instructing* or *programming* because one reformulates logically structured knowledge that is on a level with knowledge of human experts (compatible with what humans know).
As a rule, intelligent activities that are taught or instructed are first candidates for LS implementations (because formulations of symbolic knowledge already exist). On the other hand activities that are more trained than instructed (such as tennis) are probably more natural to solve by ANN. The challenge in choosing the right paradigm for specific AI problems will also apply to music systems that emulate musicality or musicianship in the digital field. Nonetheless, we must always consider the possibility that non-formal and trained tasks may be successfully reconstructed on the symbolic level and vice versa.

We saw in the learning cycle of music [ch2], that musical activities or learning stages are connected and dependent on each other both practically and theoretically. Extending music learning to musical machines, we need human-machine-relations (H►M), where humans program and guide machines, and machine-human-relations (M►H) where humans react and play with machines. In a more distant future we may even imagine exclusive machine-contexts as well, i.e. machines that program and play with each other without any human interference (M►M). Central to our subject is a distinction between predominantly compositional *assistants* (H►M) and more autonomous composers or *agent* based machine composition (M►H), see figure below.



This model or dimension is related to the way one understands 'interactivity' within MC. Machines that are performance-oriented and react to real-time input from a human performer are *improvisational interactive systems*. Machines that are score-oriented and conducted by intra-generated material and in addition to material supplied by humans are *compositional interactive systems*.

What is the meaning of an *interactive system*? Actually, any system of value will be interactive to some degree. An "inter-passive" system is an automaton that uses no input. Such an autistic or solipsistic system is probably of little musical interest, since music by nature is linked and enriched by communication'.[299] Let us think of an automaton that varies musical subjects continuously, computing (but not creating) progressively complex results without any human interference. Such sound installations are known from avant-garde museums. Are they machine composers? Certainly these are not improvising machines in the full sense of the learning cycle. Some kind of real-time interaction with humans is a precondition for machine composing systems.[300] In other words machine composing systems are systems where humans and machines interact at varying degrees of participation (see figure above). In AIMA we find the following definition of an agent:

> An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors.[301]

AI is according to AIMA the enterprise to design successful agents that do a good job of acting on their environment.[302] Artificial musical intelligence seems to presuppose environments that at least

include human musical agents. A machine composer acts in an environment where humans and machines in cooperation find aesthetically satisfying solutions in musical design. Therefore we establish in analogy to AI the following conditions for musical agents or agents in MC[303] :

- The performance measure that defines the degree of success.
- The agent's complete perceptual history (percept sequence).
- The agent's knowledge of the environment.
- The actions or expressions that the agent can perform.

The second and third conditions are often represented and solved symbolically. They constitute the explicit formulation of knowledge that makes up a specialized musical agent. It presupposes a computational approach and hence reformulation and adaptation of music theory and analysis.

4.2. Types of MC systems

Rowe classified[304] interactive systems relative to several dimensions **[ch3]**. He distinguishes score- from performance-driven systems, instrument- from player-systems and finally identifies possible response methods as generative, transformative and sequenced techniques. In this way he characterizes systems in a multi-dimensional metrical system where systems are defined by their positions relative to these dimensions. I believe these dimensions have common traits and may be reduced to one meta-dimension with triggering on one end and generating on the other end of an axis.[305]
What Rowe classifies as sequenced technique is essentially the starting and varying of received complex structures or sequences. The instrument paradigm and score-driven systems also belong to the triggering side. We find performance-driven systems, player paradigm and generative methods on the other side (transforming simple input material i.e. generative method) or listening to real-time material (e.g. performance-driven player) and responding by generating complex new structures from it.

The resulting axis or dimension distinguishes between triggering-type systems (T-MC) that perform more like tools or MC assistants[306] and generating-type systems (G-MC) that are more autonomous machine composers or agents. T-MC are less pro-active but more predictable. G-MC are more pro-active but less predictable. T-MC behave more like an orchestra (score-driven), instrument or accompanist (score-following[307]). G-MC is more like an improvisational partner or composer (generative). But both types of compositional systems transform material at varying degrees and with different methods.
T-MC uses more structured and bigger-sized input for data manipulation and responding. G-MC on the other hand takes rather small bits of input for data construction and generation.[308]
Generative systems lean towards relative autonomy and are typically in computer music (see above). Even if computer music and machine composition by definition are algorithmic **[ch1]**, generative systems rely more substantially on algorithms for de-constructing and reconstructing. Algorithms used in triggering applications focus more on time-ordering coordination or conducting

of material (predominantly scores and sequences).

Pure algorithms are evidently not goals but means for meaningful musical activities with computers. Even algorithms in 'computer music', how abstract and mathematical they may appear to their opponents, stand for more than formal structures only. MC in contrast seeks to establish a platform of *effective* algorithms capable of *musical expression* after computational reformulations of established music theory and analysis. "MC fleshes out computer music*"* through the *informed* nature of its algorithms. MC builds on functional architecture of music knowledge with *representational content*. Once formulated, musical knowledge can be reused in other forms of automated and autonomous composers/players that may embody multitudes of musically intelligent styles. MC agents interact with human co-composers in a process of intertwined evaluation and proposals in relation to co-generated structures. Concluding this preliminary description of different types of MC, we may observe how similar composition and improvisation fit into this picture. They are not different in kind and they share a lot of descriptions. Both are interactive and autonomous to some degree. Both respond and act freely (tools vs. agents), only in different time-scales. Improvisation is real-time fast composition while composition is slow and reflecting improvisation. We could compare these practices to playing speed-chess versus chess or writing versus playing theater plays. Does this reflect the division between LS and ANN in AI?

4.3 What is musical knowledge? How does it relate and lead to musical intelligence?

Agents that know about their world and reason about their possible courses of action are knowledge-based agents. If our agent is supposed to com-pose music it must form representations of its world based on musical input and use a process of inference to derive new representations that enable valid musical output. We saw in **ch2** that the fundamental questions of choosing and using appropriate representations are questions that both require a solution for natural and artificial agents. Let us start with a formulation from LS(AIMA), AI's symbolic approach:

> A knowledge-based agent needs to know many things: the current state of the world; how to infer unseen properties of the world, how the world evolves over time, what it wants to achieve; and what its own actions do in various circumstances.[309]

We can distinguish between *factual* knowledge about input, *inferred* knowledge about structure, *intentional* knowledge about the aesthetic plan for the action or output and *background* knowledge about the listeners preconditions for music understanding.
These parts of musical knowledge in MC must be formulated first at the knowledge or *epistemological* level (ie. where it is first found, see AIMA), then encoded into sentences at the *logical* level and finally represented at machine or *implementation* level. For example could knowledge about the 'circle of fifths' [310] ("dominants are resolved to their tonic") be reformulated logically (V->I), functionally (cof[step]:[if [step = 5] then 1 else nil]) or in quasi-code.[311]

The current state would be the facts or descriptions of musical input and listeners cognitive preconditions for understanding. Inferences to non-surface properties of musical input would include higher level descriptions such as scales, chords, melodies, harmonies and keys. These virtual properties would necessarily involve computational reformulations of music theory and analysis. Inferences to possible future events will in improvisational MCs be necessary to respond to *during* playing. How the world (or dynamic environments) evolves over time is more relevant for triggering or accompanying systems. But even in generating MC systems world descriptions may stand for the totality of musical works at that moment.[312] What it wants to achieve is a plan or idea of a high level transformation from input to output, i.e. a compositional sketch or idea. What its

actions or musical reactions are in various circumstances, is part of its plan to build broadly consistent and intentionally successful sound constructs from actual sound input.
A hierarchy of *knowledge representation languages* describe events in highly abstract terms down to the very concrete phenomenal level.

We now try to describe the informed character of MC in more practical terms. Certainly our example of a "spirtual" or melotonic echo in **ch3** is evidently non-informed. It only transforms input conforming to arithmetic rules. It is nothing more than an imitating and moderately responding echo (alas 'echo') without will or musical intelligence notwithstanding a pretentious nickname. The reason why some listeners could assume some spirit behind its curtain is the computational power of random operators combined with cliché-like behavior **[ch1]**. Quite like ELIZAs fooling conversational victims to believe in her power to understand their assertions. To transpose a melody (one half-tone higher (+1) or more) proves uninformed; contrary, to the transposition of a melody one tone higher relative to an assumed *scale* presupposes informedness because it carries over information about *contexts* and *meaningful representations* of musical *content*. *Informed MC* means therefore

> machine composition *informed by music theory and analysis* or simply *music knowledge*.

Since we in this essay distinguish between 'computer music' as uninformed composition and MC as informed composition, the above clarification is pedagogical only.

Music *theory*[313] is the systematic study of musical elements and abstract principles embodied in music as found in music teaching. Music *analysis* works the other way around, splitting singular works or collections of works (styles) into component parts and relations[314]. Essentially this is the reverse process of com-posing or putting together component parts into compositions **[ch1]**. Music theory and analysis (musicology) are somehow analogous to positive and negative patches in MAX presented in **ch3**.

Musical theories in Western culture (WMT) have been explicitly formulated (and often written down) for around three thousands of years. They describe basic properties of single sounds, like pitch, duration and timbre, and collections of sounds like acoustics, tuning, temperament, intervals, consonants, dissonants, chord, meter, rhythm, form and organization.

A recurrent theme in theorizing about music is the pursuit for balance and imbalance between vertical and horizontal events and forces. Pitch and time, designate the two dimensions of quality and succession, dichotomies we find equally in intervals/chords/timbre vs. melody, meter, rhythm and form.

This is usually called the tonal system in WMT, a specific tone or pitch organization in time structures. Like different language systems or logical systems are characterized by a particular syntax or grammar, WMT accepts only semi-tones as atomic pitches derived from dividing an octave-interval into twelve basic pitch-classes (pc). Just as there are no letters between a and b or x and y (or natural numbers between 0 and 1), micro-tones, i e intervals smaller than semi-tones, are only allowed outside of tonal music. The tonal system then defines rules for valid compound structures like scales and chords. Earlier modal pitch systems actually included a higher number of rules for well-formed structures thereby widening the space of potential, legal structures.[315]

A *scale* is a sequence of atomic pitch-classes (pc's) or interval-relations, depending on the choice of representation. In tonal music, only major and minor[316] scales are allowed. Many chords are valid structures, dependent on their function relative to their surroundings or tonal environments. Still,

tonal music builds on and exploits the defining power of triads in chords (vertical intervals with semi-tone distance of 4 or 5) as some kind of markup for higher-level-structure. This preference for triads derives from the role they have in determining scales (reducing ambiguity), root tones and especially key, one of WMT's central properties in interpreting structure and content in larger contexts.

Key or tonal center stands for the specific scale (starting from a specific pitch-class) that is needed to understand the underlying, global structure. The key is supposed to govern or organize overall structure and content in musical works, contributing to the coherence in an "organic system". We will return to other high-level-features of works later. Understanding key and categories such as thematic material, melodies and form in relation to key (*C* in C-major) and scale (*major* in C-major), are crucial for making sense of WMTs hierarchical tradition. Just like modal cadences underpinned the modal intuition of medieval music listeners, tonal cadences and other key-supporting conventions make up our days musical intuition. Because of this complicated tonal make-up, both music theory and AI must collect and rely on extra-musical knowledge from psychology and especially cognition. Tonal theory has been expressed for centuries in elaborated teaching treatises. From as early as at least Pythagoras we find psychological explanations related to sound and music reception.[317] For instance, treatises in the period of baroque with their quasi-deterministic systems of 'figures' and 'affects' in music and correspondence to human emotions continue this long tradition. Romanticism's affection for personal expressions through art and especially music (Schopenhauer a.o.) is another example of the necessary connection between music theory and sciences of emotions.

Theories about attractions and tensions in music (and organisms in general)
Many theorists saw the inherent difficulties in the competing forces of attractions and tensions in vertical and horizontal directions. Experience of motion in time collects energy from both scales (horizontal forces) and chords (vertical forces), as well as their perceived key(s) and form(s). The most apparent force herein is the main motor of tonal direction: the *descending fifth* (called dominant – tonic or V-I in tonal functional terms). The emerging 'cycle of fifths' was in baroque applied as a successful formal principle (see Corelli citation). Schönberg held around 1900 the view that tonal music's expressive space had been used up, blocking any harmonic innovation. This gave him a reason for developing a new tone system, an effort he himself saw as a logical continuation of tonal theory and historical determinacy. His new 'serial tonality' used the whole chromatic scale without discriminations thereby weakening established feelings of tonal centers or gravity. Ironically, many practitioners in 'computer music' were attracted to this new sense of freedom from attractions.

But tonal music lives on today. We seem to have this "*special* attraction" to it. Music theorist Meyer furnished a psychological colored theory about the power of tonal music. It was compatible with Dewey's conflict theory of emotion[318] where ignorance (or especially interpreted as lack of control) arouses strong mental drives towards clarification (or solutions for survival). Musical suspenses or tensions (dissonances) frustrate anticipations (related to melodic contour, chords, key and formal unity) and arouse expectations of musical relaxation. Forces of attraction (expectation) and tension (frustration) are integrated into a wider psychological and cognitive model. At the same time Meyer wished to give an explicit description of meaning in music, in a style that earlier only provided syntactical grammars for tonal discourse.[319]

Research into the tonal system led later to the classic theory of GTTM (Generative theory of tonal music) by Lerdahl[320] and linguist Jackendoff. It is a correlate, if not application, of generative grammars (Chomsky a.o). Both theories of structure and meaning rely on certain generative principles for production of surface structures and shapes. Such grammar-inspired theories, similar

to classical Schenkerian analysis, assume principles of order at a higher placed hierarchical level than is necessary in Meyer's approach of finding of patterns. These generative theories seem powerful and share with computational algorithms a vast expressive rooms of possible constructs. Nevertheless, a not too obvious separation of constraints into *well-formedness-rules* and *preference rules* and somehow suboptimal results for non-paradigmatic or strange samples remain a source for malcontent. Recently, several theories in family with both GTTM and Meyer's 'Emotion and meaning' apply geometrical models that incorporate multiple features or dimensions *in parallel* using powerful mathematical apparatus. Lerdahl in Tonal Pitch Space[321] and the Spiral Array hierarchical model (SAH) is a geometric representation of tonality with spatial analogues of physical and psychological phenomena of various kinds. Interestingly, newer theories rely more on geometric thought than syntactic or logical formulation, in part to attack the problems of parallel causation of their underlying complex phenomena (see also Gärdenfors on concepts, and Petitot/Godøy on meaning and reductionism). It is far from accidental that we dwelt on various theories of tonal music to elucidate the nature of music theory. Tonality is by far the most complex, high-level and unifying property of musical works in Western music. But it also demonstrates the difficult nature of explaining and understanding bits and pieces of culture without falling prey to greedy reductionist methods and hence limited results. Dilemmas of data representation were presented in **ch2**. We saw that every type of syntactic representation (MIDI-notes, sounds, notes, chords etc) excludes and includes different perspectives and features. Dealing with meaning in music we encounter related problems of parallel influence of a multitude of factors. These problems are already well established from music analysis.[322] Models with static structures in the analysis will have early shortcomings due to the dynamic and immanent nature of time and memory factors in music reception and its interpretation (musical understanding).

Implementing such knowledge complexities into MC models seems therefore to be very demanding. However, there are reasons to be cautiously optimistic. Complicated and integrated theories will as machine models emulate them experimentally. MC models might become an effective arena for testing theories by piecemeal support or rejection of hypotheses (just like other scientific theories are falsified and confirmed). Especially theories of Machine Listening can be tested empirically against cognitive and musical background theories before applying them ultimately in MC systems.

Computational models of theoretical components (features of music theory)
Representational schemes try to stabilize and conform objects in a grid of time and space. Ojects are necessary from the very bottom up of all theories. Theoretical components or basic features of music theory and compositional method must be cut out of the *immediate* sound space or the *sensed continuum*, manifested as discontinuous sound “energy” in the acoustic representations (see **ch2** for representation and semiotic aspects regarding sender/receiver).

Here is a preliminary proposal for relevant representational *levels*:

- Sound space (unsegmented and not parsed).
- Sound spectra (segmented? by acoustic and non-semantic criteria).
- Notes as sound objects are represented symbolically in scores (symbolic representations).
- Midi-events represented as MIDI-bytes contained in MIDI-files (machine representations).
- Pitch, duration and timbre description (classification) of *single* MIDI-events/notes/sound objects, i.e. with a context size of one (neighborhood=**1**).
- Calculated delta-times and intervals between two events, i.e. with a context size of two (neighborhood=**2**).
- Multiple notes segmented/grouped into chords, roots and scales (neighborhood **>2**).
- Melodies and chord progressions (key) in horizontal contexts (neighborhood **>> 2**).

- Semantic subjects like harmonic cadences and melodic ornaments.
- Formal and prolongational (time-developing) structure or content of complete movements, or successions of movements, i.e. entire works. For example AA'BCB'A"A.
- Multimodal environments (i.e. integration with non-sound, e.g. physical movement), that include contextual full-body movement information and control (gestalt approach).

Continuing the issue further and extracting properties from groups of works of a single composer[323] an epoch or universal elements of music from an unlimited musical data set, we create abstract style descriptions. Identifications of style or genre require discovery of likenesses in sets of works. Such relative similarities express styles or genres which is subject of **ch5**.

Data representation has to start with forming objects from an experiential flux (*segmentation* or *discretization*). We find objects (or make concepts) at every level of formalization. We must delimit a sound or note from the next by certain criteria. How long should a silence be before it qualifies as borderline between two sounds or notes? Or how far can a pitch deviate from its "fundamental" pitch before breaking the threshold for the next semi-tone (evt. micro-tone)?

And should we consider pauses between notes as part of the first notes' duration time (*delta-duration*) or not, being very much relevant for phrasing in performance? And particularly, when does a tonal quality (timbre) change enough to become a different sound-type (e.g. *frullato* or *pizzicato*), notwithstanding the continuous transitions between them?
I abstract, for reasons I outlined in **ch2**, from all these legitimate aspects and use only sounds with perfect pitches and durations. This ideal and abstract "note-world" is nicely taken care of by the MIDI formalization language. Pitches 'are' integers, pitch-classes 'are' dividable by 12, and chords 'are' sets or lists of integers. Now we try to present systematically pitch relations in a collection of MIDI integers.

Key induction may serve as a convenient illustration. Key induction[324] is the task of identifying the tonic of a set of chords in a tonal framework of functional harmony.[325] To induce the key of a time span we need to know chord types *and* roots (fundamental chordal tone). We start with some simplifying assumptions: chords consist of three or four notes without repeated pc (triads or four-note-chords). In classical music fundamental chords are triads, in jazz there are mostly four-note-chords. Member notes are described by their pitch class (pc) alone i.e. all eight C's on the piano belong to one and same pitch-class C. This means we represent at some middle level, since many actual, surface chords (c-e-g, e-g-c, g-c-e, c'-g'-e' etc) belong to *one* identical chord type (C-maj).

The first depicting question is a combinatorial one about possible objects. How many chords are there totally? That depends naturally on the number of members in a chord. Rowe lists the combinatoric results in table 2.4.[326] With three pitch-classes i.e. triad-types, there are 55 distinct chords in terms of interval relations. Reasoning similarly about two-note chords with an interval relation, pitches of notes a and b are classified as pitch-classes (0...11). Every possible set [a b] has exactly one interval-inversion [b a] or more generally a circular permutation. Restricting sets to pitch-classes means that pitches [0 5] and [5 12] are identical sets in this system. There is no concept of root in such elementary chords. Every larger chord type is made up of intervals or sets of two-note chords. 'Root' depends on distinguishing between recurrent notes relative to major or minor triads.

Rowe starts with programming a context-independent three-note chords or triad classifier. He finds "220 distinct tree-note chords" i.e. chord with no-repeated pc'es. If we describe them by their intervals, e.g. [4-7] instead of pc'es [0 4 7], we get the 55 types of triads from above.[327] 10 of these 55 are named[328] and classified in tonal music[329], 21 are derivable from them while the last 24 are

uniquely classifiable. We realize that pitch class descriptions alone do not uniquely determine neither type nor root in tonal terms. Rowe expresses this situation like this:

> A context-independent identifier [of chords] will work faster and always produce the same result [listing standard chords] Table 2.2 for example is not style-insensitive, but rather encodes a set of assumptions about the contexts in which it will be used.[330]

In a wider perspective we should implement a complete chord identifier for all combinatorial or possible chords that are numbering 2048[331] But the vast majority of these chords with unrestricted interval relations and unrestricted number of chord members turns out to be without meaning to our ears e.g. the chord C-C#-D-D#-E is a perfectly valid 'quintad' (five-tone-chord), but is perceived as a too dense dissonance with insufficient structural function from (predecessor) or towards (successor) tones. In Western tonal tradition, triads based on tertian intervals are the most defining chords in respect to functional harmony or directed harmonic motion. We can even reduce the 220 or 55 triads to a lower number of meaningful or strategic important triads inside this tradition. Every triad that either consists of tertians or is reducible to tertian intervals is classifiable as major/minor chord type.
Rowe develops a tentative triad classifier[332] as a combination of table lookup for functional chords ("the most common and important identifications") supplied by a kickout-member-algorithm that recursively reduces strange notes from less obvious chords and finally checks with a manually updated exception list that describe typical malfunctions or errors of this rather ad hoc strategy. As long as we use unique note names (such as MIDI numbers), not C# or Db we need not to distinguish chords further. The chord C-E-G might be called Dbb-Fbb-Abb with identical physical sound result, but yet with different harmonic meaning and referring context. Just as one word can mean two or more different "things", essentially for any metaphorical meaning in language, can a chord be spelled *enharmonically* different, i.e. notated with different note name and hence different context relations. [333]

The 'Serioso Music Analyzer' (SMA) is more principled in deciding root and spelling for chord members. SMA distinguishes between neutral and tonal pitch class systems (NPC/TPC). MIDI numberings are only sufficient for NPC. TPC include enharmonic (multiple) spellings for unique MIDI tones. To be able to spell chords "correctly" SMA uses the ordering principle of "line of fifths"[334] for computationally finding chord spans with harmonic roots from only pitch numbers and metric placement. SMA finds the correct spellings based on finding the smallest distance along the line of fifths. Still, some non-fitting SMA-examples indicate the non-completeness of this approach.

> Now, the spelling of pitch classes depends not only on relationships within the current event, but on the context established by all the material leading up to it.[335]

SMA divides pieces into segments labeled with roots. It proceeds therefore away from a context-independent classifier (as Rowe's triad classifier above) into the domain of harmonic level analysis using preference rules (as known from generative theories such as GTTM) that assign privileged status to certain legal structures over other equally legal but less preferred ones by human ears.[336]

Global order versus local structure :
the Parncutt Root Salience algorithm (PRSA) for root-finding
Just as our perception of complex sounds (i.e. tones) presupposing a well-defined overtone-series **[ch2]** leads to paradoxes like the virtual or 'missing fundamental' **[ch2]**, similar things happen in the perception of chords. Common to these phenomena is the fact that organismic perception does not work like a passive data collector but more like an active theory constructor. Humans are able to hear root tones in chords that don't exist physically in the input but that emerge during processing in

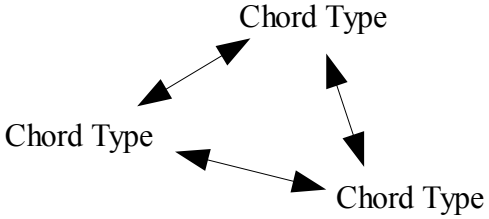
our heads. We try to make sense out of input not only by conservatively extrapolating. A perceived root tone *means* that this tone *represents* or describes chord structure. E.g. we sometimes feel that people around us are part of a bigger group that is represented by a non-present party leader. We therefore naturally associate the environment with that missing leader. This somehow illustrates why our mind *looks for* representative fundamentals, roots, progressions, (e.g. Schenker-style) and other traits that may not even be there in the surface material but only in its assumed deep structure.

Much of tonal music works like that: a system that listeners decode using *fifth-circle-type harmonic forward progressive potential or energy*. This is the central problem for all kinds of meaning-seeking activity. Meaning lies in the context or global order, contrasted with syntax that is described local and understandable as straightforward surface structure.
Also computational models will have to look for pitch relational categories in a contextual perspective. Richard Parncutt's root salience algorithm (PRSA) looks for the most "salient" (prominent or representing) pitch in collections of any NCP-chord-type using a *vector of weights* for all candidate pitches. Weights are aggregates of (context-independent) *root-supporting intervals* (roughly equivalent to the acoustic harmonic series called overtones), (contextual) *voicing criteria* (placement of tones relative to their neighbors, e.g. giving additional weight to the lowest note of a chord), and, finally, *influences of tonal context on perception* (empirically found pc-stability profiles [pc-SP] that let us quantify the hierarchy of stability in tonal contexts; (Krumhansl and Kessler,1982).[337] Such a Parncutt-root can now be used as input for Jazz and Rameau-theories of chord *type* that produce *qualified spellings* for chord members, i.e. naming the intervals above the root tone stacked as thirds.[338]

Resolving tensions between global and local: parallel and recursive solutions for key induction

> Neither pitch distributions nor chord progressions alone, then, seem to comprise all of the information used in key recognition.[339]

Even if we found an acceptable way to induce root and type of chords, we did partly so by begging questions: to use the pc-SP method we actually presuppose knowing key and mode (scale). How can we find them? Only by knowing roots and chord progressions? We are dealing with the classical *circularity dilemma*. We need to know everything for knowing something, and something for knowing everything (in the context of deducing pitch-structure i.e. tonality from pitch-relations or intervals).
Key is the *prevailing tonic* in a set of chords. It is a typical high-level percept. To induce key stability, i.e. finding the most central pitch to harmonic function, is analogous to finding of the "salient" root on a lower level. In other words: key and root are representatives of local (chord) and global (progressions) structure, but cannot be induced independent from each other. How do brains assign these structures and representatives? They work presumably in parallel networks (see ANN above) where no circularity arises. To crack such a vicious circle with no entrance, we must look for weak spots under suboptimal conditions:



Rowe entered this dilemma by iteratively applying weights to all 24 possible major/minor key

candidates keys (or key theories) and progressively supplying meaning through context.[340] Practically, this means to start with syntactical descriptions, then using the results for inducing preliminary semantical descriptions, that are again tested at less semantical or lesser order descriptions and so on. E.g. an algorithm finds first a chord root candidate *r*1 without context, using *r*1 for finding voicing environment (horizontal relations to neighbor tones) or context 1 (*c*1), then using *r*1+*c*1 to find *r*2, and *c*2 to find *r*3 and so on. This "backtracking" scheme or key induction feedback, enables us to enter the circle. It is not a theoretically perfect method, it will err sometimes, but so do the parallel processes in our heads.[341] To make machine recognition of harmony more robust in the sense of catching human abilities more comprehensively, we need to remind ourselves that we until now only employed information about the vertical harmonic forces (chords, roots) alone, except from voicing. Even though pc-SP uses bigger sets of pitches, they are applied without use of horizontal (or melodic) relations of succession. These melodical relations make up scales or modes and are the other class of input for the Western tonal system.

> The [pc-SP] algorithm correlates pc distributions without regard to the order of their presentation.[342]

Processing information about scales and chords simultaneously increases the contribution of context further, thereby excluding pitch-relations of lesser importance and retain the more robust and consistent ones. Including horizontal pitch distributions (melodies) and their durations (weighting their relative perceptional contributions), introduces new psychological factors such as the *primacy factor* (emphasis on early members of a series)[343] and other memory related constraints. But pc-SP is a more logical model that does not include such time-relations. Models that include horizontal contexts, satisfy psychological and realistic constraints underlying human listening.

The Vos/Geenen Model (1996) for key finding is another example of such context-rich and parallel processing tactics. Each key (theory) has updated melodic and harmonic scores derived from analysis."When the maximum scale and chord weights of all 24 possible keys are associated with the same key theory, that key is held to be the prevailing tonality. If the chord and scale analyses produce different leaders, the key is ambiguous."[344]
Like Rowe's simpler model, mentioned above, certain events contribute decisively as negative (or penalizing), i.e. weight-substracting factors, leading to both better and particularly quicker results. Recent extensions of Vos (1999) include tonal signatures such as dominant-tonic leaps in starting melodies as further factors that reinforce certain key theories.

How does this relate to musical intelligence or musicianship?
We saw that musical theory arises from both practical and theoretical perspectives. Modern theories of tonality, e.g. SAH, use abstract and generalized models with geometrical perspectives. Computational models of musical meaning, that develop key induction from bottom-up, are limited to pitches (without duration and temporal order). We found that such feature reductionism in musical analysis leaves out exactly what defines the higher level features of musical meaning, namely *context relations*. And even worse, we found that low level and high level (local and global properties) support and presuppose each other. This circle can only be solved pragmatically. We realize that the lessons from Minsky's agent architecture with communicating agents of low levels, also applies to musical understanding or musical intelligence. Because musicianship is action-oriented, we may build models of complex or high level traits with networks of smaller networks and so on. To construct semantical music systems we need to build agent networks that work from the ground and up, while reflecting and recurring, sometimes recursively. This is how musical intelligence has to be strata-wise implemented in machines with computational formulations of musical knowledge, as indicated in relation to pitch structures. In contrast, earlier computer music did not define transformation models with psychological relevance. MC systems on the other hand must be on speaking terms with modern musical theories such as SAH. Operative models of MC are

like laboratories or test grounds for music theory. In some future, models from music theory and computational MC might be expected to coalesce or at least become compatible.

By now we might conclude that MAX with its message communicating object model and Lisp with its recursive and open-ended nature seem to fit rather well into this picture of contextual complexity of musical meaning, knowledge and ultimately intelligence, or simply musicianship. After providing an example of a melotonic echo with knowledge, we turn our attention to the evolution of some well-known models of informed or knowledge-guided systems for MC.

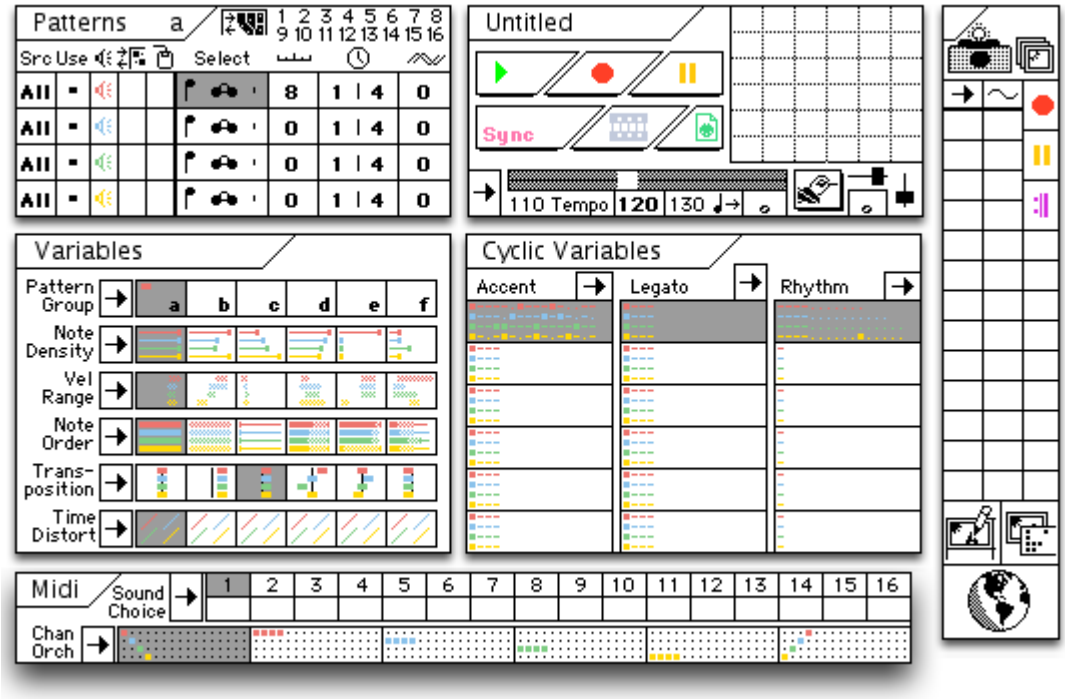
Example of a melotonic echo with knowledge?
In [ch3], we presented the melotonic echo that mathematically transformed incoming "melotones". What should we add for making it informed? First of all, objects are exclusively represented by MIDI-numbers only. We can in MAX easily display pitches as note names. But as mentioned above, the correct spelling of note names in contexts (note collections) needs an understanding of their common functions or meaning. Correct spellings presuppose knowledge of either scales in action (in monophonic examples like ours) or both the scalar and chordal forces in power. Let us sketch a knowledgeable 'melodic echo'. While 'melotonic' was related to collections of pitches alone, that we treated mathematically without interests in their inherent interval relations (actual scale affinities), a melodic or musical echo must transpose the input in such a way that the underlying *scale* is *conserved* in transposition.

How can a MAX patch find a scale affinity from melotonic input? We may use data structures for storing candidate scales (e.g. the four most used tonal scales: maj+3min). Each candidate scale is then represented as a collection of pc'es in a MAX table. All input pitches are transformed into pc'es and compared with every scale candidate. For each membership we add one, for each non-membership we subtract one. We further add extra values for starting members (see primacy factor) and ending members (cadence factor). Finally we may add values for the presence of the fifth step of scale. Thereby we get resulting weights for every "possible" scale candidate assigning various probabilities or consistencies to the collective input. Next, we use the winning scale with the highest conformity or probability to pitch-adjust the output tones. While our melotonic echo disregarded scale affinity and transposed rigidly or arithmetically, melodic echo transposes musically applying a certain minimum degree of musicianship or crypto-intelligence. Doing that we add and apply horizontal scale-context. We still disregard durations (time-relations or rhythmic features), as well as vertical or chordal pitch dimensions. In such simple environments, reductionist planning and implementations are both valid and successful, but at the expense of only permitting crypto-musical input of the analogous type. That means we encounter Eliza and other "context-stealing" installations.

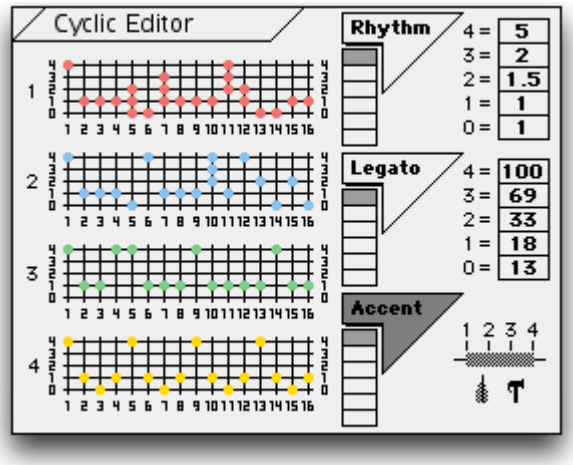
4.4 Cyphers parents: Music Mouse, Jam Factory and M

In 4.5 we look closer at the classic "Interactive Music System"(IMS) *'Cypher'* by Robert Rowe. Prior to Cypher most of computer music was essentially non-interactive and non-real-time. With the widespread diffusion of personal computers, the public gained access to the developments of interactive instrument-like possibilites in Computer music culture. "Music mouse" ('Intelligent instrument') by Laurie Spiegel (1986) was basically software that received pairs of values (or coordinates) from a mouse pointer and transformed or mapped it into four-voice harmonized output through the Mac or Amiga. In addition there were 44 "buttons" controlling higher level features of melodic, harmonic and orchestrating domains. Intelligent Music Inc. was a trend setter that started the use of computers as musical high-level instruments outside academia. In the course of the following two years a number of so-called intelligent instruments saw the light of birth.

Especially successful were *M* and *Jam Factory* by "Intelligent Computer Music Systems" (David Zicarelli, Joel Chadabe a.o).[345] *M* has a control-panel-like interface where stored presets (called variables) are manipulated in real-time. The user is acting like a conductor that influences the musical events with mouse-gestures on-screen, instantly reacting to M's output. *M* is a typical algorithmic systems. In the screen capture below we see how *M* on screen divides into blocks that control patterns (upper left), presets/variables (middle), sound quality choices (lower), time controls (tempo, start/record/stop/sync) and macro presets that store settings that may be recalled later .



Master settings of patterns or sonic material (such as pitch sets) supply for instance metric patterns (4/4 beat) that again are processed further by *cyclic Variables* (accent, legato, rhythm).



Other static variables or presets control higher level features (note density, note order, transposition type) and ranges for random generators (velocity range, time distortion). They are positioned on the middle left. Using *M is* something in between composing and performing, or is better described as pseudo conducting. Much of the sonic result of a "session" in M is determined by a few pitch sets, defined in advance, and the time governing tools, most characteristically the *baton*.

Jam Factory[346] is similar to *M* in many respects, but leans more towards playing and triggering than conducting and generating so typically of M. *Jam Factory*'s "sessions" are also more predictable. It lets four independent players output simultaneously, mostly transformations, according to settings and material chosen by the user and session leader. Risking over-emphazising their differences one could characterize *M* as specializing on structure and composing with Jam Factory excelling in automated improvisation.

The innovative software of *Intelligent Music was* perceived as and called "intelligent instrument".[347] Concepts like "intelligent" were cherished in those days, at a time of promising personal computers and high-flying and hyped expectations in AI. Today, it feels odd to to think of M as intelligent, because the software actually understands nothing more than the gestures and control information from its user. Input that M directly transforms, in pre-programmed ways, to typical "computerish" sound output. As in Eliza, or magicians in general, we see that the complexity of control is significantly smaller than the complexity of the result generated, a well illustrated phenomenon with Braitenberg's vehicles (see above). A truly intelligent system, displaying discerning and understanding powers, should therefore at least understand and hence *listen* to music itself and not only respond to human control interfaces. In the case of M we may say that the real complexity is in the intentions of the player/composer on the one hand and the imagination and ignorance on behalf of the listener.

Again we stumble on a recurrent theme of this thesis of how to define the products and activities springing out of the wish to use the "Ultimate Intelligent Instrument, the sound-producing general purpose computer itself".[348] Are MC systems foremost composers, improvisers, instruments or players? Answering the question we have to explore some more ground and provide other examples.

4.5 Cypher: an overview

Cypher (C) as presented in Robert Rowe, "*Interactive Music systems. Machine listening and composing", (*1993) and has become a paradigm system for MC. It has been used numerous times in teaching, theory and practice.

Components

Cypher has been called 'interactive computer music' system by its author because it falls between the categories of composition and performance. In Rowe's classification system C is a performance-driven, transformative and algorithmic player system. It consists of two main components: an analysis section or Listener (L) and a composition section and player section or performer (P). C does not work on stored material or scores. It analyzes free form input and generates output using algorithmic styles that define C's personality (see on meta-styles or aesthetics below).

Elements

C's components consist of *features* analyzed by its L and *transformations* that relate C's input to the output of P. L classifies on the first level [L1],[349] the features of density, speed, loudness, register, duration and harmony, assigning each tone or sonic event a unique point in a six-dimensional *feature space (L1)*, for instance [low, fast, very loud, legato,a minor].[350] On the next level [L2], L looks for changes of these features over time (temporal feature functions).

On higher levels, C looks for phrase-length in input, regularities/irregularities of different lower level features. In a sense we might look at the L1-feature space as the range of functions for MIDI-notes, while higher level functions are first- and higher-order derivatives of these functions.

Resulting vectors or feature spaces are defined within restricted value ranges such as 'loud' and 'soft', using thresholds for classification. Scaling features down to a few values results in a rather coarse gradation, but leads also to fewer categories, i.e. clusters of points in the featurespace or cube.

> "Further, higher listening levels will use the feature space abstraction to characterize the development of each feature's behavior over time"[351]

Using "fuzzification" of MIDI-data on every higher level, data resolution is reduced to practical sizes for processing.[352]

P is defined by its *methods* of response, e.g. initialization of algorithms under specified conditions (*variables*). Response methods are transformations to input notes such as transposition, delay, inversion and acceleration (see echo in **ch3**).
Users of C will function as "composers", performers or both when they define and tune the values of the connections between *features* of various levels and their *transformations* and algorithms (a total set of these values is called *state*). This is practically done by drawing lines between L1 and L2 features and transformations. An important modus and component of C is its rehearsal mechanism that permits restoration of whole states saved in earlier sessions in C, i.e. memorizing successful "takes" in repeated rounds of "playing around" with C. States are metadata of C that comprise variables for connections, sound choices etc. Sets of states are stored in *performance files* (storing sets of states). Successions of state changes, controlled by cues in input, may finally represent, not unlike scores, a complete performance of a work.

Architecture and inner fabric: hierarchies and progressions
Rowe is inspired by Minsky, Meyer and Narmour. In 'Society of minds', Minsky develops his theory of the Mind built around networks of partly autonomous agents. It is a non-hierarchical model of meaning and intention, that presupposes somehow *parallel* representations and perspectives[353] taken by "freelance" agents that participate in a sort of "grand symphony". This view is very much in line with Narmour's systemic implication-realization model, implemented as networks[354] of musical events on various levels, ordered horizontally and compatible with listening expectations. So are the intuitions of others, like Dewey and Meyer, that treated events in time-directed perspectives, ideas that Rowe implemented into Cypher:

> Eugene Narmour's theory tends to assume goal-directed, expectation-based model of music cognition.[355]

Listener:
Rowe uses ideas coming out of many music theory strands. He treats low level objects for instance as collections instead of rigid structures. Higher levels "use the abstractions and results produced of lower levels" and the higher a level the longer will its structures span in time. C may be described as a *nonuniform hierarchy with expectation-based dynamics*. There is certainly less hierarchy than proposed by GTTM (above) and there is certainly no pendant to well-formedness and preference rules. About the progressive perspective Rowe writes:

> The progressive perspective is adopted on other structures as well; harmonic progressions, patterns of rhythm or melody, and higher level groups of Cypher events all are related, at times, by the operations of succession and precedence... All events are connected in a hierarchy and simultaneously tied together in relations of succession and precedence... Many prominent music theories devise a single structural perspective within which to describe musical behavior...[356]

This multi-structural perspectivism is also reflected in Rowe's liberal or eclectic use of methodology. In harmonic and rhythmic analysis he uses connectionist-like principles from the

ANN approach. To find the root and mode of a section of piece, twelve input nodes (for each semi-tone in MIDI) update 24 Chord theories (12 major and 12 minor) with either negative or positive increments based on above mentioned tonal principles, adjusted by Rowe, through trials and errors.[357] Similarly, a C agent will update the scores of 24 Key theories on higher levels, with negative increments or weights contributing more than positive key weights.[358] Winning theories of chords and keys are associated with confidence values that measure the strength of the winning theory relative to the total strength of the surroundings scores.
Like musical theory, C uses information from other agents (density, register, beat a.o.) to improve the results of the analysis. e.g. density agents may inform a chord agent about the sufficiency of constituent chord parts, or register agents may help chord agents to assign greater weight to the lowest tone in chords. Then there is the contribution of beat agents in weighting events on the beat by factor of 1.1. And the interdependency of chord and key classification, above characterized as a vicious cycle is handled in C with information fed back from key agents to chord agents, similar to what musical theory prescribes.
Rhythm analysis is called beat-tracking[359] in C. Beat agents find the lowest level of Cooper/Meyers classic three-level analysis[360] of hierarchic rhythmic activity: pulse (regularly recurring succession of undifferentiated events). Starting out with no knowledge about the expected beat pulse, C needs a dynamic approach to succeed in real time for interactive use. Again C is somehow related to connectionist models, maintaining simultaneously many theories of possible periodicities in parallel.

> In the multiple theory algorithm, separate theories are maintained for all possible centisecond offsets within this range [288 to 1500 milliseconds]; in other words, offsets from 28 to 150 centiseconds (a total of 123 possibilities) are regarded as possible beat periods. (page 144) ...
> The first thing the tracker does is to examine the expected event arrival times of all theories. If the real arrival coincides with an expected arrival for any nonzero theory ... , points are added to that theory's score. ... If the real offset arrives later than an expected offset, points are subtracted from that theory's score. The heuristics here is that syncopations are unlikely; that is true beat pulses will usually have events aligned with them. [361]

Again thresholds fuzzify incoming data into manageable resolution. Then, the problem of “false negatives” or syncopations is handled by a 'syncopation heuristic part' of the algorithm that memorizes and updates five most likely interpretations of incoming event placements relative to a factorized patterns. The underlying logic of this agent is that geometric subdivisions or multiples (of inversely divisions) of 'two' and 'three' span networks with “good” places (having simple integer ratios) that we may use for predictions or 'theories' that ultimately are rewarded and penalized according to the real 'arrivals' or evidence. Aside from handling syncopation as natural exceptions a beat tracker must also catch the larger scale time deviations or tempo fluctuations carrying expressive intentions. Many smaller scale fluctuations parallel bodily dynamical contours or shapes. Tempo fluctuations are tracked using continual factorizational adaptations[362]. C does though not detect meter or higher level rhythmic organization.

Both vertical key- and horizontal beat-analysis can be informed by knowledge about how events are segmented into sequences. Grouping of events is in C done by a phrase boundary agency that uses *discontinuities* between classified features as criterium:[363]

> In Cypher, phrases are musical sequences, commonly from around two to ten seconds in duration, that cohere due to related harmonic, rhythmic, and textural behaviors. The level-2 listener detects boundaries between phrases by looking for discontinuities in the output of the level-1 feature agents. Each agent is given a different weight in affecting the determination of a phrase boundary; discontinuities in timing, for instance, contribute more heavily than differences in dynamic. The phrase boundary agent collects information from all the perceptual features, plus the chord, key, and beat agencies. When a discontinuity is noticed in the output of a feature agent, the weight for that feature is summed with whatever other discontinuities are present for the same event. When the sum of these weights surpasses a threshold, the phrase agent signals a group boundary.

Composer and player:
The player component outputs what the composition section has transformed and generated. A configuration of C tells the player how to respond to the results of analyses of input to the listener. The response methods are the agents of the composer in C and they operate at different levels, similar to listening agents. Configuring C means establishing and defining connections between features of $L_1..L_n$ and the various generative and especially transformative methods of response. Transformation is done in C by chaining many small response modules or objects, very much like listening was accomplished by small interacting agents. They can be reused several times in loop or later in the chain. Such object chains result in rather complex but deterministic output to the player. The behavior of response objects can be tuned with arguments that vary their precise transformational algorithms.[364] There is a clear and probably intended similarity of response objects in C with objects in MAX. Row uses MAX patches in his two books to illustrate the structures of transformations and generations implemented in the C language for Cypher. These are the Level-1 objects that transform and generate material:

Accelerator	*Gracer*	*Sawer*
Accenter	*Harmonizer*	*Solo*
Arpegiator	*Inverter*	*Stretcher*
Backward	*Looper*	*Swinger*
Basser	*Louder*	*Thinner*
Chorder	*Obbligato*	*TightenUp*
Decelerator	*Ornamenter*	*Transposer*
Flattener	*Phraser*	*Tremolizer*
Glisser	*Quieter*	*Triller*

These objects start to generate output when they receive the message *continue.* The process is then repeated with increasing/decreasing values for each call until a limit or its defined overall *duration* is reached. Second-level methods, somehow analogical to the second level in listening, settle the *degrees of direction* and *regularities in groups* of *events* or sequences. Methods of level-2 are usually triggered by messages from level-2 listeners and operate on the level of phrases. But level-2 objects also autonomously (i.e. without the interference of the user) establish, change (*mutate*) and break connections between L1-features and L1-transformations (example[365]).
The *level 2* composition objects are :

VaryDensity	*MakeBass*	*AccMutate*
UndoDensity	*BreakBass*	*SawMutate*
Phrase	*BeatPlay*	
JigglePitch	*BeatStop*	

Meta Listener: Critic
C typically interacts with a human player, i.e. listens to human input interpreted with built-in preferences and defined connections. This chamber-musical personality switches to soloist primadonna behavior when its own transformations or input stops: C autogenerates novel output in coherence with previous output of himself. Rowe calls this “composition by introspection”. It means that C can reflect on its own compositional output. It can also be influenced by a human “director” that regulates the solo-performances. In such cases it reminds of *M*. A critic that listens to

the players output and changes the configuration for solo-performances on the fly, works like a controlled feedback loop. Rowe illustrates:

> Feeding back on themselves, many transformations lead to registral, temporal, or dynamic extremes, where they will remain until something disrupts the state. Using the connection mechanism to reorder the modules called by different configurations of the featurespace provides just such a disruption. Another approach is to mutate the low-level transformations when level-2 analysis finds features behaving regularly. Pinned behaviors are flagged as regular, and a subsequent mutation of the transformations, ordered by the level-2 player, sends output off in another direction.[366]

4.6 Other informed MC systems

Cypher has been reviewed as some sort of prototype of MC systems. We now widen the perspective with presentations of other informed MC systems that share fundamental structure and outlook. In **ch3** we saw how transformative objects in MAX could produce complex structured output. Both delay and transpose are echoed in L1-methods of C, but were not implemented in **ch3**'s hierarchical and systemic integration with analysis at listening and critic agent levels with interactive and self-referential meaning as they are found in C. Neither M nor JamFactory were informed in a sense of musicianship capturing and manifesting musical concepts. M is highly interactive, just as C, and it lets the user configure massively algorithmic interfaces that control rather abstract featurespaces of sound containing moderate musical meaning. The algorithmic expressivity of MAX on the other hand aloud the construction of musically informed patches. A well-known strategy is to employ data structures of MAX, such as tables, to define musical forms as Rowe demonstrates with Guido's method[367] for composing “medieval” chant and isorhythms.[368]

Music Theater (MT) in 'Cybernetic Music'

In a book titled 'Cybernetic music'[369] Jaxitron (pseudonym) presents his computer-aided composition (CAC) system, called 'Harmonization/Melodization Workspace' or 'Music Theater' (MT). He uses 'cybernetic' (as governing or steering information) to express his intention to overcome the limitations known in the 'computer music' culture. Jaxitron's system is programmed in the non-standard programming language of APL.[370] The computerized music generation system is designed to automate composition routine tasks on computers applying a “logic of music”.[371] Of less practical importance today with its non-ASCII-characters in syntax, MT serves nonetheless as illustration of composition-assisting systems of MC in the days before MIDI and MAX. The author represents data numerically from very low to high level, with arrays and vectors. Higher level objects are encoded as lists. The system generates limited sound output only, results are written out as lists of notes in a rudimentary and non-standard notational format. Even so, his system has many structural levels inspired by and modeled after the intricate relations between different musical dimensions (harmonic, melodic and rhythmic, formal). A number of rules of thumb about e.g. reasonable voicing in polyphonic composition are implemented together with other constraint rules. At higher levels of musical representation concepts of order and freedom (different from random), tension, climax, distortions and tonality are applied.

A session in MT starts with setting global variables, choosing of functions that operate on 'harmonic alternatives' (HA array), 'target melody' (TM vector) and 'utilitarian purposes'. The composer /programmer then selects thematic material (example[372]) to be transformed and developed according to the system's arguments and functions, alterable even in operational mode (during execution). Jaxitron's model is heavily influenced by the today less influential Joseph Schillinger. Schillinger's system for composition (1946) may be seen as an evolutionary step backwards from musical serialism, even though Jaxitron used numbering formulas for music representation, applied mathematical laws of proportions and notions like symmetry, regularity and coordination. A theory of rhythm “extended to include “space” in a totally abstract sense”.[373] But even though

combinatorial and statistical aspects enter the presentation of MT, Jaxitron and Schillinger implement a more traditional model of rhythm, tonality and melody, unlike many of the high-level and abstract models of serialism and computer music. Jaxitron uses it

> To attack the cybernetic problem I must ask, "How do I choose rhythms of duration?" This leads us to further questions about the internal "feedback mechanism that keeps me "on track", accepting certain patterns and rejecting others. Eventually, the problem distills to,"What makes sense rhythmically, and why?" [374]
> ...
> A method that could take us too far afield uses computer graphics and what Schillinger called 'melodic trajectories' and the axes of melody in pitch/time space. Briefly, this involves the construction of 'curves' that will guide the melody... Each sampling then provides a point whose pitch coordinate needs to be adjusted the the nearest tone in the harmonic structure that is being melodized at that point in time. In principle it would seem that formal structure in the melody could be ensured be selecting and manipulating geometric shapes that repeat, reflect, expand, contract, and meet end-to-end through translation.[375]

Applying statistical arguments to pitch systems, Jaxitron is able to measure and *quantify* phenomena like tension where the total tension of a chord is derived from the *sum of the tensions* of all intervals in the chord. He splits this continuous dimensions into four tension-classes (range from less than 10, described as simple or "blah" to more than 1000, described as complex or "oops").

Combining the various parts and operations, Jaxitron speculates about stylized accompaniment possibilities and its representational questions, somehow foreshadowing the today ubiquitous Band-in-a-Box software.[376] Jaxitron concludes his exposition of often cryptic APL-code with a "Cybernetic song book" that exposes results from MT in the form of handwritten traditional scores. This helps to support his purpose to demonstrate that "computer music" in his "musical cybernetic system" does not need to speak with heavy technological accents.[377]

Jaxitrons MT makes, I believe, a strong case for the assumption that MC systems can be powerful without too much detail and veracity to the cognitive processes. It was constructed in a strictly bottom-up approach with moderate theoretical weight, almost exclusively building on the fundamental theories of Schillinger. Nonetheless he takes care of countless "cognitive facts" and traditional wisdoms while informally expressing them in MT-functions and operations. His sensitivity to constraints and relations between vertical, horizontal and geometrical aspects, as well as his hierarchic and quantifying methods may qualify his efforts to be be part of the AI-project of capturing musical meaning and not structure alone. Even though his formulations in APL seem rather unmusical in style[378], his results seem convincing enough. It may stand as an example of very weak AI, i.e. keeping high distance between conceptual representation and the represented actual structure in composing brains. We may characterize MT as a *weakly interactive* and *highly generative score-driven* composer system without *listening* functionality. It is therefore best described as an *composition assisting system* (or CAC) with *very low* (limited to some degree of randomness) *autonomy*. While it certainly is no *player* in Rowe's sense, it might be looked at as an *composition-oriented instrument*.

CHORALE: Expert system for generating Bach-style polyphony

Chorale is a widely respected approach using condition-action pairs (generate section) and generate-and-test method (test section) together with a heuristic section, constraints (negative weights) and recommendations (positive weights). The author, Kemal Ebcioglu, managed to inform Chorale with rules that map successfully to compositional practice in a specific and prolific musical style. Resulting chorals of Chorale repeatedly impress audiences with its successful emulations of Bach-style polyphonic repertoir. Often the weakest chain in expert system production is the knowledge acquisition or transformation of informal into formal knowledge. In this case though, centuries of teaching of baroque-style polyphonic techniques created a sort of canonical knowledge that only

awaited for to become translated into computational, in this case rule-based, representation. Extending the canon, many of the informal rules of thumb had to be sharpened and weighted for placing them into an expert system.

> Each step is executed as follows... : All possible assignments to the n'th element of the partial solution are sequentially generated by the production rules. If a candidate assignment does not comply with the constraints, it is thrown away; otherwise its worth is computed by summing the weights of the heuristics that makes it true, and it is saved in a list, along with its worth. When there are no more assignments to be generated for solution element n, the resulting list is sorted according to the worth of each candidate. The program then attempts to continue, with the best assignments to element n, and then, if a dead-end is later encountered and a backtracking return is made to this point, with the next best assignment, etc., as defined by the sorted list.[379]

Use of Chorale requires only feeding it with melodic material (a). Chorale then generate (a'), test (a'?), weight (a'!) and regenerate (a") and so on. Most of the searching and pattern matching functionality is supplied by the expert systems core ('engine'). The programmer can concentrate on the reformulation of specific domain knowledge into reformulations as facts, condition-action pairs and heuristic rules.

Chorale and MT share some additional traits too: they both assist in composition and output scores. They don't listen, only receive the “commissioner's” melodic input material (Chorale) or the programmers configuration and tuning (MT), i.e. they are very limited autonomous and interactive. Chorale is neither player nor instrument and generates complex output from simple input, but obviously inside very tight “freedom spaces”, speaking in MT terms. In other words: while C and MT are stylistically open-ended and flexible, Chorale is *narrow-minded* or *single-style-conforming*. Since there is so little flexible room for Chorales candidate solutions, we might even compare it to triggering-like sequence-following applications. Still, Chorale is due to its specificity producing the most streamlined results for an unprepared public. The emulation of Bach-like music produces an ELIZA-effect of astonishment on unknowing listeners.

C and M are highly interactive and therefore the improvisational systems in this group, while C's configurations and internal architecture are by far more elaborate than M's apparatus. Where M supplies only random variables, does C function with musical knowledge-based principles and laws.
In the next section MC systems are embedded in AI classifications and methods. After all, how do MC and AI systems relate?

4.7 Representation and methods of AI and MC (similarities and idiosyncrasies)
Four categories of AI

Marvin Minsky's view on intelligence and cognition ('Society of mind'[380]) was an important model for Rowe. It is closely related to AI as a proposal of both theoretical goals and practical applications. AI has always been divided between engineering-minded people and followers of cognitive-scientific programs. We encounter this difference in attitude also in language research of Computational Linguistics and Natural-Language-processing respectively. Now we will come upon it again in the four categories of AIMA[381]: Systems that *think rationally* or *act* rationally on the one hand, and systems that think *like humans* or act like humans on the other hand. The computational approach is more interested in the products of its models (act), while the cognitive-scientific approach is very much concerned about the explanatory force of its models and their validity for real brains in action (thinking models). Like-wise, we find a division between AI that focuses more on structure and hierarchy or *rationality* and AI favoring models that act *like humans* with massively parallel and non-hierarchical structures in the brain.

AIMA	*Criteria*	*Fields*	*Applications in music and other fields*
Think like humans	Cogn./empir.	Cognitive science	Meyer, Narmour, Gardenfors, Godøy
Act like humans	Engin./empir.	Connectionist AI	Chapter 5, MC, (Cypher), SAH, (MT)
Think rational	Cogn./rational	Logic, (Mathem.)	Schenker, GTTM, ES,
Act rational	Engin./rational	Symbolic AI	CM, MC, (M), Cypher, MT

Even if such a schema implies distinct divisions, it should be viewed in an explanatory light. We must constantly remind ourselves how the empirical presupposes rational methods, and how engineering "lives on" both the rational and the empirical. Nonetheless, it feels e.g. natural to place GTTM-like approaches with the think/rational and Cypher with the act/rational. Cypher is implementing a society of musical agents in Minsky-terminology.

> **Agent**: any part or process of the mind that by itself is simple enough to understand – even though the interactions among groups of such agents may produce phenomena that are much harder to understand.
> **Agency:** Any assembly of parts considered in terms of what it can accomplish as a unit, without regard to what each of its parts does by itself.[382]

It therefore contains aspects from the connectionist/empirical side of engineering as well. The core hypothesis is that complex performances may be generated from many small non-intelligent and self-contained (autonomous) agents that cross-connect and communicate inside a system or society manifesting intelligence as an emerging property. Such non-hierarchical, but multi-level architectures need to have context-sensitivity and classifying functionality at several levels. This again, necessitates memorizing information to make possible what Rowe calls 'focus and decay'. This could be paraphrased as dynamic or contextual threshold techniques, i.e. dynamically changing threshold values according to actual information or contexts. e.g. fishing net regulations could specify not an absolute mesh size, but a "self-adjusting" size relative to the quantity of fish caught until that moment. Similarly, thresholds for phrase boundary calculations in C are incremented or decremented relative to the found size of phrases. If the resulting length of phrases are small, they are increased and v.v.[383] Classifying routines, especially on higher description levels, make cooperation and alliances ('mutual reinforcements'[384]) between feature agents necessary. In such a mutual inter-agency, execution order and priority are non-trivial problems. Rowe informs his system with supposed weights for single aspects (feature agents) in various classification tasks that involve many aspects or agencies (e.g. beat and harmony agencies). Many of the high-level features are highly abstract aspects such as *regularity* and *contrast*. This applies equally to the requirements of composition or player agencies (*methods/objects*), where execution conditions control connection managers at various levels.

> The important characteristics of this architecture are preservation of the progressive perspective – that is , continual access to information associated with events proximate in time; maintenance of local memories on a per-stream basis for each agency; mutual reinforcement of cooperating agencies; and the attachment of analytic results to the events classified, where they can be read by other interested processes.[385]

The connection rules (production rules *modifying* the configuration of C's high level preferences) of the critic can even be defined with logical operators, due to the highly abstracted nature of their data. This amounts often in more than a configuration for specific musical pieces; it is more like a determining whole conceptions of aesthetics or definition of styles. Simplifying, one could identify rules at C's highest levels that emphasize *contrast*, *harmony* or *directionality*. Decisions about the appropriateness of style or even aesthetics[386] falls at least to some degree outside of the domain of cognitive science and is more influenced by cultural natures or composing personalities:

> Not all music exhibits such directionality, and therefore does not comfortably bear description using such terms; however some music, particularly much Western music, is about directionality and change. Cypher is biased toward looking for goal-directed musical behavior; however, the listener will still have something meaningful to say about music that is not primarily goal directed. In that case, useful analysis will be shoved down, as it were, to level 1. Classifications of individual features and local musical contexts will take precedence over longer-term descriptions of motion and grouping.[387]

Knowledge representation: Objects, frame, trans-frames, semantic nets

We distinguish knowledge representational issues from those that specify methodological choices in AI. Knowledge representation is, as discussed in **ch2**, the atomization or categorization of a domain.

Objects are the most common concept in knowledge representation. This constructional ontology derives from our intuitive understanding of physical objects and our natural propensity for defining concepts. Any computer programming, procedural or functional, makes use of objects in some way. OOP is abstract and systemic object-based programming, formalizing hierarchy and explicit relationships between objects. OOP uses "bio-concepts" like 'inheritance' and 'polymorphism' to build large taxonomic hierarchies analogous to biological taxonomies. Such structured approaches are also known in frames that are:

> collection[s] of related information, representing typical features of some situation, that will be used to direct processing whenever a variant of that situation is encountered... Much of the phenomenological power of the theory hinges on the inclusion of expectations and other kinds of presumptions. A frame's terminals are normally already filled with 'default' assignments.[388]

We saw in **ch3** that MAX follows a weak object-oriented approach. Not too surprisingly, objects, frames and other "molecular" representation schemes are fit for the sounds and notes in at least Western music tradition. A MIDI message can be said to be represented frame-like where several aspects sit in *slots* (or terminals). MAX and Cypher share this object- and frame-like behavior with most other MC applications. Some of them go further in applying specialized object representation like *scripts, trans-frames* and *semantic nets :*

Scripts are representations that define temporal orderings of events. Therefore they are well adapted schemes for musical events with both composite and temporal structures.[389]

Trans-scripts are scripts that include script-like slots with timing information, like 'trajectory', 'destination' or 'vehicle'. "*Trans-frames* include information about transfer or transformation, such as start,end and purpose.

Semantic nets are related to frames. (criteria: AIMA[390]).

They link concepts through relations of inheritance and specialization, linking objects with *IS-A* operators. Like trans-scripts, they have structural conditions encoded in slots for begin and end.

We will return to such structures with greater detail in the discussion of EMI's augmented semantic nets in **ch5**. Non-symbolic representation and ANNs were briefly mentioned in **ch2**.

Methods

AI is generally concerned with finding solutions to problems. Therefore search methods are central to AI. Often, systematic and full-fledged investigation of all possibilities are out of practical range, even with rather simple games. Therefore *heuristic* or *informed search* is the heart of AI. This is quite similar to the situation in MC. As we will see, informed search systems share a great deal with informed MC. Composers cannot and wish not to explore *all possible* art works. (NPcompleteness?, **ch1**) They just look for artful complexes that satisfy sufficient and necessary conditions. This implies that there is some element of non-determinism to their use of AI-methods, although this certainly not in the sense of blind randomness as often practiced by classic CM.

With unlimited memory and time we might always find a best-solution to some problem that can be modeled as a finite search tree (completeness, time/space-complexity, optimality for search

strategies[391]) But combinatorial explosions and very large search spaces make heuristics necessary. Classic search techniques, such as breadth-first, depth-first, depth-limited, A*, IDA*, SMA* , gradient descent and simulated annealing, deal with applying informed premises or assumptions about the nature of the problem and solution to reduce search costs.
Constraint satisfaction problems (CSP) will tests for goals (for solution candidates) and must satisfy whole sets of constraints. CSPs are typical for games like Sudoku but equally pervasive in music composition problems.

> In most real problems, however, we can take advantage of the problem structure to eliminate a large fraction of the search space. The principle source of structure in the problem space is that, in CSPs, the goal test is decomposed into a set of constraints on variables rather than being a "black box".[392]

For MC problems this requires optimization of constraints related to horizontal and vertical structure, voicing and rhythm, without asking for unrealistic computing resources.[393] Backtracking search and constraint propagation (forward checking) deals with complexity reduction of the problem space. Still, informed search methods for CSP implement knowledge into the search process and fare way better when choosing deliberately which variable to instantiate with which value:

> The intuitive idea is called the most-constraint-variable heuristic. It is used with forward checking, which keeps track of which values are still allowed for each variable, given the choices made so far. At each point in the search, the variable with the fewest possible values is chosen to have a value assigned. In this way, the branching factor in the search tends to be minimized... The most-constraining variable is similarly effective. It attempts to reduce the branching factor on future choices by assigning a value to the variable that is involved in the largest number of constraints on other unassigned variables...
> The least-constraining-value heuristic [means to] choose a value that rules out the smallest number of values in variables connected to the current variable by constraints.[394]

Another heuristic method is called heuristic repair or min-conflicts.[395] It uses a modification operator to move the actual state towards a goal state, by assigning values that result in minimum number of conflicts with other variables, thereby repairing inconsistencies during the solving process.

Constraint-based problem solving is also part of expert systems like Chorale (as well as integrated in Logic programming languages, such as Prolog). Such systems use implications as their basic representation. Rules constrain possible matching of patterns (unification), until combinations satisfy all rules and facts on the agenda. Forward-chaining control structures may lead to some arbitrary choice between valid alternatives if not supplied with conflict-resolution-mechanism to ensure deterministic results. Production systems are much used in cognitive architectures or models for human reasoning, including many MC systems (rule-based decisions in Cypher[396]).

Tentative generating and testing of candidates in searching or production systems is also known as *generate-and-test* as more general approach in AI.[397]

While search as concept is more tied to machines, thinking or reasoning is the equivalent for humans. The more energy a search process is diverting to strategic or heuristic goals (instead of the pure mechanical searching part), the more informed is the process. In MC, systems that use many rules linked to higher musical levels and abstractions (see Cypher above) are implementing constraints based on musical theory and listeners cultural expectations. Such empirical rules inform a system even further to avoid lengthy testing of irrelevant (or uninteresting) musical constructions. Practical planning and acting from a higher level or whole-work perspective constrain possible choices even further. Planning algorithms in AI constitute often trade-off dilemmas between

expressiveness and worst-case efficiency. This is somehow analogous with decisions about heuristic search techniques.
Planning in AI-problems requires hierarchical decomposition, i.e. an abstract operator is decomposed into many steps that combined form a plan that implement the operator. In AI-planning one takes great care of implementing it efficiency-minded, e.g. properties of downward solution and upward solution[398] that guarantee well-structured search trees relative to efficiently pruning away subtrees at a whole.[399] Typical planning examples in AI involve transactions like restaurant visits and building interdependent systems like houses.
In MC, planning is relevant for e.g. ensuring well-formed harmonic deep structures (overall Schenker ursatz e.g.) or formal consistency and coherence. If the last movement of a work should reflect thematic material from earlier movements, this would be a planning constraint or premise. (see planning system STRIPS; AIMA). Actually just like house building plans are decomposable into sub-plans and so on, planning full-scale compositions naturally involves decomposable sub-plans for parts and aspects that need orderly build-up and inter-genesis.

> In particular, composers use compositional algorithms to develop high-level methods, procedures that operate on a control plane governing the long-term evolution of notes and durations without forcing the composer to consider each event individually.[400]

Planning is also a necessary component of multi-agent based systems, such as Cypher societies (some planning involved) and Minsky's agent societies in general. Such systems must include or devise dynamical script-like plans for what or who is given priority to at any point and use feedback (like MT as well) and introspective adaptation mechanisms. Adaptive planning is referred to in **ch5.**[401] The more systemic the domain, the more planning will be necessary (unlike for ANNs). Systems with high interactivity or CAC must reduce planning requirements to a manageable size (e.g. Peter Beyls 'Oscar'[402], uses means-end-analysis for solving subgoals until a goal state is reached).

Another much applied formalism for high-level planning is GTTM that is mentioned at several places. The distinction between deep and surface structure, echoed as transformational and preference rules. These rules can easily be applied for generation of structures in specific styles, as is the major theme of the next chapter.

Other methods of AI seem to apply less to MC. e.g. Decision networks are often based on rational theories using utility as measure for success. Aesthetic problems seem to be more directed towards consistency, coherence and novelty than with values like optimality, efficiency and costs. This applies also to games that are solved with the shortest, and certainly not the prettiest, search path in mind. Decision making is nonetheless notoriously present in aesthetic reasoning. Most decisions will have to submit to criteria coming from informed music theory that is part of MC modeling. While random decision making was more accepted in CM, *successful MC increasingly substitute non-informed random-variance for informed theory-variance*. This may apply at absolutely every level as was discussed in relation to 'Jitter' and expressive performance generation in **ch3**. There is in fact randomness involved in Markov-chains, a popular generative formalism. But Markov-chains are random in a limited and controlled way, without threatening the coherence and communicability of its rendered musical structure. (see ATN, **ch5**).

Finally, AI has in the two last decades increasingly explored probabilistic reasoning systems and computational learning algorithms. Those approaches will be presented in **ch5** and developed further in **ch7** and **ch9**, discussing MC applications that exploit genetic algorithms and other new paradigms.

4.8 Conclusion: knowledge and laws in informed MC

In the opening of the chapter we introduced the task to implement musical concepts, instead of arithmetic functions. But looking inside the Pandora's box of music theory and analysis, we found neither context-free musical representations nor atomistic concepts. Many of music's deepest situated notions lay intertwined in a mosaic of tonal/rhythmic aspects and perspectives. It seems that lessons from Neurath's boat in philosophy and phenomenological issues in hermeneutics[403] apply equally insisting to machine compositional aspirations than to music theory. Modern solutions for filling semantic (and even pragmatic) gaps in syntactically or symbolically formulated ontologies propose increasingly geometric and topological methods, a trend that reveals itself in both linguistics (Lakoff), philosophy (Gardenfors, Petitot) and musicology (SAH, Godøy). This theoretical reorganization falls between well-established approaches of LS and ANN. Very much in accordance with the Cognitive sciences, it seems to solve old puzzles of meaning, induction, learning, non-monotonic reasoning and even creativity in general. Are 'Voronoi tessellations' **[2.5]** a common ground for theories that are thought to explain equally well tasks of perception as well as high-level planning/reasoning? Computer models of music theories will become experimental grounds for all sorts of conceptual spaces of musical representations.

Did we, after all, find the nature of informed MC? Humans that compose use their connectionist brains to come up with relevant and pleasing sound contexts. They create musical meaning as they are learners, performers, improvisers, critics and finally composers in different times. The integration of all those roles in the cycle of musical learning **[2.2]** is the precondition for creating meaningful compositions. Only complex and adapting beings have the powers to instantaneously combine all the aspects of tonal music with mutual coherent results. And only beings that have been inside this cycle for some time, and do therefore possess immense experiential histories, will be in the position to construct fluid concepts and styles that can be recombined and reused into seemingly novel compositions. The moral may be that human composers are truly and inherently complex agents (experience/roles), far from what machines at the moment can fulfill completely. But equally true seems to be that the compositional solutions of humans seem always decomposable into more primitive solutions. Even though the number of constraints for a solution is immense, there is no reason why it should not be finite. If finite, we should be able to build agent after agent, releasing them into increasingly populated and communicating musical societies (of agents) demonstrating, as a whole, musical discerning and understanding, in other words *emerging musicality* (or musical intelligence).

Applying formal and methodological strategies from problem solving AI (model-free heuristics) conjointly with applied heuristics from domain-specific musical theories (model-based heuristics) in a framework that ensures systemic adaptability and constructability upwards, and moving stepwise away from provisionary fundamentals towards learned and reconstructed walls under supervision of the equally evolving fields of cognitive sciences and others, is the key to informed MC.

Cypher's agents may not constitute a quasi hominid society yet, but it seems to have the capacity to grow in tandem with increasing processing power of machines and thinking power of MC theory. Often one expects of machines the equivalent of a GPS solver, a universal aesthetic problem solver for musical machines. This is actually an heretic thought in natural history terms. Just as there doesn't exist holders of universal intelligence among humans, or owners of universal musical intelligence among musicians, there should not be such a “theological” goal.

Also Cypher showed itself clearly as a *biased* system on nearer inspection, and so will any other MC system do. Actually, any natural system, including systems of systems when speaking about computational machines, will have to choose their profile. Chorale was even conceived as a copyist

of style limited to a unique composer, similar to the copying of famous paintings.

What we may have learned was that any interesting quality of music had something to do with reference to itself. In other words, since music does not signify external referential meaning, unless interpreted actively by a benevolent listener, its meaning is restricted to refer back and forward at the temporal theater of sound. Music gets meaning as a repetition, as variation and as a contrast (opposite); **9.5**. Sometimes subjects are played in parallel, variated in another voice or contrasted in another instrument, but as a rule, the order of meaning is related to temporal cues and references. We found contextual meaning at *multiple levels* of representation and contextual range. A defining property for human art lies in the multiple layers of meaning, or parallel contexts that may or may not be explored by a listeners attentive focus. This opens up space for intentional and free choice of deliberate concentration on one aspect rather than another and is what makes aesthetic experience different from a pure efficient solutions to a chess game, where winning and loosing are the ultimate outcome of the task.

Machines that are programmed using musical knowledge and theoretical laws of music, solving computational problems with heuristic methods and algorithmic mechanical efficiency, should be able to approach our creative processes, and surpass us in certain other respects relevant to composition. But more on this in **ch9**.

Cypher is very much responsive and an impressive listener at the same time. Cypher has self-referential behavior, if not introspective, capacities. But what Cypher is not able to do, is learning new tricks from his human partners, or adapting to ill-founded cases of responses. That is what we have to explore in the next chapter: how can MC learn from experience, learn new rules by induction, reason non-monotonically and especially learn styles from longer patterns?

Chapter 5
Rules and probability : Inductive Machine Composition
EMI and other experiments

1. Introduction to "the game"
2. Rule-systems and machine learning
3. The prehistory of EMI (CHORALE etc)
4. EMI: an overview
5. EMI discussed
6. Related inductive systems in music

Good artists borrow; great artists steal.
(David Cope: "I stole this line from Pablo Picasso who borrowed it from Stravinsky"[404])

I am enjoying the most productive period of my creative life, not in spite of ,
but because of, *virtual music. (David Cope, 2001)*

5.1 Introduction to "the game"
The general regularities or laws in musical compositions ('music theory') constitute the knowledge domain that algorithms will have to draw on to generate new compositions (or performances, MCs in **ch4**). Interactive MC systems analyze sound input in order to respond meaningfully to it. Several of the levels of analysis in Rowe's Cypher **[ch4]** supplied some kind of theory about the "sonic now". However, in addition to the general laws of music theory, experience from prior sessions (or works) will have to be kept in some form and reused later. In other words, Cypher is to some degree *rational* (music theory), but also forgetful ("deficient" in retentiveness[405]). This property of Cypher allows users to tune the connections between *listeners* and *players* [modules] in both setup and real-time play mode (see components of Cypher; **ch4**).

This chapter is about the 'experiments of music intelligence' or EMI[406] by David Cope. EMIs fundamental approach is to induce different types of sound patterns from a selection of actual patterns in sample pieces and learns in this way a particular musical style. Instead of the intentional rather narrow approach of Cypher, EMI makes use of large samples of music with expected similarities in period, style, composer etc. and extracts their significant stylistic features (e.g. from a collection of samples/works by a particular composer). This process of *style induction* consists in the finding of several types of patterns in given material. Often used patterns (identified as instances of type patterns within *error tolerance ranges)* are called *signatures* of the style. The resulting database of stylistic traits (signatures, unifications, rules) enables in turn the generation of new instances (let us say "Mozart-compositions") using the formerly extracted style definitions. In the terminology of IMS (Rowe), EMI is a triggering and score-driven system.[407] Cope likes to call the activities of EMI in the aftermath of both learning a style (*phase 1*) and applying the style (*phase 2*) for "the game" (*phase 3*). This last phase is the Turing-test-like "performance" of EMI-generated compositions, testing the quality of these new replicated style examples against a selected audience. We will present this last phase and its challenges in the discussion of EMI below **[4.6]**

5.2 Rule systems and machine learning

EMI learns by extracting patterns from many examples. It is pattern induction learning from examples alone, without feedback during learning lessons. It is therefore an example of unsupervised learning. It bears similarities with data mining, something we will discuss later **[5.7]**. When EMI's works are staged during the "game", a human performer, e.g. David Cope himself, are playing EMI's works. Cope will after the performance get some comments that supervises the shortcomings of a particular EMI-style, but mostly will the feedback from a public be reinforcement learning for Cope: either people hear a difference, or not. With this scarce form of information, Cope tunes the learning element and problem generator **[2.2]** (composing module) further to improve EMI's performance. Therefore we can say that EMI is a static and unsupervised learning system by itself. But the programmer Cope is actually taking the place of the critic element and part of the learning element as well **[2.2]**. In addition Cope is inspecting the output of EMI and discarding a part of it in preselection. EMI and Cope are therefore best understood as a unified learning agent architecture, where EMI does what she knows to do best (pattern recognition) , and Cope specializes on his more broadly expertise.

5.3 The prehistory of EMI

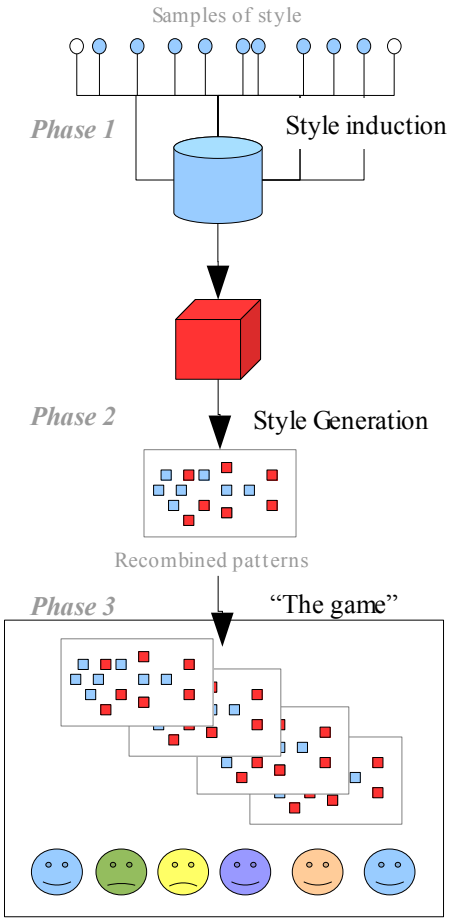
CHORALE, discussed in the preceding chapter, shares several features with EMI. It is rule-based, formulated as predicate calculus and uses generate-and-test methods. More than 400 rules describe as a whole the well-formedness of baroque (or "Bach style") counterpoint and are applied in generating harmonization of target melodies[408]. But in contrast to EMI that learns rules by itself, CHORALE rules are explicitly specified by its programmer (Ebcioglu). Therefore, CHORALE is a static defined expert system that applies pre-formulated rules in computationally efficient and user-friendly ways, when searching the "rule-space" for good solutions of vertical and horizontal optimalizations of poly-phony.

Other systems that may qualify as predecessors to EMI are Markov chain systems (Xenakis, 1971) and liberally interpreted the general practices, common in the 18.century, to apply non-deterministic decision processes, known as "Musikalische Würfelspiele" to generate new compositions in style-approximating ways. But since these methods, unlike EMI, rely on chance and probabilites, their success to "copy" a style will be most limited.

In contrast to such systems, EMI offensively seeks to blur the line between an *original* and a *copy* (or "fake"). The 'game', as Cope calls it, presents selected audiences with four music samples. Audiences are only told that at least one sample is composed by the human composer and at least one other sample is generated by the machine (EMI). Audiences are in turn asked to identify the correct source of each piece accordingly.

This method of testing the quality of specific EMI machine models is similar to the "Turing-test", and at the same time presents and diffuses Cope's work with EMI within a wider audience than that of specialized collegues in academia only. Much of EMI's apparent success is attributed to the "wow-factor" that accompanies the perceived and demonstrated difficulty to discern correctly "fakes" from "reals".

5.4 EMI: an overview



Phase 1:
EMI algorithms carry out processes of pattern matching that discover a number of *signatures* and similar identifiers. The '*Rules analyzer'* collects information about *constraints* and rules that are found via induction from the sample collection.

Phase 2:
The resulting information or knowledge of a style is then applied in EMI's association nets[409] *(AN)* or augmented transition networks (*ATN*) to build new works (instances of the style) hierarchically (top-down) as a tree of smaller musical increments, i.e. starting from the whole composition (tree) to sections (branches) down to voices and notes (nodes).

Pattern matching means here to compare "almost-matching patterns" with frequency lists of already detected patterns. Different musical aspects such as pitch, rhythm and the pitch/rhythm-correlations are analyzed by *tuners* that flag *amounts of variation*. Patterns that fall inside the of EMI defined 'error tolerance range' are kept for later treatment. Each sample (i.e. work) is represented as an *image* that "depicts" the results of the full scale pattern matching of the entire composition with earlier catalogued style traits in the database. By superimposing multiple images it becomes possible to distinguish work-specific *local* patterns from the more stable *style* patterns extracted from the the entire work selection. The more stable patterns are named *signatures* and are persistent[410] (occur repeatedly) in the superimposition process. *[SS-3[411]]*

The *analysis section or rule analyzer* for extracting musical rules is not too unlike the modules of Cypher's listener (Rowe). A series of subprograms in Cypher detect abstract features from the 'feature' space' **[ch4]**. They count certain events and supply statistical models of them.
The analysis section in EMI is more concerned with correctness than with style. Rules are rendering the infrastructure in terms of counterpointal style of the generated piece and are especially important at transitions between signatures. Thus, they provide a kind of "distraction" from signatures that otherwise might degrade to clichés. *[SS-4]*.

The meaning of signatures, not unlike words in natural languages, is often location dependent. The "narrative of signatures" must therefore adhere to a certain logic of musical values in order to avoid stylistical anomalies *[SS-5]*. This is taken care of by a specialized section module of the AN. It fills in signatures at the right places of the skeleton of a new work.

Rules found in analysis will now guide the process that fills in the transitory sections. The rules generate adequate texture by slimming thick textures or doubling thin textures phrase-by-phrase. This job is done by lower level modules for phrases in the association nets.

EMI generated works are scores that need "humanizing" by a human performer to ensure that style alone and not an eventual machine performance itself will become the object of attention. Therefore are EMI games conducted by human players, usually on a piano *[SS-6]*.

EMI is written in Common Lisp (CL) and fits well into the hierarchical frame used by ANs and the signature paradigm. Works are represented by lists of numbers that describe intervals and lengths, a subset of the expressive apparatus of MIDI-representation. Since EMI produces notated scores, the performance related parts of a MIDI-representation are omitted. To find a signature e.g., EMI recursively applies the Lisp-function `superimpose` to lists of numbers that represent candidate patterns:

```
(defun find-signatures (type first-work second-work window threshold allowance)
      (superimpose (analyze  type window first-work allowance)
            (analyze type window second-work allowance)
                  threshold)))
```

Superimposing two works (or images) means here to compare motive lists (from `analyze`) and search for global signatures that stand in contrast to local, more thematic motives in the list of candidate patterns. In general, signatures will increase in frequency, while mere motives will slowly "dy out" during the superimposition process:

```
(defun superimpose (image-one image-two threshold)
      (if (null image-one) ()
            (let ((test (assoc (very-first image-one) image-two :test 'equal)))
                  (if (null test)(superimpose (rest image-one) image-two
threshold)
                        let ((test-1 (+ (very-second image-one)(second test))))
                  (if (> threshold test-1)
                        (superimpose (rest image-one) image-two threshold)
                  (cons (list (first test) test-1)
                        (superimpose (rest image-one) image-two
threshold))))))))
```

Naturally, new style definitions in EMI reflect a particular selection of samples. If waltzes, ballades and rhapsodies of Brahms are supplied to EMI, the resulting signatures and rules should generate pieces with exactly this mixed identity. In addition, one will hope to find a 'Brahmsian quality' from the underneath.[412]

Douglas Hofstadter describes EMI's "central modus operandi" as chopping up and reassembling, and uses the analogy of a jigsaw puzzle, where reassembling is done in similar ways: [413]

Form/syntactic meshing	Content/semantic meshing
Make the **local** flow-pattern of each **voice** similar to that in source pieces	Make the **global** positioning of **fragments** similar to that in source pieces
The **shape** of each piece meshes tightly with those of **neighboring pieces**	The **stuff** shown on each piece makes sense in the **context** of the **picture**

The formal considerations of voice-leading and matching of texture (local flow) are dealt with by the musical rules from the analysis section. Content considerations need a closer look here. There are several major mechanisms that inhabit the deep waters of EMI:

First, *unifications* in EMI, a higher level object than the rather superficie signatures. Such patterns help to unify the work and Cope calls them 'unifications'. Hofstadter calls this idea "templagiarism", standing for template plagiarism. Unifications are structures associated with information about their pitch and temporal displacements relative to each other. It functions more like a network of

signatures. The higher-order stylistic feature of unifications is about *how* motives (or signatures) recur within pieces. Hofstaedter calls them “astonishingly effective style-evoking devices” after his hearing of EMI's virtual[414] 10th sonata by Prokofiev from EMI. Hofstaedters words are:

> Cope's idea of templagiarism is itself brilliant and devilishly impish: it borrows a touch of genius from the composer at such a high level of abstraction that when the pattern is simply quoted lock, stock, and barrel – plagiarized , no more, no less – it once again sounds like a touch of genius, but an utterly fresh and new one.[415]

Unifications strictly belong to single works, i.e. they are not superimposed pictures of style like signatures, but they may be reused in new works to replicate the target composers compositional technique and to ensure unity and cohesion.
Another pattern-based device are *earmarks.* Earmarks are location-specific gestures that provide the listener with clues about what structural changes are coming next.[416] Cope demonstrated earmarks in a Mozart piano concert. It feels like a sudden shift of character (often using syncopated rhythms) that prepares listeners to the coming near of a structural section, i.e. structural anticipations. Such “icons” hold little interest in themselves. Earmarks are more passively supporting meaningful structures, and are easier to notice when absent. They are somehow like structural DNA that doesn't contain RNA information, but may be in use for indirect functions. Earmarks are therefore expected to appear near ends and beginnings of important events or sections.

> Earmarks are principles rather than data and hence the earmark pattern-matcher returns an abstraction representing the type of material used rather than actual musical events as required of recombinative composition...Earmarks play a critical role in EMI's ability to generate logical musical structures. Using earmarks appropriately in algorithmic composition enhances stylistic integrity, formal and structural balance, cohesion and ultimately, I feel, aesthetic value.

Second, EMI uses what Cope calls SPEAC-system. Fragments or patterns are given identifier labels for *S*-tatement, *P*-reparation, *E*-xtension, *A*-ntecedent and *C*-onsequent to describe their function inside a Tension-Resolution Logic. It is derived from ideas of Schenker-style analysis.[417] This is nothing novel in itself, apart from the fact that multiple layers or levels from the most local to the most global (notes, measures, phrases, periods, sections) are made to fit into one single system.

> Thus progressions of identifiers such as P-S-E-A-C and S-E-A [describing the Tension-Resolution status or function at multiple layers on two distinct events; *my insertion*] seem logical, while progressions of identifiers such as A-E-P-S and S-A-P-C, while still possible, are less plausible. SPEAC identifiers follow an A-P-E-S-C kinetic order with the most unstable function to the left and the most stable function to the right.

EMI looks for a “drama description” of musical events that obey the extracted style description in SPEAC-terms from the original works. It is like borrowing the dramatic structure from a style and applying it on either archived or new melodic material, material that may come from either a human user or may be supplied by EMI herself. In Hofstadter's words:

> In the end one winds up with SPEAC labels attached to sections of many different sizes and, perforce, at many different structural levels. The upshot of this many-leveled labeling process carried out by Emmy is that any local fragment of an input piece winds up with a set of labels – its own label, that the larger fragment inside it sits, then that of the next-larger fragment in which that one sits, and so on, and so on....
> Now the trick is to use these labels to guide composition, ...Suppose that in our piece-under-construction we find ourselves in a location whose tension-resolution status is P-A-C-S-C-S (moving from most local to global).... And so, in choosing a fragment to borrow from an input piece and to insert right here, our main criterion will naturally be that the chosen fragment's tension-resolution status inside its original piece was exactly P-A-C-S-C-S – in other words, that the fragment we are going to quote lies in “the same place” inside its original piece as in the new piece....
> Thus, like a crystal growing outward, is built up a piece of music by Emmy.[418]
>
> ...how these various methods operate in tandem. Here we see signatures, earmarks, unifications, SPEAC, and

> so on interlocking to create new music. Signatures help produce stylistic continuity. Unifications ensure that in-progress compositions continue developing similar materials. Earmarks provide clues for important structural events. SPEAC contributes to the contextual selection of logical choices during recombinancy.

Hofstadters analogy with crystals is thought provoking, since fractal ideas in music composition are far from exotic. Musical grammars (generative ones especially) and psychological theories (Meyer employs notions and qualities of coherence and geometrical formalisms, see also Gärdenfors). It is in line with Cope's own analogy of recombinant structures in the universe. Atoms build up into molecules, genetic recombinations build new viruses and humans, and even language itself results from the recombinations of words and sentences. In general, cultural expressions are built from dead or dying expressions, not much different from the reuse of pitches and durations that result in new pieces and styles in music. Experiments in Musical Intelligence (EMI) can be seen as a model and tool for *reverse engineering* of such processes. As Hofstaedter accentuates, the proof of this pudding is in its eating, i.e. the successful production of the over 6000 compositions it has produced by 2001. EMI has been updated continually and expanded since 1981. It consists now of over 20.000 lines of code. Much of its expressive potency lies in the *clever integration* of a multitude of devices, types of patterns and levels.

> ...how these various methods operate in tandem. Here we see signatures, earmarks, unifications, SPEAC, and so on interlocking to create new music. Signatures help produce stylistic continuity. Unifications ensure that in-progress compositions continue developing similar materials. Earmarks provide clues for important structural events. SPEAC contributes to the contextual selection of logical choices during recombinancy. [419]

After this short scratch at the surfaces of EMIs depths, a more general and fundamental outlook is in place. Commentaries about EMI are presented in the next subsection before EMI will be compared to Cypher and put in a context with AI.

5.5 EMI discussed

In 'Virtual Music - Computer Synthesis of Musical Style', EMI is first described by its author and then by Douglas Hofstadter. The commentaries participated in a colloquium on computers, creativity and EMI at CCRH and Stanford University in 1997.

Limitations of EMI:
EMI is data-driven. In other words, the selection of works that EMI analyses is in a significant way defining its outcome and style. This selection is done by human intervention (Cope), i.e. not by the machine itself.
Timbral[420] and dynamic information are left out of the EMI database:

> "Musical styles that are distinguished principally by their timbral or dynamic qualities are not suseptible to analysis in this program."(SS-1) [421]

As a consequence performers of EMI-works should be aware of such extra-EMI-stylistic features that should be added in human performances of EMI works. Cope prescribes (SS-6) that human performances should be executed in expressive style. EMI presumes therefore a defined and clear division between composition and performance.

Bernard Greenberg in "Virtual Music" looks at the Bach-style output of EMI and finds a certain "timidness" and smoothness in its counterpoint. He finds in the Passacaglias and vocal works by Bach a more dramatic and less formal perspective, that includes semantic extra-musical meanings described as "tragic drama":

> The style of Bach is, vor allem, the routine application of titanic, incomparable, multilayered conceptual and intellectual depth to the problem of making music speak...
> We have yet much to learn from Sebastian Bach of Leipzig, not merely of counterpoint, the implementation language of his choice, but of his use of it as a tool to craft miracles of insight into depths of the human mind and heart.[422]

He suspects the presence of a different "emotional and dramatic structure" in addition to the common sonic tools that build musical structure. He recommends to add a similar model of dramatic rhetorics. Without such a model of the "space of emotional dynamics" or drama he does not believe in emotionally fulfilling art created by EMI or other automated composing systems.

Steve Larson, too, concludes with his impression that in Bach's inventions:

> ...one can usually *hear* a clear relationship between the melodic material of such a sequence and the invention's subject or countersubject. In your sequence, I can *reason* out such relationships, but those relationships are hard to *hear*.[423]

Jonathan Berger stresses the importance of the creativity in the act of *listening*, leaving thereby less importance to the creative aspects of the work itself:

> The interplay of expectation and realization demands creative mental processing. Any definition, characterization or replication of style that fails to capture this interplay will fail to produce anything beyond a local musical surface with convincing stylistic identity. [424]

All commenting contributors of "Virtual Music" are clearly impressed by EMI's compositions, but there is nonetheless a perceived lack of depth in EMI-compositions, notwithstanding their persuasive powers demonstrated from behind the Turing-like "curtains" of the game. Cope admits to some of the commented issues, especially regarding the limits of EMI's counterpointal rule-based subprograms (from the analysis section or 1.phase, see above). He would rather have prefered to extract formal counterpointal knowledge from the samples directly and use a database as for signatures and the other features.[425]
But the lack of depth experienced by listeners is not what Cope takes to be part of the work itself really, but thinks of it as derived from context mostly (e.g. textual meanings in voices). Like music philosopher Peter Kivy, Cope believes music to signify or mean nothing more than itself, ie. a chromatic fughe in itself (apart from texts of voices) signifies or communicates not pain but only chromaticism.

Eleanor Selfridge-Field has objections of more philosophical nature. She asks if computer simulations of a style (or any simulations really) are epistemologically of the same kind as their original compositions. She sees EMI ultimately as a "reprocessing engine", as *generative but not creative*, and that EMI's output are meta-compositions with "recycled" material from the "fabric" *of* actual works. She asks whether they are "simply mimetic imitations" and attributes the genius of EMI to Cope himself [426]as an acquired genius and not an authentic.

These arguments rely to varying degrees on the fact that EMI has neither "original" material nor real intention. Therefore, some of them believed that the "simulations" of EMI will have to do without the emotional dynamics or architecture (and hence without the communicative power) found and heard in human composed works. Cope defends his transformational and recombinative methodology by pointing to similar and well-documented practices in music history. Composers according to this view always borrow motivic and structural material from teachers, other composers or even from their own prior works. He thereby likens the computer-acquisited database of style elements to the analog minds of historical composers. And as a consequence, talk of

derivative, represented, abstracted or simulated nature of works are then not much more than pejorative names for a practice that critics associate with “second-hand”-products. Cope's reply is:

> Early in 1992 I and EMI composed 5000 works using a small database of Stravinsky's music and then inserted the program's *output* into the database, *replacing* the original Stravinsky(...). I hoped that the program would develop its own style which I believe that it did, ultimately. ... I personally composed a work in the style of [Stravinsky-style] EMI. In effect, I reversed the process by which my program creates music. My question is, which of these two works are the “simulations”? The tables have turned – thus I presume my imitative [personally composed] work is the simulation. However, being human and having spent a significant amount of time composing this new work, I flatly reject that label. I argue that my work has more inspiration and intuition than EMI works have. However, if you maintain that the program's works are simulations, then we can deduce the true rationale for using the term “simulation” instead of “composition”: machine programs cannot, in this *narrow* view, compose real music – a singularly homocentric and seriously limited view of creativity[427]. [my italizations]

Selfridge-Field repeatingly insists on the argument that since EMI could not have invented itself, it can not be creative either. Cope answers that neither did Mozart invent himself, “nor was he unbounded by the human values of his culture and historical context”.[428]

Copes opinion is very much supported by Daniel Dennett who goes even further in postulating that all “sorts of creativity” are algorithmic in nature and that all are “fruits of the tree of life”, ultimately. Dennett looks at EMIs products (compositions) as the products of a product of a product of the tree of life:

> ...but I want to argue, its means of production are simply special cases of the very same processes that created both the compositions by Bach and Greenberg [who happens to compose in epigonic Bach-style], the apples and spider webs, and the organisms that made them.

In Dennetts grand view everything is made up of huge sets of combinations (or recombinations) that in turn are reduced by choosing agents. This process is either called generating diversities and natural selection, or generate-and-test (trial-and-error). The selection of popular composers by audiences or effective versions of a composition by their creator all depend on the “billions of years of irreplaceable design work”.[429] For Dennett there is a “continuity of all sorts of creativity”. Dennett's philosophy of nature will be further discussed in **ch9**.
For now, we may only state that Cope's principles of recombinancy in music creation are fully compatible with such theories. What is more, it makes the fact of other influences in the works of composers (memes), such as religious culture and patterns, a natural consequence. Neither does the lack of intention or the absence of “genuine” musical material pose a fundamental problem for EMI.

Dennett's main argument against EMI is that it is not subjected to selected pressures (however non-material there might be). He mentions the example of the virtual creatures of 'Sims', and video games in general. Even though EMI demonstrates a high level of complexity, Dennett requires in addition evolved levels of environmental pressures (environmental complexity). As a corollary, EMIs composing in Bach-style is not sufficiently intruded by spontanous “noise”.[430] Noise that might eventually transform itself into “interesting” inspiration or depth-providing intuition. Like the “collision detection” of virtual creatures on the screen makes up a more credible version of moving behavior, Bach's Coffee cantata gains interest owing to the memes[431] of general life and culture associated with it. To Dennett the exploitation of accidents (or failures) is a key to creativity as phenomenon (“whether what is being made is a new genome, a new behavior or a new melody”[432]). He believes that learning as well as surviving presupposes great (i.e. diversified) worlds as well as great intelligences.

But with such an all-encompassing view on creativity, Dennett creates some trouble for EMI. He

thinks that EMIs compositions may in its worst case have meanings like babbling, scribble or gibberish only (like e.g. randomly generated Hegelian texts, manipulated by ways of using the n-gram-frequency method[433]), something he calls *muselot* in the case of music. In other words, Dennett thinks there may not be clear and cut distinctions between meaningful music and music empty of meaning.
Cope's answer to this objection is the music philosophical answer that music certainly is not like a catalogue of names with meaning or designating objects (standing for things). Even so music is full of *abstract* relationships, or what Cope calls *aural architecture*, that *becomes* meaningful after deciphering or perception only when trained to understand. Nonetheless, Cope admits the lack of selection-pressures in respect to EMIs models of style. He proposes the reverse process, where Cope artificially adds noise to the otherwise consistent styles. But what would happen if a database based on hundreds of Mozart samples is spiced up with samples of Bartok? I think, Cope's answer to Dennett's objection is rather off track, because non-deterministic ornaments to a style will not substitute for the effective crane-technologies of environmental dynamics proposed by Dennett. More on evolutionary programming in MCs **[ch 7]** and Dennett's relevance to our subjects **[ch 9]** later.

Cope concludes ironically that his various commentaries have completely different ideas about machine creativity; Selfridge-Field that excludes vague domains from machine creativity, Larson relating creativity exclusively to human cognition, Berger reducing it to listening only, and finally Dennett who finds it “merely” accidental.

But at core there remains the question whether EMI is providing a deep or shallow style definition. Eliza fooled some people to think that it had more depth and bredth than it actually was endowed with. But in the case of EMI, how many analyzed works for style definitions are sufficient? And can superficial memes such as signatures really be at the center of style after all?

> Cope's signatures correspond in my view, to an inventory of a composer's clichés rather than to the essence of [...] style.[434]

Are signatures of EMI “subconscious musical tics"[435] only? Certainly, EMI is more concerned with structural depth than signatures, and that may well be part of explanations to its success. Still, it seems to be an indisputable fact that EMI opts more for data (bigger databases) than rules.[436] This is possibly at the core of Hofstadter's argument about the nonexistent soul-emotional substrate in EMIs works. Cope's answer is that Hofstadter confounds deep and complex with good.

Rolf Inge Godøy[437] (1993) writes that “some kinds of more macroscopic information, such as more high-level harmonic patterns, seem to be not very well captured by the system, i.e. there is a lack of direction or kind of “pulverization”, perhaps because the principles of ATN do not grasp these structural levels well.” Godøy thinks this is due to the bottom-up approach, but actually EMI is using more an horizontal approach, collecting non-hierarchically sorted signatures. Godøy thinks that EMI's as well as Lerdahl/Jackendoff's GTTM suffer from a kind of rule-based thinking that hinders certain properties of “musical substrate” to emerge in “properly fashion”.

Finally, there is the obvious lack of a critic module in EMI. It has no “self awareness” based on self-changing or adaptability. EMI doesn't tune itself based on its own output and the results of its “games” with unsuspecting audiences. Therefore, EMI can neither judge nor enhance the quality of its own output. It is a clever system for acquiring certain knowledge *automatically*, i.e. dealing with machine acquisition of knowledge, but for help it is utterly dependent on its creator.

Gareth Loy[438] asks whether EMI is intelligent. Loy takes advantage of the updated experiences with

EMI in the very last years and compares it to 'Deep Blue', the chess machine that wins over humanity's prime players. Like Deep Blue, EMI essentially uses a brute force approach in AI terms. It doesn't reason so much in heuristically ways (search trees), but collects huge signature databases. Deep Blue uses "a large catalogue of chess moves created by analyzing many games of chess masters".[439] The similarity of approaches is evident. The question is whether intelligence is best represented as a "Humean association net" or rather like a Descartian reasoning mind? Such questions will reappear in **ch9**, but we will here close with the elegant argument of Loy about the quality of EMI in the longer term. Loy discusses one of the techniques employed by Cope, to extend the input not with more compositions of the target composer (say Haydn), but with the Haydn-"copies" of EMI. But instead of only adding new signatures, he thinks to remove the old Hadyn-signatures with EMI-Haydn-signatures. That is, one uses Haydn first for style definition, and then continuously substitutes the original Haydn style with instances of it,i.e. the children signatures of Haydn-style replace father-Haydn-signatures. Loy's question is illuminating and posing a rather precise question as to the real nature of EMI's intelligence. Does the music of second-generation signatures (children-signatures) develop musically in some way? Or does it degenerate? In case of the former, we talk of real living musical intelligence, in the other case we may rightly call it a simulation.

> It would be particularly interesting to know if stylistic stagnation resulted if the targeted style were not mixed with others. This might open up an understanding of the interaction of personal creativity and social forces.[440]

Prospects of EMI

In any case has EMI changed the whole debate and premises of MC theory. In its current shape, it might well be able to judge disputed authorships of music manuscripts more expertly than musicologists. May be future machines like EMI will even reattribute specific works that we think are of Händel to be actually composed by Bach or v.v.? Genetic knowledge and their revolutionizing methods are already today revising whole scientific theories,[441] as they often turn out to be better arbiters using carbon dating and DNA testing than former reasoning scientists and their argument-based conclusions. We may well be prepared for major changes in musicology as well.

But the by all standards most promising road for EMI is the opening up to learning in dynamic ways. A combination of selecting environments, and critics could well turn its achievements to become a winner in style-complying composition. But even in new styles, the methodology and research tactics of EMI and David Cope might well turn out to be superior to the modularizing network paradigm of Cypher for instance.

5.6 Related inductive systems in music

The idea of mining or exploiting data for valuable information is subsumed under the term 'data mining'. It means in general to search through databases in intelligent ways to discover the treasures of unknown lands of data. Such an approach is very similar to Cope's first phase of EMI. But several attempts have been undertaking to use such thinking to be able to classify music samples, without necessarily going further with generation of similar pieces (EMI). Paul Vitanyi has presented such an "Algorithmic clustering of Music",[442] that does apply a fully automatic method to analyze music pieces by first compressing strings that represent it. It is in no way limited to music applications. It uses concepts like 'information content' and Kolmogorov complexity', information distance, and a universal similarity metric. Using these methods he is able to rather correctly classify both genres and composers of sample pieces. This distance-formulating classification makes rather "good-looking" output trees, spatially grouping e.g. symphonies of several classical composer into families with reasonable distances to each other. What happens if humans and

machines disagree? Just as with EMI, in principle the machine could be right, somehow paralleling the way historical knowledge has been subjected to revision by later generations. Only, that this time, the new generation are machines. Vitanyi e.a. admit that the "experimental results got decidedly worse when the number of pieces grew". As we know from other AI cases, challenges with upscaling knowledge technology are common, but in spite of that often solved through various clever designed schemes (e.g. Deep Blue).

The internet allows for growing numbers of sites that offer "Music discovery services". Pandora.com e.g. goes for a mission to "reward the musically curious with a never-ending experience of musical discovery". Their musical genome project has already classified over 400.000 music pieces along with 40 "genes" (features). But their classification is done by human experts with musicological training. Still it parallels in some ways the signature paradigm of Cope. When such projects are open-source and openly distributed, in similar fashions to amazon, they may exploit as well the contributions and selections of numerous distributed users to make their classificational models expand and discriminate growing numbers of styles and features. Some projects use in cleverly ways the grouping of music composers by their many users exclusively, i.e. without using any lower level of features altogether, to "understand" styles and their inter-style distances from the empirical clustering of inputs alone (an example of unsupervised learning actually, **2.2**).

With time we may see an integration of intelligent classifiers (EMI) that have semantical models of features with the smart classifiers that use only syntactical relationships exploited from superficial preferential groupings of human listeners (Vitanyi e.a.) to classify semantical distance.

Such an approach may in the future furnish us with music discovery services[443] that may well become universal sources of expertise for new *ideas of patterns* as well as *musical piece stylistic properties* and thereby with the highest possible probability propose and predict appealing and pleasing music material, not unlike amazon and other distributed services already today.
But many of the successful leaders of popular music industry[444] are certainly very much aware of the important elements and mechanisms in, what we later **[ch 9]** will call the memetic industry of top-something music industry, successful tunes with pervasive potentials.

Classifying systems like the ones mentioned in this system, are very much *related* to the listening modules in MCs. But where classifying systems are more concerned with inducing *high level labels* and metrical categories, *listening systems* in Machine Listening and Machine Composition on the other hand are inducing information on many levels and are more interested in understanding the *systemic complexities and functionalities* of musical pieces. MCs will in turn apply this induced knowledge and combine it with the informed knowledge (pre-defined; **ch4**) to generate new pieces of music that make sense.

Chapter 6
Lists and sounds: musical Lisp environments
CommonMusic, Csound, OpenMusic a.o.

1. What is it about Lisp?
2. What is CommonMusic?
3. Structures and processes in CM (CommonMusic)
4. Examples of CM (CommonMusic) and CLM (CommonLispMusic)
5. CM (CommonMusic) compared to MAX
6. Other Lisp environments for composition

An intrinsically open language like Lisp encourages embedded languages.
(Curtis Roads)

6.1 What is it about Lisp?

Lisp stands for **lis**t **p**rocessing. The only data representation in Lisp is the list. A list has items of any data type that are enclosed by parantheses. A typical data process in Lisp is recursion, i.e. a function that recalls itself with arguments from its last call. A rather unique property of Lisp is its liberal style in handling data. There is no distinction between data and programs. Therefore functions can be arguments in other functions for example. Further, data representations have not to be completely defined from the beginning.[445] Therefore Lisp is often regarded as a useful tool for programming of AI problems. Lower levels of abstraction can be taken for granted and remain unspecified at first. Hence, reflective and non-hierarchical processes are often modeled well in Lisp, and developed incrementally. This kind of programming is called proto-typing or explorative programming. In situations where a solution to a problem is not clearly defined or where the problem itself is not even understood completely[446] in an early stage of a project, proto-typing languages like Lisp can be an effective choice. Art projects are typically open-ended processes, where "solutions" are the result of multiple trials and errors[447]. Machine Composition systems are therefore often programmed on proto-typing Lisp platforms.
From a user or composer perspective, there are two types of environments built on Lisp.
Firstly, environments that are built entirely on the Lisp level, i.e. where the user interacts by defining and calling Lisp functions directly. Secondly, environments that are programmed in Lisp, but interfaced graphically to their users. In this chapter we will concentrate on the first type (like CM), because such systems permit composers to unload the full power of Lisp programming. Systems of the second type could be programmed in e.g. C, and that even without much changes in their functionality from the user stance (composer). For example, Cypher that is literally written in C, but could be re-programmed in Lisp as well, just as EMI (programmed in Lisp) might be translated into C without changing their respective functionalities. The reason one chooses one programming language instead of another is really not tied to its expressive possibilites, but its style and ways to do it. This is the result of the Turing-compatibility of all machines and programming languages today. Any program can be programmed in other languges as well. But some programs do certain things a lot easier, shorter or transparently than others. Therefore choosing platforms, languages or environments can have significant user related consequences to productivity.[448] Lisp has certain qualities that we will look into below.

Therefore will we in this chapter confine ourselves to the applications where composing is done in Lisp or within a Lisp-like style. Some of the environments that may be realized in Lisp but where the actual controlling or composing is done on higher (non-Lisp-like) levels will be treated in **[ch7].**

6.2 What is CommonMusic [CM]?

CommonMusic is a major music composition environment. It represents musical structure on different levels and converts or exports its contents into various of the existing sound control protocols (MIDI e.g) and sound display schemes (notational symbol systems). CM was initiated in 1989 by Rick Taube (around the same time as MAX), and was stimulated by the upcoming of affordable sound synthesis boards for personal computers. From its very beginning CM was intended to facilitate control not only of sound events but of micro-sound-features as well, i.e. sound "shapes" (*envelopes*) and "tone colors" (timbre), typically involving the control of the sounds' synthesis processes (CLM below).

MIDI-files are *one* of the possible output formats, CM can generate. Other formats aloud the control of DSP-hardware or other specialized sound production systems[449] directly, i.e. from within the general control language of CM. *Csound* is the part of the control language that "operates" sound synthesis system, most commonly the native DSP56000 in NeXT-machines or compatible external sound-boards natively (at low levels). The sound synthesis platform of CLM is tightly integrated with CM. Its purpose is to build or design *virtual instruments.* This is done by defining and connecting components (*opcodes*). Opcodes can be classical oscillators, envelopes or sound samples, but also MIDI functions or any other less-conventional schemes that are able to define sound structures.

Csound instruments (*orc*) are in turn played by MIDI instruments, or may react to external sound samples coming from microphones for example. Finally, they may also be entirely controlled by a CM program. The tight integration of CM with a sound generating language (Csound) has historically been and is still today one of its substantial and defining properties and strengths. Some composition languages delegate their final actualizations to performance agents (not unlike composers that let musicians perform their compositions). In contrast, CM is typically used in contexts where computers both generate and perform the musical ideas and compositions.

From a user's perspective, CM is a vast set of *standard Lisp functions* grouped into libraries. As a consequence almost any dialect of Lisp and even Scheme can run CM code. The homepage of CM lists downloadable and free[450] libraries and resources that are necessary to begin with CM and Csound.

6.3 Structures and processes in CM

A Lisp function is formulated as a sequence of several expressions. A typical CM procedure (or Lisp function) is exemplified by the following definition:

```
(defscorefile (play nil pathName "OurGeneratedfile")
        (with-part Pluck-Poly (name "instrument")
        (setf duration 1)
        (setf rhythm (item (rhythms q q e e q)))
        (setf freq (item (notes c4 b3 a a f) :kill T))))
```

This function (*defscorefile*) writes and returns a scorefile, that defines its result, or more concretely the notes in the resulting composition. Its first expression ("*(play nil ...)*") or more precisely sub-expression since defscorefile itself is an expression as well[451], is an option list that tells us two things. First, it determines that its scorefile is not to be played after generation (value *T* would

reverse this setting and initiate playing after returning it). Second, it binds a name ("*OurGeneratedfile*") to an explicit-made path in the file system that identifies the generated scorefile for future references.

The next expression is a macro (*withpart ...*) that creates the first part of the scorefile. Parts are any groups of sound events. Parts may be viewed as equivalents to the staves of a traditional partitura, but are more flexible and general in their possible uses (representations). A part may consist of monophonic or chordal note groups, depending on which instrument type of Csound is used (in the example specified as *instrument*[452]). This distinction derives from synthesis specialization where monophonic instruments need more performance variables for expressive nuances.[453]

Part definitions consist of slots that are inherited from higher classes or their parent class. All expressions that specify properties of notes are slots of the with-part macro.[454]

Any part is limited to a single Csound instrument. To write a succession of sounds with different timbre, one has to create multiple parts for each timbre. One can mix monophonic parts and chordal parts to create multitimbral and polyphonic complexity.

After the part-class and option-list (of the withpart!) comes the body of the expression. It consists of expressions that define and produce values for common parameters associated with single note events, such as pitch, loudness, timbre and timing. The first body expression defines the length of the quarter notes to be of 1 second. Timing is defined as *rhythm* that lists or constructs durations of *q*uarters and *e*ights. Pitch is defined in common note names.

Both *rhythms* and *notes* belong to the significant CM-class of *item-stream-constructor*. They create streams, i.e. repeating event sequences, somehow like the repeat symbol in standard notation (:|). But while ' :| ' means "repeat once!" (if not specified otherwise), item streams continue repetitions until explicitly stopped. Therefore the *freq* expression ends with a termination clause, *:kill T),* to limit the stream to one single cycle or period. *Item* is a pointer to the *next* item in the list. Item streams are defined by their data type[455] and their *pattern.*[456] Patterns are *control* mechanisms for generating sequences out of the item set. For example, the implicit defined *sequence* pattern will output *sequentially* until stopped (e.g. by *:kill T*), while sequence patterns *random* and *heap* select items from the stream in specified ways without respecting the initial order.

Item streams may consist of mere numbers as well, mostly out of purposes to abstract some features (example: *(setf x (items 1 2 3 4 in heap)*), where x becomes a global variable). Item streams can be elements in other item streams, thus item streams are *recursive*. With the item-stream-construct we can formulate a very short and single sequence of pitches and combine them recurrently with several different formulated rhythms. In this way, we express structures very econonomically:

```
...
(setf rhythm (item (rhythms q q e e q e e q q  e q q e q q e q e q q q q e e ))
(setf freq (item (notes c4 b3 a a f ) :kill T))
...
```

In other words, the short pitch sequence *(freq ...)* needs only to be written once. We see here the generational power of the item-stream with its seperation of single aspects or parameters of a tone or note, parameters that in our musical traditions and notations are tied inseperable together in the unit of *tone.*[457] The following is an example of a "melody" that uses a reccurent (or "same") rhythm that is combined with melody, or better the different versions of a melody, i.e. transformed and repeated infinitely until stopped after one cycle of the longest stream using *(:kill)*.

```
...
(setf rhythm (item (rhythms q q e e q ))
(setf freq (item (notes c4 b3 a a f  a g  f  f d f e d d h g f e e c) :kill T))
```

```
...
```
We can automize the transpositions by using another setf-expression that changes values of our note list :

```
...
(setf rhythm (item (rhythms q q e e q r )) ;; r is placeholder for a non-sounding event or rest
(setf freq (item (notes c4 b3 a a f ) :kill nil))
(setf  notes ( + notes 1)    ;; pseudo code
...
```

(setf notes (+ notes 1) generates downwards falling transpositions by a halftones for each time we loop through the note list, continuing from the top of the actual pitch space after hitting the bottom ad infinitum since there are no given terminating conditions. The attentive reader will have noticed that *rhythms* and *freq* are of non-identical lengths. This will under looping conditions (patterns) lead to unlike intertwinings and hence a varying of the polyphonic outcome. In other words, we need not really do much more than to create assymetric structures (unequal structure sizes) in order to generate significant and in some cases sufficiently transparent complex sonic outcomes.

Another example of such a dynamic use of value settings lies in the gradual decrementing of the strength or amplitude of the contributing notes:

```
...
;; initialization of amp to 1.0 is done in the option expression of the defscorefile
(setf rhythm (item (rhythms q q e e q ))
(setf freq (item (notes c4 b3 a a f ) :kill nil))
(incf  amp -0,05)
...
```

Generally, a process of CM begins with a definition of a musical idea using the syntax of CM and eventually the expressive and controlling apparatus of CLM (Common Lisp Music; synthesis and signal processing). Evaluating the code results in an intermediary format, the scorefile, that is interpreted by one or several sound output modules to produce the physical sound,[458] or possibly other sound-representing symbols or notation:[459]

COMPOSER ->
CM ->
SCOREFILE
[if (play=T) then ->
SOUND **SYNTHESIS** PROGRAM ->
SOUND *else NIL]*

CM is implemented in Common Lisp's Object system (CLOS), and takes as such advantage of the vernicular structures of object oriented programming. All CM events and parts (created by withpart) are objects that define the meaning (both syntactical and semantical) of output events from the employed output modules. Their meaning is defined by their class or more accurately their class-defining slots. Default values of slots will propagate from class to subclasses (= inheritance in OOP). Events have *time slots* that are incremented by values defined in rhythm slots of parts. Part classes define their respective slots and the higher-level classes (MusicKitParts e.g.) propagate their values to slots like *'Note-type'* (e.g. 'Note-off') and *'Duration'* (unless determined by rhythm slots all the way down to lower classes (parts and events e.g). This OOP-approach of CM makes shure that many variables are automatically defined by default values as long as the programmer doesn't explicitly defines them, and thereby reduces CM-code to a minimum of desired control of detail. By setting instruments to 'PolyPluck' or 'Simplus', we automatically inherit many standard settings from their higher located classes.

Streams and patterns
In short, the two most important constructions in CM are *structures* and *processes* of streams and

patterns. Not too unlike the main ideas of 'M' [ch 3], data are constructed into streams by item stream generators and controlled by item stream accessors that read elements from item streams of specific data types. We understand how these constructions or paradigms fit well into the Lisp syntax[460] of list (stream) processing (processes). Once we have obtained streams we play them by using specific patterns in execution mode. We repeat once again the fact that CM's resulting streams of events are "played" always *into a scorefile* before their actual sonic realization.

The following example of a recursively defined stream of durations incorporates different pattern types in one single expression:

```
(rhythms (rhythms  q e e h in heap)  ;; [a]
         (rhythms h q e s in cycle)   ;; [b]
         (rhythms e h e e h q in random for 12) ;; [c]
         in cycle tempo 60)  ;;[d]
```

This stream will go on infinitely *[d]*), until externally stopped, it will be repeated as random sequences of *[a]*, or streamed sequentially as wholes by *[b],* and "played" in sequences of twelves in random order with items picked from *[c]*.

Pitch height values of events are represented either as notes (*A3*), pitches (*220.0*) or degrees (*MIDI number 57*). They are convertible in any direction using the functions *'note, 'pitch, 'degree* respectively, e.g. *(pitch 57)* returns *220.0*. This is an important feature of CM to guarantee flexibility in terms of existing, and user-defined formats.

Sound events in CM can be shaped timbrally by wave-lists and overall dynamically using envelope functions for manipulation of line segments or envelopes of single sounds. Single sounds are in this terminology *streams of samples* etc.The distinction between micro- and macro-sound by Roads [2.5] is very much reflected in the control languages in CM and CLM. We will look into some examples to concretizise this relation further.

6.4 Example of CM and CLM

The preceding section contains only small fragments of CM code. We now look at more complete examples with selected comments about their structures and workings:
The example of our first sound definition is the programming of a synthesized sound using only libraries of CLM:

```
(definstrument simplus (start duration frequency amplitude)
        (let* ((start-sample (floor ((* start sampling-rate)))
                (end-sample (+ start-sample (floor (* duration sampling-rate))))
                (sinewwave (make-oscil :frequency  frequency )))
                (Run ( loop for i from start-sample to end-sample do
                ( outa i ( 'amplitude (oscil sinewave))))))
```

We see that we smallest element in this definition is the oscillating sample, paranthesed by their start- and end-samples, like lisp items in a list. This is the atomistic style of sound synthesis in CLM and its Lisp-functions. To call it from CM we will apply it in a *with-part*-construct as part of a *defscorefile*, pretty much like we saw in 6.3. This is still a very basic formulation of a user defined instrument. We use the new instrument **simplus** instantly in the following CM program substituting built-in instruments such as MIDI-patches or *MusicKitParts* (like e.g. *PluckPoly* from 6.3):

```
(defscorefile (play nil pathName "OurGeneratedfileForSimplus")
        (with-part Simplus (name "Simplus")
        (simplus  0 1 440 .5)
        (setf duration 1)
        (setf rhythm (item (rhythms q q e e q)))
        (setf freq (item (notes c4 b3 a a f) :kill T))))
```

The next example of a CM program stems from the help files of CM and plays a simple melody based on three notes, applying the Pluck instrument from SynthPatch in Poly mode:

```
(in-package :common-music)
(in-syntax :musicKit)
(defscorefile (pathname "score-heavy")
        (with-part PluckPoly (name "guitar" amp .5 decay .2)
        (setf rhythm (item (rhythms e-T16 e e+T16 tempo 320)
        (setf amp (item (amplitudes .4 .45 .5 .55 .6 .65 .7 .75 in palindrome )))
                        ;; palindrome is a special pattern
        (setf bright (item (items .2 .3 .4 .5 .6 .5 .4 .3 .2 .1))
        (setf freq (item (voicings [0 3 7] [0 7 11] [0 4 6] [0 14 24] in heap from
          (steps 1 2 3 4 5 6 from 27 for 8) returning note ) : kill 8)))  ;; stop after 8 events
```

Score-heavy constructs a score-file that is played with tye Pluck instrument of the musicKit (using halfway loudness and some limited decay) by combining sequences of rhythm and freq, that are varied with different types of patterns according to the formulated sound parameters.

6.5 CM compared to MAX

We now recall the other Machine composition paradigmatic tool, MAX from **ch3**, and put CM and MAX in a comparative light. There are both common and idiosyncratic traits. We start with the obvious *commonalities* of CommonMusic and MAX:

- **General purpose** musical composition environments.
  Both are general purpose environments to design and produce musical compositions or any other systematic sound constructs. In contrast to older tools, mostly from academic sources, no special "aesthetical direction" is implied and embedded in their functional architecture. They are general purpose composition tools.[461]
- **Object-oriented**
  Max is object-oriented from the ground. Its architecture around messages and box objects is both computationally and representationally (graphical user-interface) based on objects. The encapsulation methods of MAX are intuitive and part of their basic user paradigm. So is CM with its CLOS[462]-based and open-ended design. Somehow like Lisp-functions define objects relating input to output in special ways, do messageboxes transform and control the incoming messages to become transformed outgoing messages (lines). This makes both very much friendly tools for programming in explorative ways. Just as we can postpone certain processes in Lisp programs, we can use preliminary zero-patches or zero-messages (like identity functions) to defer their detailed commands to later completionary stages.
- **Extendible language**
  Both MAX and Lisp are well-known for their contributing user-communities and their natural ways to import code from libraries or foreign code and languages.
- **Multiple representations** of structures and sound
  Both environments are open to multiple forms of data representation and sound representations. Their many import/export formats underline this liberalist attitude that intentionally gives priority to the emergence of a variety of design philosphies rather than constraining their users into certain rational standards of elements and structures.

- **Interface-friendliness** towards many sound synthesis packages. MAX uses its own MSP signal processing tool kit. CM is tightly integrated with functions from the CLM libraries for sound synthesis. But nearly any other libraries and sound tools are possible to integrate with both MAX and CM.

But there are differences too, and some are definedly relevant to choosing one over the other. The most important *differences* between CM and MAX are:

- Max is written in **C** programming language. CM is programmed in *Lisp*.
- Max is processing sound results in *real-time*, while CM produces *intermediary* scorefiles that are sonically actualized (played) after completion of writing the scorefile.
- Max has a graphical interface. CM has no special interface of its own. It has a rather programmer-style mode of interaction (interpreter or compiler).
- CM has its control language formally seperated from the production language of CLM. The intentionally distincted formalisms can lead to a more orderly program structure, and ease debugging and understandability. An advantage with non-real time modus (in contrast to MAX and MIDI for instance) is that it is *not* necessary to group notes syncronously, i.e. timing and hardware issues are less prone to interfere with functionality.

This is certainly not a complete list of the many nuances distinguishing these two tools, but we will refer to this subject again in a more practical context of use, when we discuss the combination of both tools using the embedded version of CommonLisp (minus the expressional and sound-specializing apparatus of CM/CLM) in **ch8**. We append at last an example that reverberates somehow the echo-example from **ch3** implemented in MAX. As we may recall, the “melotonic” echo repeated several times an input-stream of pitches in variated forms and decrementing loudness. We only begin the process of building such an echo in CM and imagine how the input of pitch (“melotonic”) material, basically freq streams from an external user, might be fetched in a CM program. Its value for *bright* is feebly decreased (setf, -0.1) in each succeeding echo and its loudness inversely related to brightness and hence changed as well.

```
(defscorefile (play T pathName “OurGeneratedfileforEcho”)
        (with-part PluckPoly (events 4 rhythm 0.5 bright 0.3)
        (incf bright -0.1)
        (setf amp (/ bright 2))))
```

This sketch of 'echo' in CM reminds us of the obvious and fundamental difference between MAX and CM. MAX is interactive and easy to interface in real-time; CM is more planned and less real-time-based.

6.6 Other Lisp environments for composition

OpenMusic originates out of the 'PatchWorks' environment from IRCAM. It has a visual programming interface to Lisp and CLOS. Objects are viewed graphically as icons that are repositioned and connected on screen. These icons or boxes are not too unlike the boxes of MAX.

Compo is a language that borrows from Lisp syntax but simplifies it to describe musical structures in more natural ways, e.g. will *(note (:c)(:e)(:g)(:e)(:c :h))* generates a broken triad.

Nyquist is a Lisp-based language for music composition and sound synthesis.

Symbolic Composer: in its latest versions 5.x, Symbolic composer has become a very large and high-level language for composition. It is fully extensible with functions for CommonLisp. It is loaded with extensive objects, e.g. data of scales and chords. It contains over 300 microtonal scales that meet the material needs of any conceivable music tradition throughout history. Additionally, many of the latest compositional strategies, that are in this paper covered in **ch7**, such as mathematical tools for generating chaos, fractals and number-theory-utilities are included in this swiss-knife for music composition. Genetic and neuronal programming is included as well. In total, more than 1000 algorithms can be connected in vast processing networks. Symbolic Composer is commercial software.

Chapter 7
New Directions of Composing Machines

Paradigms and strategies

1. Machine composition so far
2. Models from Mathematics
3. Models from Psychology
4. Models from Physics and Artificial Life (ALife)
5. Models from Biology : Evolutionary Music and learning
6. Discussion: “'Between 'chaotic order' and 'hopeful monsters'”

Musicians intent on creating algorithmic composition systems with the spark of human creativity would do well to adopt this combination of coevolution, learning, and rule-following, and thereby with luck avoid the horrors that were visited upon Victor Frankenstein and his creation.
(Frankensteinian methods for Evolutionary Music composition, 1999)

7.1 Machine composition so far

Ch3 and **ch6** introduced tools for building algorithms and programs in machine composition projects. MAX and CommonMusic are programming languages as well as specialised music composition environments with interfaces to sound synthesis and music structure generation. In **ch4** and **ch5** two of the most prominent composition systems developed over longer time spans in the 1990's are portrayed. They represent very unlike approaches. Cypher is real-time based and interactive. EMI on the other hand, learns style features by induction and generates style-complying pieces that are performed by humans. I classified Cypher as an informed MC system and EMI as an inductive MC system, on the grounds that Cypher is more straightforwardly modeling cognitive structures in the listener's mind, while EMI extracts bits of more superficial[463] meaning (patterns) that are put together or 'recomposed' into new pieces that show enough likeness to qualify as stylized in the same time as they add enough novelty to qualify as genuine repertoire-additions.[464] EMI and Cypher are knowledge-based systems; data-driven by material from a style or better sample collections (EMI) and programmer that informs Cypher about prefered structures.

7.2 Models from Mathematics

Early computer music, as exemplified by Xenakis and Hiller **[ch4]** was by and large highly experimental and tended to generate abstract and complex structures that often demanded too much from a human listener. It sounded 'too mathematical' to a majority of listeners, except to those with a fascination for complexity and its mathematical origins. We might say that those algorithms were not sufficiently anthropogenic or anthropophile, echoing somehow the critiques against the principles of serial polyphony by the New Viennese school. Later there is the attempt to reduce rigidity by using stochastic algorithms and exploring an “aleatoric aesthetics”, examples are Xenakis, Cage and others.[465] As we saw in **ch4** and **ch5**, algorithms in machine composition, in contrast to some earlier practices are *informed by* AI and music psychology and will therefore be more understandable. 'Computer music' developed into 'Machine composition' in the terminology I chose above. But lately, new trends turned again back to formal and mathematical inspirations.

Fractals
Fractals are geometrical objects that seem complex, but are based on simple and recursive definitions. Their structures are *self-similar* at arbitrarily small scales.[466] Formally, fractals are geometric objects with fragmented and rough geometrical shapes and of higher dimensions of Hausdorff-Besicovitch dimension than their Euclidean dimension or topology. Mandelbrot's computerized visualizations contributed to their general diffusion and popular interest.
There are three prominent techniques or superclasses for the generation of fractals:

- escape-time fractals (recurrence relations at each point in a space)
- iterated function systems (fixed geometric replacement rules)
- random fractals (stochastic processes)

Random fractals are the result of stochastic processes, escape-time fractals and iterated function systems are deterministic, but generate *approximately* identical structures as well as *exact* identical structures at their various scales. *Iterated* functions are usually the engine for exact self-similarity. But not in all cases, as we will see when we look at a specific implementation of the iterated function class called *noise generation fractals,* the *one-over-f noise* (*1/f*) that has been applied extensively in the generation of melodies in fractal music examples.

Are fractal classes discovered or invented?
Fractal phenomena are certainly not limited to the domain of pure or mathematical structures. We find them in natural phenomena as well. But the fractals in nature tend to be of the approximate kind of self-similarity. Growing things like plants and trees are rather intuitive examples, where fronds and branches reflect the whole or bigger parts quite well.[467] It seems that evolutionary processes generate fractal structures.[468] But the formulation of *pure* fractal classes is of *inventory* nature as well. The Mandelbrot or Julia sets that generate visually contemplated pictures may be inspired by nature but create new and abstract structures, and must as such be called invented first of all. But other fractal theorists try explicitly and intentionally to model plants or other natural phenomena. We may discern between descriptive, modelling fractals and pure, formal fractals. The many classes of fractals are often named characteristically after their inventors/discoverers.[469]

Non-linear systems and chaos theory
Linear systems in mathematics are both predictable and in general computable.[470] Linear systems are modular in their effects, that means roughly that their behavior can be calculated from the behavior of their parts. Mathematically, this is stated as superposition principle, that says that a linear combination of solutions to the system is also a solution to the system as a whole. Technically, properties of additivity and homogeneity[471] must be satisfied for a system to be classified as linear. Many physical systems in engineering can be modeled as linear systems with simple and closed-form solutions. More difficult systems, e.g. from biology, approximate linear systems and allow acceptable solutions. Non-linear modeling may be in some border cases exact solvable, but mostly they exhibit non-predictable or quasi-random behavior even though all those systems are completely deterministic.

From Wikipedia on dynamical systems:

> This branch of mathematics deals with the long-term qualitative behavior of dynamical systems. Here the focus is not on finding precise solutions to the equations defining the dynamical system (which is often hopeless), but rather to answer questions like “Will the system settle down to a steady state in the long term, and if so, what are the possible attractors?” or “Does the long-term behavior of the system depend on its initial condition?”

We notice here a general similarity to the problems in 'analysis of algorithms' [1.3]. The approximations and evaluations of generative functions in 1.3 and computational complexity share certain challenges as a consequence of difficulties to formalize the respective situations in both fields of applied mathematics to natural phenomena.[472]

A definition of Choas theory:

> The qualitative study of unstable aperiodic behavior in deterministic nonlinear dynamical systems.
> Kellert (1993)

Chaos theory is studying a subset of non-linear dynamical systems that "under certain conditions exhibit dynamics that are sensitive to initial conditions".[473] Such 'chaotic behavior' is *deterministic chaos* and popularly associated with the so-called butterfly effect. They are *not* predictable but often *knowledgable in statistical terms*. Chaos scientists lack a general and complete theory of these dynamic systems but concur about that they should have three properties satisfying the definition of chaotic: they must be sensitive to initial conditions, they must be topologically mixing and their periodic orbits must be dense.
Such formal criteria are not the focus of artists' applications of mathematical models of complexity, they are first of all interested in the 'chaotic' output.

Chapel's Real-time algorithmic Music systems from fractal and chaotic functions
Chapel uses iterated functions of the class of 1/f noise generators for his core model of algorithmic music generation. He examines fractional noises and finds white noise ($1/f^0$) to be too unpredictable and brown noise ($1/f^2$) too predictable for musical interesting contexts. Therefore he ends up experimenting with pink noise fractal builders of $1/f^1$-type (one-over-f-noise) for melodic constructions. These fractal functions are already known to model fairly well certain phenomena in chemical systems, ecological systems and human speech a.o. Therefore when physicists Voss and Clarke found that audio of baroque concerti (like Bach and Vivaldi) had loudness and frequency distributions of almost $1/f^1$ distributions, that motivated Chapel a.o. to choose them for algorithmic compositions as well. Chapel states:

> $1/f^1$ processes correlate logarithmically with the past. Thus the averaged activity of the last ten events has as much influence on the current value as the last hundred events, and the last thousands. Thus they have a relatively long-term memory. [474]

Even if this particular class of iterated functions generates not exact self-similarity, it exhibits *statistical* self-similarity. Others like Nettheim, 1992, have used pink noise generators for pitch sequences *and* harmonic purposes, but with more limited success as well. The reason behind the success of pink noise based music lies in the fact that it generates both *analogy* and *contrast* in reasonable portions that permit listeners to find both similarity (or variation) and surprise in the output. Similarity is taken care of the algorithms that approximate self-similarity, surprise is added by the pseudo-random quality of pink-noise.
But there is more to it. Pink noise generates nothing other than numbers really. They need to be mapped to musical attributes before generating sonic output. In addition, Chapel wants his systems to be interactive during performance. Therefore he maps gestural parameters to the parameters of the mathematical core as well. About this non-trivial mapping process, Chapel writes:

> Mapping is a very empirical subject. A good mapping should intend to conciliate two different areas of human activity, science and art, in a satisfactory way.

Chapel actually creates more than machine composition engines. He combines the realtime algorithmic music core model with interactive control, resulting in what he calls an *intelligent* or

active instrument. The choices made by the composer li.e. in the selection of effective initial conditions and reasonable tuning of pink noises, with the performer using the instruments to tune or vary the output in real-time performances. What emerges is actually a hybrid between composition and performance system, since the performer does more than is usually expected from performers, he contributes in essential ways to the structural development as well. This will be taken up again, in a more prosaic style, among the themes of **9.7**.

Another researcher of such chaos generated systems is Dabby.[475] She starts out with so-called seed-patterns or motives that in turn are varied by iterated functions in the mathematical core.

Both of these systems are player systems in Rowe's classification **[ch4]**. Chapel's systems are of the generative type and Dabby's system belongs to the sequenced type of Machine Music models.

We should remind ourselves of the relative immaturity in this field. Non-linear dynamical theory in mathematics and complexity sciences in general are fields with ongoing research, scarcely canonized, and often of partial knowledge only. Therefore the application of such formalisms in music composition will be of experimental nature:

> Mere translations of nonlinear dynamic systems into music will not, of course, achieve this end: something will have to be done with them if more convincing results are to be produced: Once again “purely formalized procedures are not in themselves composition” (DiScipio, 1990). Though math and science may aid in providing ideas to the arts, their influence is by no means a secession of creativity on the part of the composer. Systems may produce convincing and beautiful music (algorithmic composition has provided many examples in the last twenty years), but somebody must still invent the systems.[476]

7.3 Models from Psychology

Psychology of music is rooted in psychology and musicology, but draws on many other fields as well, notably in recent times, the neuro sciences and ethnology a.o. The history of psychology of music is in fact as old as both philosophy and music themselves. The normative and theoretical discussions of Platonism and Scholasticism about the psychological effects of music had deep impact on the history of music. “Platon's musicology” sorted out certain instruments and scales as dangerous, others as useful, to the education of the young (understood as character building). Such ideas of *musical engineering* continued well into the ages of Renaissance and Baroque, and are now redirected from normative to communicative goals. The objective nature of Descartians[477] emotions was now susceptible to rational inspection and scientific description. The 'Doctrine of Affections' applied ideas from language and rhetorics to music theories and the new 'musical language'.[478] In a famous controversy between Rousseau and Rameau in Paris, Rousseau critizised “French music” for favoring harmony to melody. Rousseau (and actually most revolution-friendly writers at the time) favoured melody in the name of the creative spirit and its expression of independence over the adhesion to rigid rules of harmonic tradition (polyphony) or other lawlike doctrines (doctrine of affections). Most historical theories of music psychology did not clearly separate issues from psychology from the ones from philosophy or aesthetics.[479]

Modern music psychology is primarily empirical, its knowledge comes from interpretations of systematically collected data about musical behavior and experience. It gathers insights from many other fields, as e.g. neuropsychology, biopsychology, the cognitive sciences, AI, and investigates wide-ranging topics about the nature of preferences, psychological processes during listening (and performing), as well as the structures involved in them. Major trends in todays music psychology are the neuronal, cognitive, ethnical and computational perspectives and research methodologies.

They are all relevant to compositional methods and practices. Modern classics are Leonard Meyer's 'Emotion and meaning'[480], Diana Deutsch's 'The psychology of music',[481] John A. Sloboda many publications and last but not least Lerdahl and Jackendoff's generative theory of tonal music (GTTM).[482] The influence of GTTM on compositional systems has alread been reviewed in **ch4** in relation to Cypher and others.

In 2001 the linguistic approach of GTTM was converted into a consistently computational framework (*Melisma Music Analyzer*) by David Temperly in his 'The Cognition of Basic Musical Structures'.[483] Temperly builds his models of cognition and listening on the established preference rules approach[484] (generating many, choosing the prefered or best) like GTTM. But where GTTM remained somehow qualitative only and was often criticized for being too vague, the computational version of Termperly proposes precise numerical weights for the parameters of his "updated GTTM". This makes it more testable and falsifiable, and more rigorously in its formulation as well:

> If a preference rule systems is proposed for an aspect of structure, and one finds a situation in which the preferred analysis cannot be explained in terms of the proposed rules, than the theory is falsified, or at least incomplete...
> If a preference rule system can be made to produce good computational results, it provides an elegant, substantive high-level hypothesis about the workings of a cognitive system.[485]

Temperly uses dynamic programming methods with *best-so-far* analysis and *backtracking* known from AI. He describes thereby listening as an retrospective, revisionary or tentative activity.[486] In his ninth and tenth chapter Temperly discusses ambiguity and expectation in a psychological as well as computational tractable perspective. This integration with computational topics is increasingly becoming one of the defining traits of modern music psychology, in addition to its inclination towards experiment-based knowledge. Music psychology has long orbited around phenomena of listening and lately also performing. Temperly's book is certainly no exception to this rule. But he hopes that "many aspects of music cognition may be common to all of these activities",[487] including composition. And in his eleventh chapter, he discusses matters of style in relation to preference rule systems. Different styles will reflect different preference rules (PR) systems.[488] He then proceeds from rules that model the properties of a piece (used in listening models) to claims about the process of composing with them, i.e. using these rules as compositional constraints. Temperly arrives here to conclusions known from other theorists like Meyer when he states:

> Rather, it might be more accurate to say that pieces are judged as grammatical and acceptable when they fall within a certain range of scores. A score too low indicates excessive simplicity and lack of interest; a score too high indicates incomprehensibility and ungrammaticality.
> As well as providing a basis for quantifiable claims about musical styles, then, preference rules scores may also account for judgments of acceptability within a style.[489]

Temperly's scores are certainly always relative to the particular PR system's style defined in the first place. No absolute musical value can be inferred from particular systems scores, only the internal style value standing for its "complexity" (Temperly calls it structural comprehensibility). This places them on a line between *boring straight* and *uncomprehensible strange*. A composer aiming at a sense of chaos and confusion in a listener will choose lower scores of their PR system, while a constructor of shopping music might aim at a rather high score. In this general sense, the PR approach is relevant to questions about musical value and aesthetic satisfaction after all **[9.5]**. Because Temperly's theory is quantified, he can apply his analysis further and partition the domain between too boring and too weird further; relative to a certain PR system or style. He divides the style-complying range of acceptable scores i.e. middle area, into calm and typical pieces (bordering boring) and tense and unstable, i.e. daring, pieces (bordering the weird or ungrammatical).[490]

> I suggest, then, that the scores produced by preference rule systems for segments of a piece may be a good indicator of musical tension. Within the range of acceptibility, higher scores are associated with an effect of lower tension; lower scores mean higher tension. If we also include the range of unacceptably high or low scores, this produces a **continuum from boredom to normality to tension to incomprehensibility** (fig.11.11) Ways that other preference rule systems relate to musical tension can readily be imagined. A heavily syncopated passage creates metrical tension, since either the regularity rule or the accent rules will be violated; a line with many leaps creates contrapuntal tension. A passage with a great deal of chromaticism stretches the key rules, since it probably will not permit a good fit with any single key-profile (at least not without frequent modulation). A passage without clear phrase boundaries will convey tensions by not permitting any satisfactory grouping analysis; this is a common feature of transitions and development sections, as noted in chapter 3. When several of these kinds of tension are combined within a single passage, the effect can be dramatic.

Another interesting clarification of interest to composers are the concepts of ambiguity and tension. A PR system may assign similar score values to several preferrred analyses and induce ambiguity. A preferred analysis with low score will generate tension. Then, a PR system with low scores for several preferred analyses will produce both ambiguity and tension.
Composer's interest lies specifically in interpretations of scores with respect to the overall-structure of a piece (large scale tonal tension e.g). This level made part of GTTM, but Temperly decided to not include it in his system[491].
Temperly understands the concept of meaning in music as somehow identical to function.[492] And the diverse functions of infrastructural levels are intertwined and interdependent in ways the system his preference rules are able to sort out. He looks for ways that PR systems may shed light on high-level aspects of music, such as musical tension and schemata, narrative and drama:

> Through harmonic and tonal structure the common-practice music has the ability to create complex, multi-leveled journeys. These journeys also carry complex emotional associations, related both to the tonal center and pitch collection used, which can in part be accounted for by the positions of these elements on the line of fifths. Motives – repeated patterns of rhythm (reinforced by meter) and pitch – are also an important aspect of common-practice music, and rely crucially on several aspects of the infrastructure. As other authors have noted, motives often serve as "agents", entitities with feelings, desires, and the capacity for action. Combined with the expressive powers of the tonal system, this allows composers to create complex dramatic narratives.(...)
> Metrical and harmonic analysis, along with motivic analysis, are also part of a complex process of searching for order and pattern in a piece – that is, of "making sense of it" - which has inherent appeal, analogous to many other intellectual activities.

Another major contribution to modern music psychology (and music philosophy) is found in Gareth Loy's double-volumed 'Musimathics'.[493] In his chapter nine about composition and methodology, he goes into topics, mathematical as well as psychological, as diverse as randomness, chaos, combinatorics, composing functions, probability information theory, Markov chains, learning , connectionism and theoretical aesthetics; all areas very much relevant to composers that we will revisit in considerably more detail in **ch9**.

Other interesting recent titles to composing are:

- **Music and Probability**[494], David Temperly; MIT 2007.
- **Musical Communication**, Dorothy Miell a.o.(ed.); OxfordUP 2005.
- **Ways of Listening: An ecological approach to the perception fo Musical Meaning**[495], Eric F. Clarke; Oxford UP 2005.
- **Suspensions of Perception, Attention: Spectacle, and Modern Culture**[496]; Jonathan Crary; MIT 2001.
- **Music and Memory**, Bob Snyder; MIT 2001.
- **Music, Cognition, and Computerized Sound**, Perry R. Cook; MIT 2001.
- **Sweet Anticipation: Music and the Psychology of Expectation**, David Huron; MIT 2006.
- **The Cognitive Neuroscience of Music**[497], Isabele Peretz a.o. (ed.); Oxford UP 2003.
- **Tonal Pitch Space**[498], Fred Lerdahl; Oxford UP 2001
- **Empirical musicology: Aims, Methods, Prospects**,[499] Eric Clarke a.o. (ed.); Oxford UP 2004.

- **Hearing in Time: Psychological Aspects of Musical Meter**[500], Justin London; Oxford UP 2004.
- **Origins of Music**[501], Wallin Nils L.(ed.); MIT 2001.
- **Conceptualizing Music: Cognitive Structure, Theory, and Analysis**[502], Lawrence M. Zbikowski; Oxford UP 2004.
- **Embodied Music Cognition and Mediation Technology**, Marc Leman; MIT 2007 fourthcoming.
- **The Musical Representation: Meaning, Ontology and Emotion**, Charles O. Nussbaum ; MIT 2007 fourthcoming.

Which particular MC systems are using models from psychology? In a broad sense, every system of composition will have to take care of psychological realities, and hence must take some kinds of theoretical constraints in consideration. In a more explicit sense though, both EMI **[ch5]** and even more so Cypher **[ch4]** are fairly formalized models of psychology and perception. Both are *knowledge-based* systems that model the *competence* of a human composer, more specifically the relevant brain capacities and cultural capabilities of a music composer. At their conception, knowledge in these fields was sparse and immature. Rowe and Cope had therefore to find many of their strategies pretty much by themselves.[503] With the growing literature, and its to some extent[504]even collectively reuse of knowledge[505] that has emerged in the last decade, one may expect significant progress and convergence in the attempts to build MC systems informed by knowledge from music psychological models and methods from AI to solve these challenging tasks.

7.4 Models from Physics and Artificial Life (ALife)

Xenakis, one of the classic composers with mathematical tools likens behavior of kinetic gas with music:

> The basic principles of kinetic gas theory, which are described by statistical mechanics, are very simple and very general. They can be found in music as well.[506]

In a similar vein, Artificial life (ALife) looks at how many interacting simple parts can generate complex and life-like behavior as a system. One of the pioneers in this new and "utopian" research is Valentin Braitenberg and his 'Vehicles'.[507] He starts in his "experiments of synthetic psychology" with very small machines that consist of one or a few motors and sensors, connected either inhibitory or actuating. What Braitenberg demonstrates is that very simple machineries are able to exhibit complex behavior if their control and motor architecture are well adapted to their environments.
Already in his fourth vehicles, amusingly called "Values and Special Tastes", Braitenberg installs threshold-connections instead of linearly functioning motors and sensors, used in lower level vehicles. Braitenberg describes the apparent instinct-like behavior and sums up laconically:

> Whatever their origin, thresholds in some behavior patterns make a lot of difference in the eye of the observer. These creatures, the observer would say, ponder over their DECISIONS. When you come close to them with a lure, it takes them some time to get going. Yet once they have decided, they can act quite quickly. They do indeed seem to act in a spontanous way: none of these passive being attracted one way or the other that was so obvious in the vehicles of the more lowly types. You would almost be tempted to say: where decisions are being made, there must be a WILL to make them. Why not? For all we know, this is not the worst criterion for establishing the existence of free will.[508]

Complex movements of swarms[509] (as e.g. studied in birds) have been applied to music at the Santa Fe Institute. In an interactive music improvising system, Blackwell and Bentley (2002) have used these ideas. Peter Beyls[510] continues such work in his "Molecular (object oriented) collision model of musical interaction in real-time. He starts out from general cellular automata research and

interprets interacting cells in a CA network as particles or molecules. This idea has been called the metaphor of ballistic computing referring to the billiard model (Ed Fredkin). And Beyl applies such a framework to music by redefining balls as musically acting particles holding a single variable for their angle of movement.[511] Such an atomic model reduces complexity to a core. Beyl calls his model "the imaginary physis of the musical world", described by a set of two-dimensional arrays. Those arrays control the paths of particles after they hit others or are hit by others as they update their respective angles of movements. An autonomous genetic algorithm **[7.5]** optimizes this dynamics and produces greater diversity. The system is interactive in the sense that the genetical algorithms will not control the ballistic theater alone, but are co-controled by gestures from an external user that interferes or deviates the course of the musical particles. The results of this theoretical model are finally mapped to MIDI parameters to allow sonic instantiations.[512]

> Actually, a particular type of man-machine cooperation emerges; the GA clearly aims optimization by creating arrays that often look quite similar; they converge to some spot in genetic space. On the other hand, user activity both selects and modifies some arrays, often profoundly disturbing the GA instantiated structures. The system thus permanently fluctuates producing patterns of variable regularity. User initiated actions have two effects; they tend to favor short-term disorder while also influencing long-term behavior since some of the modified arrays wil survive in the next generation. Machine initiated activity can be described as background autonomy according to the single criterion of global fitness...
> ... genetic algorithms function as generators of surprise in an otherwise stable system. For musical continuity, we rely excusively on the arrays. The system as a whole exhibits a certain inertia... in other words the perceived melodic continuity is an implicit byproduct of the global systems behavior – it does not result from any form of melodic memory. Consequently, coherent melodic form issues from the accumulative forces of explicit physical gesture and implicit genetic evolution.[513]

Formally, a system is a tuple {M, R,A}, with M denoting the finite number of building blocks (molecules), R denoting all possible interaction rules and the algorithm A that is equivalent to the sensitivity matrix describing interaction thresholds between molecule types inside the system. Like it would be the case for billiard balls, values of A are symmetric. Non-symmetric values would produce higher non-linearity in the system. The interaction rules R state how *n* different types of molecules will interact once paths are crossing. R is an array of *n* by *n* elements.

This abstract model of patterns is then mapped using MIDI drivers of CommonMusic **[ch6]**. The mapping is leading to note events derived from rules of molecule interactions, i.e. their angle of movements. Even though the sonic output very much reflects the particular mapping choices, the structural nature or "soul" of this systems lies in the *emerging* properties of the *unmapped* collision model underneath. It is precisely this systemic and emergent nature that ties Beyl's collision model to ideas of artificial chemistry[514], artificial life and thinkers like Prigogine[515] and Braitenberg that have explored complexities coming out of very simple seeming systems with basic agents and straightforward and elementary actions. Beyl concludes :

> "it proves that the economy of expressing simple angular relationships in an array does indeed suffice to support interesting non-linear dynamic behavior. "

Swarm Music is a real-time accompaniment system[516] of Tim Blackwell who describes it :

> Swarm Music produces musical improvisations with a swarm of musical events. It does this by exploiting the self-organisational properties of swarms. The musical events organise into a swarm-like shape to produce melodic, harmonic and rhythmic patterns. Swarm Music is interactive: external musical events from humans (or other swarms) are captured and positioned in Music Parameter Space as attractors. The swarm is drawn towards these attractors, converting spatial patterns into music.[517]

A more composition-oriented system type is "DNA-music". These systems seem again to be inspired by Pythagoreaen thoughts of unity.[518] The basic idea is to use literal genomic information

(e.g. from the human genome) as musical raw material before they use mapping rules or functions to translate DNA bases (nucleotides) into values of pitch and duration. Hugh and Barton's mapping splitted DNA sequences into groups of four and interpreted the first pair as pitch information and the last pair as duration information, thereby "harvesting music from different combinations of the four letters". The sonic results are characterized by a certain dreaminess and "eerie but soothing qualities" [519] The results are then post-processed through expansions and contractions, as well as selections.
John Dunn[520] converted characteristic features of amino acids (molecular weights and volumes) to musical features (pitch and duration).
The question here seems to be how much "similarity" there is assumed to be generated in *generous* listeners without any basis in the objective data. If all these systems do have in common are some kind of repetitions, we may after all talk only about structural phenomena without need to go much beyond. I believe DNA-inspired music belongs not really to the most interesting paradigms of MC, since its "intelligence" *lies only in the mapping* and not in the data. It may be better termed a quasi-compositional approach. Its connotations point nevertheless to biological implications, which I believe are unwarranted, something that will emerge naturally when we look at seriously biologically motivated and influenced systems in the next section.

7.5 Models from biology: Evolutionary programming and learning

The preceding section was about strategies to exploit natural processes or structures without sonic content by converting or mapping parameters of these external models to musical models.[521] In this section we look at learning of individuals or evolving of populations as a model for composing systems.

We know from **ch2** that learning is a component of composing. Humans learn by an adaptation process of neuron networks in the brain. These neural networks are modeled in AI as artificial neural networks (ANN, **ch1**). Computers work much faster in their sequential switchings, but brains make up for for it with essential parallelism and they are therefore billion times faster. Brains are more fault-tolerant, manage novel input better, 'degrade gracefully' (don't stop working under difficult conditions) and learn inductively (open-ended), compared to machines.
AI has long traditions and experience for building ANNs dedicated to solve simpler perception in problems or decision problems. Knowledge is encoded (represented is less appropriate here) in the weights of connections between simple processing units in a parallel distributed processing system PDP. Such connectionist models of brain activity or cognitive processes are quantifiable in their connection strengths, but do not represent knowledge in discrete parts, e.g. there is no specific localized part of the network that handles A# pitch classes or 5/4 meter. It will always be the whole system that "represents" a response to a stimulus. ANNs can be feed-forward networks without loops and typically have hidden units between input and output units. Hidden units mostly contribute to the indexing of non-linear functions from their input values. The emerging proto-intelligence of making decisions comes from the ability of non-linear functions to respond to quantitative input with qualitative output. With other words, continuous input can be output in the form of as structures (or qualitative) e.g. turning something on or off. Another central feature of multi-layer feed-forward ANNs (hidden unit layers) is back-propagation. Back-propagation is computationally interpreted as a gradient descent in the weight space towards more optimal output[522].

> Back-propagation provides a way of dividing the calculation of the gradient among the units, so the change in each weight can be calculated the unit to which the weight is attached, using only local information. Like any gradient descent, back-propagation has problems with efficiency and convergence. (AIMA, 582)

> The trick is to assess the blame for an error [non-perfect behavior] and divide it among the contributing weights. (AIMA, 579)

The important feature of ANN learning is that they need *not* to be told how to do something but only *what*. Training ANNs means to reward better outcomes and inhibit or punish worse outcomes. How can such artificial systems be made to compose like human brains? Patterns input to an ANN can for instance be completed by its output. By training the ANN with selectively rewarding good melodic continuations (to certain patterns), the systems will learn to generalize, roughly stated. Peter Todd[523] did this with folk melodies as input, and generated novel melodies from his ANN. The difficulty of upscaling ANNs for larger scale composing of structured pieces of some interest, has been a major criticism about the value of connectionist models for MC altogether[524].

Machine learning in symbolic AI is another way to think of learning artificial composers. AIMAs general model of learning agents **[2.2]** consists of effectors and sensors that operate and sense relative to the agents environment using a performance element. In addition there are learning element, problem generator and critic. The crucial idea is that an agent learns not only by responding but by trying out actions and maintaining a model of the environment in the learning element, a model that is changed by the agents own critic. The problem generator will then generate proposals for tests in the environment. This model is easier to control and understand than ANNs above. We can e.g. program some prior knowledge into the learning element of the agent to speed up the learning process. But many applications of this general model are exploring more unknown environments with an autonomous agent.

In the field of machine composition, a more specialized model of machine learning has risen to popularity: genetic algorithms or programming[525] (GA). The idea is to use a start population with some tunes, that are tested by a fitness function (similar to the critic above) to find the best (complying) tunes. The best tunes are kept for breeding new tunes by recombination. This next generation will again be tested and so on.

For compositional systems, the four decisions to be made for applying genetic algorithms are :

1. How can the fitness function be defined?
2. What representation is appropriate for a musical unit ?
3. What are the selection criteria in addition to the fitness score?
4. How do the musical units reproduce?

From biology we know that genes are units of reproduction through processes of cross-over and mutation a.o. and these units are tested for their fitness to decide their survival chances. In the history of music we know that single composers have been reproduced by music schools and teachers, tested against an audience that functions as critic or fitness function for determining the successful composers of the next generation. Successful composers are often influenced (cross-breeding) by other composers and other external conditions (mutations). This darwinistic perspective on composition history is also fertile in observing the output of a single composer. He might propose 10 compositions, but is only rewarded for two of them. Then he takes those two as a major *inspiration* for recombining siblings of them. So the general public or audience acts as fitness function relative to the populations of composers and the populations of works by single composers. Another example is the top listing industry of popular music in the media. Musical trends are reinforced by the policy of radio and tv stations to play songs logarithmically more often the higher their current place on top lists. In other words, when listeners buy songs they act as critics rewarding not only this particular song, but even facilitating trends based on top placed songs as a whole. We may continue this interpretation for both arrangers and composers[526] (as well as improvising musicians) where they internally test multiple solution proposals to find the best one

according to their personal criteria.

In a general and algorithmic description, an evolutionary framework will loop in the following order:

- an excessive number of variations is generated (**candidates**)
- candidates are measured against fitness criteria (**score**)
- candidates are excluded/selected resulting in a reduced number of successful selections (**selection**)
- successful units are reproduced mixed and varied into bigger numbers (**reproduction**)
- ...(**back to start**)...

Many of the most recent and original contributions to machine composition in the last decade are placing themselves at least to some degree into this new paradigm of biological or genetic programming. It seems more or less natural to look at MIDI sequences (made of zeros and ones) as chromosomes for genetic breeding. Moroni e.a. defines the overall fitness function as a function of multiple fitness criteria, like harmonic fitness, melodic fitness and voice range fitness.

In a major article of Todd and Werner, "Frankensteinian Methods for Evolutionary Composition",[527] evolutionary music composition is surveyed. They "follow in Frankenstein's footsteps" when they construct machine composing systems with living qualities that are under some form of evolutionary control or guidance. They present a system where composition populations evolve through sucessive generations submitted to modifications and reproduction. The more "fit" or better desirable compositions are, the higher is their probability to be aloud to breed with each other and so on. This fitness function or *evaluator* is similar to the critic in the learning model, from above, and can be performed by humans (as in connectionist training sessions), rule-based **[ch4]**, learning (**[ch5]** and ANNs) or finally by evolving systems themselves. This last approach is the most radical hitherto taken in MC, since it simulates evolution on both sides of the interplay between variation (reproduction) and selection (critic). Humans are mostly removed after setup of this virtual world that should remind us of the models from ALife mentioned in **7.3**.

The authors start with reviewing existing systems where critics are human. Human critics may be single persons or groups of persons. In these cases, however, practical problems associated with the concept of fitness bottleneck, stubbornly remain. Critics have to do time-consuming judging. In addition, we have to rely on intuitive or personal taste(s), somehow arbitrarily chosen as aesthetic trainers of the system. Therefore some systems[528] substitute the human critic with a rule-based critic to automate and speed up the selection process. A rule-based critic will use rules or rule systems to compute the overall fitness from aspects or components in candidates. Problems with this approach to the critic is that it performs well inside the evoluted materials, but fails to work as well when exposed to different start input. In other words, they fail to generalize. Todd and Werner summarize:

> But there remains a deeper problem with rule-based critic approaches in general, as people found earlier with rule-based composers. Artificial critics who go strictly by their given rules, as opposed to more forgiving (or sloppier) human critics, are generally very brittle. Rule-bound critics may rave about the technically correct but rather trite melody, while panning the inspired but slightly-off passage created by just flipping two notes. In fact, for good composers it is critical to know when to break rules. As a consequence, for critics it is imperative to know when to *let* the composers break the rules. Rule-based systems, by definition, lack exactly this higher-level knowledge. Critics based on learning methods such as neural network models, on the other hand, can generalize their judgments sufficiently to leave (artificial) composers some much-needed rule-breaking 'wiggle room' – though this too can end in tragedy, as we will see in the next section.[529]

The next step up in automation of the critic is a learning critic based on neural networks that are trained with preference material. But again, problems of mal-adaptations arouse when feeble generalizing in the melody resulted in erronous harmonies. Some properties are better handled by

hard decision-making rules than soft or approximate generalizations, after all. Spector and Alpern tried therefore hybrid systems for the critic, evaluating fitness as a combination of rules and networks. But worse still, such learning systems seem to converge on local best or first found solutions and stop from then on to look for further innovation or novelties.

> This can be especially true when the population quickly finds a loophole in the fitness function – the 'cheating' solutions will have such a fitness advantage over other members of the population that they will rapidly take over, killing off any other alternative approaches. ... But how can we build such changeable criteria into artifical critics, to ensure continued evolution and generation of novel creators? We must un-fix them. We can allow the critics to evolve as well, tying both critics and creators together into a co-adapting skein.[530]

The idea behind this Frankensteinian scenario is to un-fix the critic to evolve *simultanously* with the evolution of creators (composers). There are now two populations, creators *and* critics that operate under Darwinian conditions (variance and selection) without human interference whatsoever. There are clear reminiscences with virtual worlds of ALife from **7.3** and the learning environments of learning agents from above. Humans only set up an experimental playground for musical evolution conducted by creator programs and critic programs that influence each other somehow like two tennis players extend their technical repertoire and competence by selecting positively good or winning hands on the expense of bad or loosing hands. Todd and Werner use birds and sexual selection as intuitive model for what they do, when they operate with a population of "male singers" and "female critics":

> Each female listens to the songs of a certain number of males who are randomly selected to be in her 'courting choir'. All females hear the same number of males, and the size of the courting choir – that is, a female's sample size – is specified for each evolutionary run. After listening to all the males in her potential-mate choir, the female selects the one that she most preferred (i.e. the one with the highest score) as her mate. This female choice process ensures that all females will have exactly one mate, but males can have a range of mates from 0 (if his song is unpopular with anyone [of the females]) to something close to the courting-choir size (if he has a platinum hit that is selected by all the females who listen to him). Each female has one child per generation created via crossover and mutation with her chosen male mate (so this child will have a mix of the musical traits and preferences genetically encoded in its mother and father). This temporarily puts the population at about 50% above a specified 'carrying capacity' (target population size). We then kill off approximately a third of the individuals, bringing the population back to a predetermined carrying capacity. This whole process is repeated for the same desired number of generations. [531]

Three different formal methods based on table lookup evaluate male songs. They are based on 'local transition preference', 'global transition preference' and 'surprise preference'. They ensure that successful songs will first build up expectations and then both satisfy *and* violate them.

> In this way we tested our expectation that coevolving preferences would allow more change (or diversity) in songs over time because the targets for males would themselves be moving.

This assures three properties of co-evolving learning systems for composition: song change over time (diachronic diversity), song manifold within populations[532] (synchronic diversity) and avoidance of creators or singers to find easy ways (to trick) to habituate critics to easy listening samples in return for high fitness scores. All in all, these properties stands for values roughly described as *development*, *diversiveness* and *complexity*, all values that belong to biological phenomena as well.

The structure/novelty trade-off or dilemma between control and creativity

Todd and Werner conclude that their system produces novelty and diversity, admitting that the arising songs lack structure. Therefore they propose for future versions to build intra-generational learning of expectations that would speed up the evolution process further. Then, learning could be operating on notes and phrases of songs (see the signature concept of **ch4**) instead of entire songs

only:

> In this case, females would seek novelty and expectation-violation *within* each song they hear. To sing preferred songs, males would have to balance the amount of repetition and newness in their song. We expect that this type of 'real-time' preference learning will lead to increased complexity of the internal structure of the songs themselves, not just of the population of songs.

Another way to force higher structure (audible) into the co-evolving system is to “hard-wire” certain evolutional premises in the form of fitness rules into the critic, creating *hybrid fitness* criteria:

> These rules can be used to eliminate (male) songs from the population that are in flagrant violation of the human user's encoded aesthetics. The power of sexual selection [and therefore evolution] through female choice will continue to produce a variety of attractive males singing preferred songs; but now the human-provided fitness rules will act akin to natural selection, keeping the population healthy and well-adapted as well as attractive [to us humans :)].

Todd and Werner seek new ways to build autonomous composition systems, but they are continually challenged by the fundamental trade-off or dilemma between excessive and rigid structure arising from symbolic AI and excessive and floating generalizations and novelty as consequence of sub-symbolic AI or neural networks. Todd and Werner's system of *co-evolution*, *in the spirit of Frankenstein*, is fundamentally based on the integration of rule-following, learning and evoluting to mitigate these contradictions. They conclude therefore rather optimistic:

> By coevolving female song critics with learned musical preferences alongside the male-produced songs they choose between (which can also be honed by learning within male's lifetime as he encounters multiple choosy females), sexual selection will produce ongoing novelty. By pruning that novelty with human-produced rules, natural selection will keep the system within appropriate musical bounds. Structure and novelty, like the effects of natural and sexual selection can be balanced.[533]

7.6 Discussion : Between “chaotic order” and “hopeful monsters”

We have followed several different paradigms and strategies in the more recent projects of MC. Is there a convergence or general trend to spot? It seems rather difficult to conclude anything about a field with such a relatively short history. I tried to show that the paradigmatic diversity with models from many natural and human sciences may even question ideas of convergence or at least put it farther away into the future. All in all, the question is whether current MC goes more towards engineering/order/self-developing or organismic-like “hopeful monsters”? Psychological and mathematical models are knowledge-oriented and seem engineered, relative to physical and biological models that see open ended systems as the best road to creative complexity.
Self-learning systems and “genetically” evolving models can be said to leave control of their genesis or knowledge base to systemic grown adaptations between an artificial environment and the MC agent that responds to the pressures of selection humans facilitate but not specify in detail.

The mathematical examples of recent MC on the other hand, seem more “phythagorean” in both style and philosophy. The idea of building music fundamentally on the grounds of 'pink noise' or 1/f-noise is really quite abstract and therefore not specifically directed towards humans. The idea of “order in chaos” refers in classic Greek philosophy to a kind of balance between noise and sound, or dissonance and consonance or known and unknown.
Information theory has some interesting contributions to make about order and chaos, a theme we will follow up in **9.5**.
MC systems based on such general notions assume that humans fit well into more general contexts

of nature, shared with other organisms, some even with complex non-life phenomena [7.1]. These sonic models seem to be built on the belief that meaning and interest can be established without recourse to more anthropic constraints collected by musicology and music psychology.

Knowledge found as a consequence of scientific progress and as a result of studying music in more human contexts [7.2, 2] are focus for the psychological models presented in ch4 and ch5. We see now that also these models operate heavily on premises similar to information theory and more generally are expectation based approaches. Most models from 7.2 are related to the tonal systems approach from musicology. One might say that this dependency on cultural phenomena is unwarranted for a general model, but as we saw in Temperly e.g, modern approaches in music psychology are trying to encompass different cultural practices and concessing to other features (such as rhythm from non-western cultures) a more central place in style descriptions. In any case, both mathematical and psychological models are relatively static models, compared to the new group of self-developing systems. Mathematical models use "qualified randomness" (i.e. theory-conscious or system-tuned) and user-interaction to create variations. Psychological models typically proposed material coming from outside users, and transforming it (variability) . In other cases they "know" the style or truth, somehow like Fux-based counterpoint or Ebcioglu's harmonizing system that is an expert in its *fixed* domain, very unlike the *open* MCs of physical and biological models.[534]

Physical models involve the setting up of new virtual musical worlds where all rules are stabilized from start until the progressively higher "forms of musical life" emerge. Humans are taking the role of a first mover ('god') and leaving the rest to the synthetic rules enfolding the musical forces and inner dynamics of the system. The more autonomy the system has the more important are the selections of start conditions, especially the choice of environment. And learning and evolution are as a consequence not anymore so unlike than is still often taken for granted.[535]

An overarching trend that seems to be give influence in all these mentioned types of approaches is the call for *hybrid systems*, i.e. integrating several strategies from within a paradigm or model, and even across models. We saw this "open approach" already in Cypher's pragmatic and heterogenic system, and found it as well in the latest systems of evolutionary composing systems, such as the Frankensteinian proposals and others like it.

Studies of hard problems in Algorithmics are steering in the same direction and are preparing for algorithmic solutions in *parallel* processing (using more than one algorithmic approach), as well as looking at DNA computing and quantum computing as possible new ways to disentangle complex and indeterminate situations (Dewey). A list of proposals of hybrid algorithmic solutions is supplied in "Algorithmics for hard problems" (Hromkovič, 2003[536]) and genetic algorithms are deemed ideal for parallel implementations of algorithms (not computing![537]):

> Communication actions between processors are necessary for the selection procedures (the selection of parents and the selection of a new population) because to estimate the probabilities of the random choice of particular individuals from the population requires global knowledge that is distributed among the processors... Thus, already a straightforward parallel implementation of the island model of genetic algorithms provides an essential speedup of sequential genetic algorithms...Divide-and-conquer and dynamic programming are very suitable for parallel implementations, because the created subproblemscan usually be solved independently. Additionally, some problems allow an efficient parallel computation of a solution to a problem instance from the solutions to some of its problem subinstances. Thus, the achieved speedup of a parallel implementation may be reasonably high.[538]

Heteronomic systems may thus be framed in collaborating or competing networks. In addition, music psychology and computational musicology, supplied with lessons from evolutionary musicology, contribute their own contentions in favor of building theories of complex musical

meaning, not by hierarchic rule systems, but by networks of rules, interrelated to each other.

Open source culture is another factor that will influence future MC efforts considerably. While many historical systems were built with inhouse tools and did not, as a rule, distribute their source code, some of todays system builders publicise their results from the bottom up. An example is the open design for computer-aided algorithmic Music composition of *AthenaCL* by Christopher Ariza.[539] It is modular, polyphonic, poly-paradigmatic, open-source and object oriented. It is designed in the easy accessible programming language, Python, and actively invites others to contribute further to its extensive library of functions. When systems will be adopted and adapted by growing numbers of groups in MC, certain standardization pressures may set in and will possibly lead to more streamlined developments of *common practices* and progress, still unseen in todays scenario.

In this short summary, more questions arise than are answered. But this is the normal state of a "revolutionary science"[540], struggling for progress and agreements, eagerly awaitening its status as normal science in a future time. We will return to such questions in **ch9** and not the least the perplexities that seem to follow them as their shadows in the concluding chapter. In the next chapter, **ch8**, we will look at a tool of integration of MAX **[ch3]** and Lisp **[ch6]**; MaxLisp. It is somehow the peak of this thesis, where new "technology" (maxlisp) is made of several familiar ones (Max and lisp), and where theory about machine composition and its underlying philosophy has to contribute in measuring its success.

Chapter 8
Integration of AI and Music
Max + Lisp = maxlisp

1. Maxlisp – the obvious and only integrational tool between AI and Music?
2. Between MAX and Lisp – A description
3. A simple example
4. MAX vs. Lisp: advantages and shortcomings
5. Micro-project of evolutionary programming paradigm with MAX/Lisp resources (sketch)
6. Conclusion? - Why this alliance seems natural after all

I have a few reports that when the [maxlisp] object is instantiated, the system goes into the "spinning disk of death" mode as CLISP attempts to allocate the memory it needs. ... Hopefully this won't happen to you.
(From potential "pitfalls"; webpage of maxlisp)

8.1 MaxLisp – the obvious and only integrational tool between AI and Music?

It is certainly *not the only* bridge to integration, as we will see. In this paper I highlight the MAX *standard* (not the program "Max/MSP" or others) and the Lisp *language* (not the CM libraries or others) as essential examples of general tools for MC. This choice is partly motivated out of diffusion, both MAX and Lisp communities are majorities as it comes to MC. As a consequence, the respective user communities contribute to the growing catalogue of libraries and externals (Max). But there are objective reasons as well.

Max is like a swiss knife specializing on interfacing control functionalities in various directions. Lately, for instance the [hi]-object in MAX lets users connect affordable input devices, based on the recent standard of HID (Human Interface Device). Gadgets like Joysticks and floor pads ("dancing pads") can now be interfaced easily with MAX to control any imaginable soundmodule in the range of MAX' output tentacles. The range of applications that can now be integrated with MAX is far reaching, extending to visual and almost any other controllable domains.

Lisp is a swiss knife in the domain of abstraction and algorithms. It is a Turing type programming language **[ch1]** that brings high expressivenes, not the least due to its homoiconic representation of code and data[541], i.e. functions may be returned by and passed to other functions (they can be processed just as data), and the fact that Lisp can manipulate other Lisp programs and even parts of itself by use of lambda and macro expressions[542] **[ch6]**. The distinction between expressions and statements in other programming langauage is in Lisp dissolved in favor of expressions only.

It seems therefore natural to assume that the combination of two swiss knives will generate a very powerful integration of potentials and union of interests in MC. Basically, the idea is that interfaces, dataflow and object parsing are handled optimally by MAX objects and messages, while algorithmic and abstract searches in data structures are more logically solved using Lisp lists and functions.

Brad Garton, a researcher in MC at the Columbia University (Computer Music Center) produced an encapsulation of a CommonLisp implementation from the free domain (GNU) within MAX, called *MaxLisp* (**[maxlisp]**[543]) in 2004/05. It is the focus of this chapter and it will be treated in more detail

in the following sections. Maxlisp is not the only Lisp that has been embedded into MAX. Scheme, CommonLisp's minimalistic variant, has been brought to Pd **[ch3]**, but it is rather immature and tends to generate bugs.

Other alternatives for integration of procedural and general-purpose programming with MAX are:

- *C*, the native language of MAX, and of most libraries and externals.
- *Java*
- *JavaScript*: MAX-external [js] executes the core JavaScript language with MAX functionality.
- *Phyton*: MAX-external [nyptho] interprets phython code of MacPython upwards.
- *Vade*: not a programming language, but a collection of object and patching methodologies for creating modules and dynamic and scriptable user interfaces in MAX. It makes it possible to ensure modular and reusable structures in MAX-patches.

In addition, there are script-oriented and domain-specific programming languages for music and sound synthesis, processing and composition (see also below). Here are some examples:

- *[chuck~]*: encapsulation of the functional-style Chuck music-programming language for advanced sound synthesis and signal-processing (expanding on MSP functionality of MAX).It has also an extensive and very effective timing and scheduling system.[544]
- *[rtcmix~]*: encapsulation of a scripting/scheduling language of "musicN" for sound synthesis and processing. Its functional and C-like syntax enables algorithmic programming.[545]

Maxlisp is thus certainly not *the only* contender but eventually *the* candidate for a tight and effective integration of MAX with AI projects of the various types we have looked into. There have been lots of attempts to do complex patches in MAX that build or model some knowledge or intelligence into MAX. An example is the patch *nn* that is the outcome of experiments in neural net programming with bits of audio as input. But many of these attempts have demoed the algorithmic insufficiencies as much as their intent. A more general and principled approach might therefore be to embed a Lisp language into MAX and hereby open it up to the enormous archives of AI and CM code in Lisp language.

8.2 Between MAX and Lisp – a description

Maxlisp is the encapsulation of a major Lisp implementation in the GNU tradition.
CLisp has the following properties:

- memory-modest (4MB).
- it implements most ANSI CommonLisp syntax and functions plus some non-generic extensions.
- it is free of cost, freely distributable.
- highly portable (Maxlisp is Macintosh/OSX only, CLisp itself is convergingly universal).
- CLisp distributions consist of both interpreter and compiler (compiling speeds up by factor 5).
- CLisp allows for "upscalable datastructures" (i.e. all data types may be of unlimited size including floating. point number precision and it can have arbitrary integer variable length).
- it includes more than 800 library functions.
- debuggable at source level (stepping through the forms of interpreted code as well).

The Lisp interpreter will work from within any MAX box that is declared as maxlisp-object and will operate in real-time on "flowing" Max/MSP-data. Therefore any maxlisp-object consists of code that either :

1. generates data autonomously (e.g. random-generated sequences, or rule-generated patterns)
2. processes output using input as material, i.e. it reads the input from *lines* (connections) coming from other objects ('boxes') in MAX for processing inside the maxlisp object. In lisp lingo, expressions that make up the

program are inserted (as text) into the maxlisp object for evaluation whenever their variables are bound to new values coming from the outside of the maxlisp-object, i.e. from any other MAX object or another maxlisp object.

3. Filtering input to streamline or order input, i.e. it reads the input and omits "noisy" data according to some function of order.

Referring to **ch3**, we might in analogy define a zero-patch as one that returns whatever is bound to x: **(defun zero-patch (x) x)**. A positive patch (2.) is generating output with more data than the original input: e.g. **(defun ornamentify (x) (list x (1- x) (1+ x) x))**. This will enrich the output by chromatic surrounding or enclosing ornaments, converting a simple ordered sequence to an ornamented sequence, eg. '(c d e f g) will enter the function *ornamentify* in MIDI-representation '(60 62 64 65 67) and return

'(60 59 61 60 62 61 63 62 64 63 65 64 65 64 66 65 67 66 68 57).

A negative patch (3.) is subtracting data, e.g. reducing a non-linearly melodic line to a linear direction: **(defun filter (l &optional (y (apply 'max l)))(reverse(list (1- y)(- y 3)(- y 4)(- y 6)(- y 8))))**

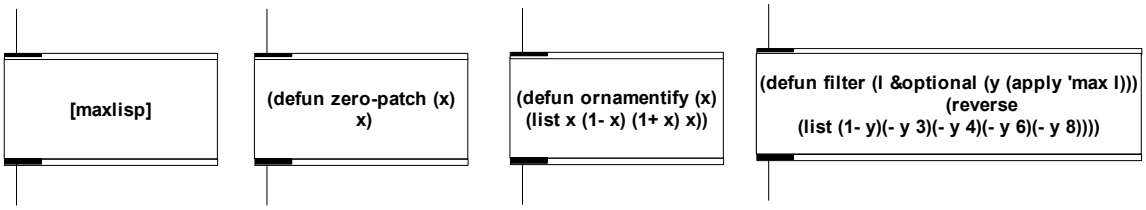


Figure 8.1-4: [maxlisp]-objects with no code, zeropatch, positive patch and negative patch

This last function filter is *no musical* function, it returns the original information (not data) transformed only syntactically, i.e. without "morphological" knowledge about scale or key. We will in the following example describe a slightly more general function (*tonalpoints-list*) that incorporates *some* "knowledge" about context (scales e.g.) and discover how early problems of contextuality enter in most kinds of musical processing.

In all the above examples the top lines (*inlets*) come from a keyboard object and the bottom lines (*outlets*) go to an midiout object **[ch3]** making complete input/output or playing/sounding *circuits*.

Max boxes or objects fit the structure of functional programming quite well. Any program in lisp is a collection of functions, including variable declarations (setq, setf etc.) that are functions as well. So a Lisp program will have an initial or top-level function with arguments that enter through the connection line and are bound to the internal variables (of *sonataform*), that in turn evaluates and returns the output through the outlets connection line to another MAX object of any kind **[ch3]**.

Lisp can be sent into maxlisp by loading from a file, sent as messages from other objects, sent as text messages remotely (remote message *send,* **[ch3]**) and from any *[textedit]*-object.

Now let us look at a simple example of a patch in Max with a slightly more interesting maxlisp object.

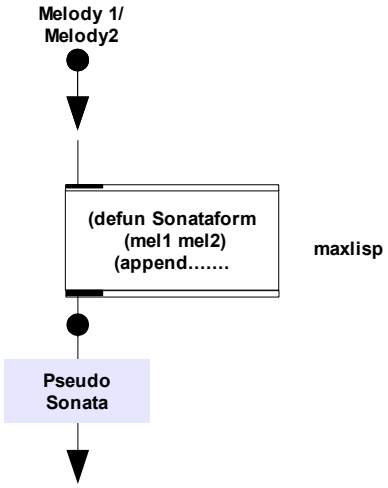


Figure 8.5: Sonataform [maxlisp]-object

8.3 A very simple example (Sonata form “Würfelspiele”)

This program is partly implemented in Lisp, and partly described and suggested only. It is an example of how incoming motives ('melody') played by a human user into Max/MSP can be evaluted in one or more maxlisp objects, to analyze them for possible tonal homes (*tonalpoints-list*). It will use rudimentary constraints, rules and statistics to assign a '*best key*” and out of this have to build “large scale” compositions in *pseudo 'Sonata form'* applying canonic key progressions. Selected variations (ornamental and structural) and contrast are applied (*variat-random)*. All the *artistic decisions* in this case are taken care of the *random*-function in Lisp, and it is therefore a rather lunatic and indifferent composer we model in this case. But since the macro-form of sonataform functions as constraining form, the inert character of random is disciplined somehow. This is, in its present form, a good example of 'anti-Machine Composition-system' since it has so extremely shallow knowledge, but may in the same time elicit memories about ELIZA.[546]

Functions:

	Mode definitions: very simplistic minor !	(setq d '(0 2 4 5 7 9 11)) ; (setq m '(0 2 3 5 7 9 11))

	Fetch-random (list)	`(defun fetch-random (l)`
		`(let ((fetchnumber(random (length l))))`
		`(loop(cond ((zerop fetchnumber)(return (car l))))`
		`(setq l (cdr l))(setq fetchnumber`
		`(1- fetchnumber)))))`

	Testmelodies	`(setq testmel1 '(60 62 64 65 62 67 60))` `(setq testmel2 '(60 63 58 69 54 61))`
	List of variations [sublist with implemented variations only]	`(setq variations '( retrograde inversion invertrograde augment diminish condense ornamentify surprisify ))` `[(setq variat '( retrograde inversion invertrograde ornamentify ))]`

	normalize (melody)
	
	Compresses melody range into one octave
		`(defun normalize (m)`
		`(let ((firstpitch (car m)))`
		`(mapcar #'(lambda(x)(- x firstpitch ))m))))`

	Positivize (melody)	`(defun positivize (mel)`
		`(cond ((null mel)nil)`
		`((< (car mel)0)(cons (+ 12 (car mel))(positivize (cdr mel))))`
		`(t (cons (car mel)(positivize (cdr mel))))))`

	Relmel (melody)
	
	;; RELativeMELody normaliserer melodi (zerofies start); based on relmel2!
		`(defun relmel (m)(relmel2 m (car m)))`
		`(defun relmel2 (m mirror)`
		`(cond ((null m)nil)`
		`(t (cons (- (car m)mirror)(relmel2 (cdr m) mirror)))))`

	Variat-random (m) Varies m in random chosen transformation, see also fetch-random and variations	`(defun variat-random (m)` `(apply (fetch-random variat)(list m)))`

	retrograde (melody)
	
	Reversed melody
		`(defun retrograde (m)`
		`(reverse m))OR`
		`(defun retrograde (m) (retrograde (inversion m)))`

	inversion (melody)
	
	Horizontal inversion/ inverted intervals around (car m)
		`(defun inversion (m)`
		`(let ((y (car m)))`
		`(mapcar #'(lambda(x)(- y (- x y)))m)))`

	invertrograde (melody) Retrograded of inverted	`(defun invertrograde (m) (retrograde (inversion m)))`
	augment (melody)	Multiplies number of all notes
	diminish (melody)	Drops repeated notes
	condense (melody)	Omits subset of less defining notes in melody

	ornamentify (melody)
	
	Adds notes around existing notes
		`(defun ornamentify (m) ;; ok +1, -1 0 osv`
		`(cond ((null m) nil)`
		`(t( append (list (1+(car m))`
		`(1- (car m) )(car m) ) (ornamentify (cdr m))))))`

	merge (mel1 mel2)	Merge two melodies (in some way!)
	surprisify (melody)	Add “daring” notes in between
	transpose_l (melody steps) Transpose litterally up or down	`(defun transpose-l (m s)` `(mapcar #'(lambda(x)(+ x s)) m))`
	transpose_m (melody steps)	Transpose musically (preserving key of melody)
	transpose_f (melody steps)	Transpose preserving function (I, IV, V, VI) (comprises transpose_l and transpose_m!: diff.!)

	Tonalpoints calls **tonalweight-points** to calculate the measure of fitness of a scale to the melody given. High numbers are good fits, low (negative numbers) are misfit scales.	`(defun tonalpoints (mel scale)`
		`(tonalweight-points (positivize(compress (normalize mel)))scale))`
		
		`(defun tonalweight-points (mel scale)`
		`(cond ((null mel)0)`
		`((member (car mel)scale)`
		`(1+ (tonalweight-points (cdr mel)scale)))`
		`(t (1- (tonalweight-points (cdr mel) scale)))))`

	Tonalpoints-list (melody scale counter)
	
	Generates list with tonalpoints for all scales relative to melody, the highest value is the best fitting scale!
		`(defun tonalpoints-list (mel scale counter)`
		`(cond (( null counter )nil)`
		`(t (cons (tonalpoints mel scale)(tonalpoints-list mel (mapcar #'(lambda(x)(1+ x))scale)(1- counter)))))))`

	Sonataform (mel1 mel2)
	
	Constructs a monophonic proto-sonata form piece using to melodies
		`(defun Sonataform (m1 m2)`
		`(append (introduction m1)(exposition m1 m2)`
		`(transition m2)`
		`(codetta m1) (development m1 m2)`
		`(recapitulation m1 m2) (coda m1)))`

	Introduction ()	`(defun introduction (mel1) (print mel1) (append mel1 (cadence (car mel1)))`

	cadence (key lasttone)	`(defun cadence (tone) (list tone (+ 5 tone) (+ 5 tone) (+ 7 tone) (+ 7 tone) tone tone))` `(defun cadence2 (tone) (list tone (+ 5 tone) (+ 5 tone) (+ 7 tone) (- tone 5 ) tone tone))`
	Codetta (mel1) combination of a/b?	`(defun codetta (mel1) (print mel))`
	exposition (mel1 mel2)	`(defun exposition (mel1 mel2)` `(append mel1 (variat-random mel1) (variat-random mel1) mel1 mel2 mel2 (cadence (car mel1)))`
	transition (mel1 mel2)	`(defun transition (mel2) (print mel))`

	development (mel1 mel2)	`(defun development (mel1 mel2) (print mel2)`
		`(append mel2 (variat-random mel2)(variat-random mel2) mel2 (variat-random mel2)`
		`(transpose mel1 7) (transpose (variat mel1) 7) (transpose (variat mel1) 7) ) ))`

	recapitulation (mel1 mel2)	`(defun recapitulation (mel1 mel2) (print mel2))` `(defun coda (mel1) (print mel1))`
	coda (mel1)	

This is a description of the structure of a sonata form to generate from **`Sonataform (m1 m2)`**:

Introduction	**Introduction**
A (s1)	**Exposition**
A (random variat)	
A (random variat)	
A (s1)	
B (s2)	
B (s2)	
Candence	
Codetta	
4 * s2	**Development**
3-4 * s1	**last in key(s1)**
2 * s2	**Recapitulation**
2 * s1	
Coda/cadence	**Coda**

Example call of main function with short comments:

```
(defun Sonataform (mel1 mel2)
(introduction)
;; optional - slower - focus on dominant - may be unrelated to ...
(exposition)
  ;; one or more themes in home key
  ;; modulation to next key 2.s (group) maj->Dom/min->maj(relative)e.g. c->E
  ;; 2.s more lyrical?
(transition)
;; perfect cadence in 2.s key
(codetta) ;; closing theme
(development)
;;continues key,retransition to tonic 1.s key
(recapitulation)
;; 2.s in home key
(coda)
;; perfect cadence in home key
 )
  ;; Sonataform picks (chooses good themes by mapping them to certain criteria) themes from human input and uses randomized repetitions of subjects with randomized selection of variation-types and contrast-types, adds cadences and shifts key according to a proto-sonataform plan!
```

These few lines of Lisp code demonstrate two things, first, hierarchical structure like the one we find in the sonata form, is easy to implement with recursive functions; second, such a program is very convenient interfaced with both melodies played into the top-level function (*sonataform*) and the returned "sonata" is played to MIDI output gear through MAX objects for its sonic actualization.

8.4 MAX vs. Lisp: advantages and shortcomings

Several joysticks **[8.1]** can be routed to Max/MSP after setting seperate index numbers for each joystick control or any other input device. This is done by a polling or initializing procedure. Andrew Benson in "Making connections"[547] describes examples:

> Now that you have learned how to assign individual controls from your joystick, you can connect these values to control virtually any parameter in MaxMSP or Jitter. Try creating filter sweeps using the *lores~*object or changing the delay and feedback of a *tapin~* - *tapout~* -network using your analog joystick. If you are using Jitter, you can assign the analog stick values to a *jit.xfade* or other compositing object to create a video mixing system complete with clip triggering and control over different video parameters. With a little scaling or remapping of the values, you can create some very interesting controls for your patch.[548]

In this article Benson goes on describing relatively simple alterations of the electronics of a very affordable USB-joystick to incorporate switches, knobs and resistive sensors like flex sensors that again are connected to moving body parts of controlling humans. Any variable resistance component, like flex sensors or photoresistors reacting to light variations, can substitute factory controls. MAX is therefore expanding the horizon for music machines, performance systems in particular, but even MC systems will profit from these opportunities to explore new directions to interface with the world.

CLisp is a fast performer overall, outperforming both Java and many other alternatives. Only within floating point arithmetics and extensive numeric calculations there are other lisp alternatives (like CMUCL) that outperform CLisp. As long as one uses the economic MIDI representation, no such numeric overload should arise. Doing raw-sound-sample data manipulation, maxlisp will have to be tested for its limits with respect to real-time applications. In this area, many of the Max/MSP objects, programmed in C and optimized for real-time, will probably excel compared to the "symbolic Lisp crunching".

> ...Lisp is the most powerful symbolic programming language yet it has trouble coping with the pressure of real-time control – to the best of our knowleddge, just one effort aims to connect the universal power of Lisp with the optimized number crunching efficiency of Max/MSP [Garton]. More work is needed to design robust multi-purpose, concurrent computational environments using symbolic programming languages.[549]

Progammers of MAX can work in a multi-methodological way. They may use different programming paradigms already inside MAX, but can choose from the even more idiosyncratic resources of thousands of community developed objects.

Lisp, especially in its CommonLisp super-dialect, is multi-farious in its own terms. Many programming styles are possible. Straightforward imperative programming, object-oriented or recursive functional programming, they are all alternatives on the menu of the Lisp programmer. So, like mentioned above, both platforms are swiss knives that seem to complement each other rather well.

Problems with MAX:
But like in the real world, there are no upsides without associated downsides. Recursion is not possible in MAX, because objects are automatically loaded. If an object references itself by name, it will load infinitely and generate an error in MAX' runtime eviroment. MAX doesn't permit lists without members either (atom type **A_nothing** is reported to not work properly).

Max has no robust and effective string-manipulation functionality. There are some libraries that add limited support, but the core of MAX is oriented towards number crunching and scheduling

Problems with maxlisp:
Maxlisp is the new kid in the block, so initial growing aches are to be expected. Still, maxlisp users have to cope with some annoying instabilities, especially due to the lack of a debugger mode. There is no operative debugger inside a maxlisp object. Therefore, whenever a break condition occurs from within CLisp/maxlisp it will cause Max/MSP to crash as well. Lisp code for maxlisp should therefore be debugged and tested thoroughly in a CLisp environment, before running it as a [maxlisp]-object.
If a Lisp program execution takes excessive time, relative to the timing and schedule setting in its Max/MSP surrounding, unexpected results may occur. Demanding timing conditions caused by multiple embedded functions (large programs with deep structure for instance), willl often lead to memory problems related to deferred garbage collecting, and will in turn be able to cause break conditions, ultimately crashing both CLisp and Max/MSP.

Semicolons in returning lisp expressions may cause trouble to the integrity of Max/MSP, since semicolons are vital syntax for MAX-messages as delimiting character. Therefore should semicolons preferably not be used in lisp code that is later put into [maxlisp]-objects.

Currently, [maxlisp] is only ported to Mac OSX10.3+ and requires v.4.3 or 4.5 of Max/MSP. Hopefully, the cross-platform ability of CLisp will lead to other compatible versions for OS and MAX (Pd, jMax).

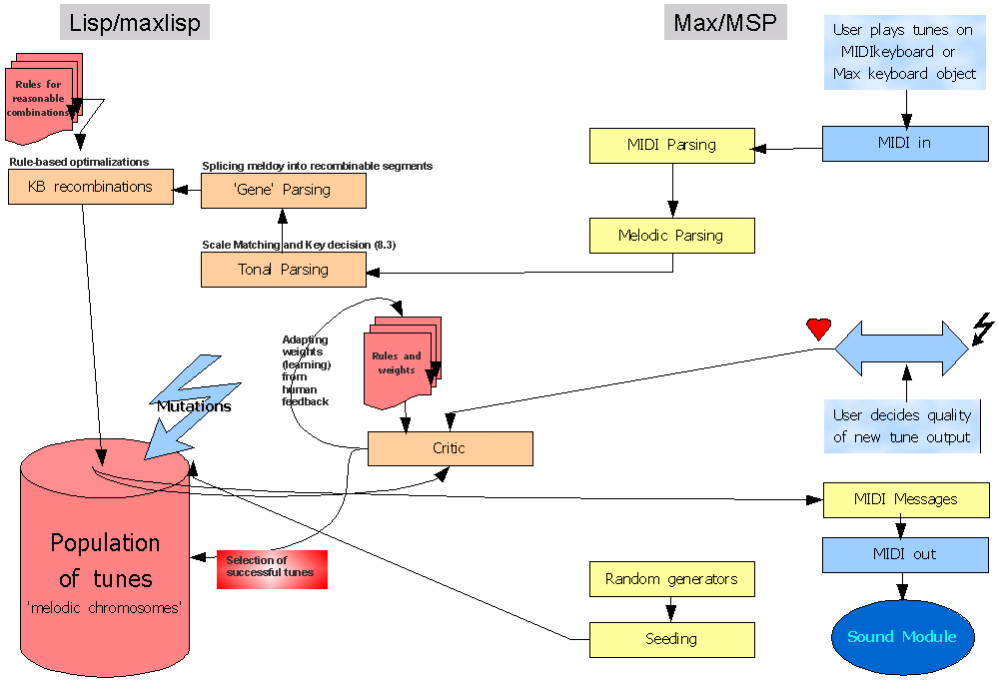
Tentative recommendations for use of [maxlisp]:

It is certainly premature and therefore presumptious to summarize good style for integration of MAX and Lisp, since [maxlisp] is of so recent date, and hasn't been tested extensively until now. The admonition in the entrance citation of this chapter is pretty much summing it up. Still, drawing on the strengths and weaknesses of both platforms in general terms, we might nevertheless come up with the following tentative and hence very much non-prescriptive “rules”:

1. Parsing of number objects or structures containing numbers are as a rule faster executed by MAX objects.
2. Straightforward processing of numbers is the dominion of MAX .
3. Input and output facilities of MAX are the logical choice (MAX has the wheels already!)
4. Before reinventing smart objects in Lisp, check libraries of MAX .
5. Functions that need user intervention in the MAX patcher may be better held inside MAX.
6. Functions that profit from debugging or visual feedback of what is going on, are only possible within MAX .
7. The more GUI that programs depend on, the more MAX objects will be required.
1. Recursion and symbolic computation is the dominion of Lisp.
2. String-manipulation is faster and more accomodated in Lisp syntax.
3. Nested and deeply embedded functions in Lisp should be avoided. Bigger lisp programs should be modularized and distributed over several [maxlisp]-objects to predict schedule exigencies and to avoid the timing problems that iterative functions may cause to the surrounding of MAX (and evt. depending maxlisp objects).
4. MAX objects that depend on maxlisp output may be explicitly prepared for retardations (adding artificial pauses e.g.).
5. Others

8.5 Micro-project of evolutionary programming paradigm with MAX/Lisp resources (sketch)

As we saw in the preceding chapter, automated programming in “biological style” is very much 'in vogue' today. We look at a component plan that integrates MAX and Lisp facilities to build such a system, again rudimentary and only pre-algorithmically:



This example of a possible system of evolutionary programming, contains several paradigms and interfaces. Its hybrid nature starts with using MAX as input and output platform. A person plays tunes (melodic fragments) on a MIDI keyboard and the results from the “biological processes” are sent back to MAX for sound generation. The incoming human tunes are parsed and *spliced* by maxlisp into *recombinable segments*, i.e. segments that are interchangeably at start and end parts. This segmentation is further refined by contextual knowledge from '*Tonal parsing'* and symbolic rule-based preference rules *filtering* out the less fortunate segments. The resulting tunes are now *collected* and *mixed* in the “melodic chomosomal population”, that is continually exposed to random generated, but seeded perturbations or simply stated *mutations* (non-directed influences). The mutations are contributing to occasional surprising effects, without tracking or understanding their sources. The critic of this system receives random recombinations of segments into chromosomes (longer tunes) as “proposals” from the population of tunes. The critic then selects “successful” tunes on the basis of its rules. The weights of the critic are continuously changed responding to the occasional quality *evaluation* by humans. Whenever an output tune is pleasing or non-pleasing, the human may *respond* to it by communicating this statement of taste to the *critic*, that again changes the relative importance of its rules, sometimes even by applying higher level rules and thereby creating new rules out of established ones. From the user's perspective this system is able to adapt to the user's taste reacting *actively* to playing tunes or *passively* to contributed feedback. The system itself will even be able to evolve without the 'helping hand' of a human, due to the possibilities of emergence of fresh and successful tunes from seeded small random generations stemming from within the population.[2.2]

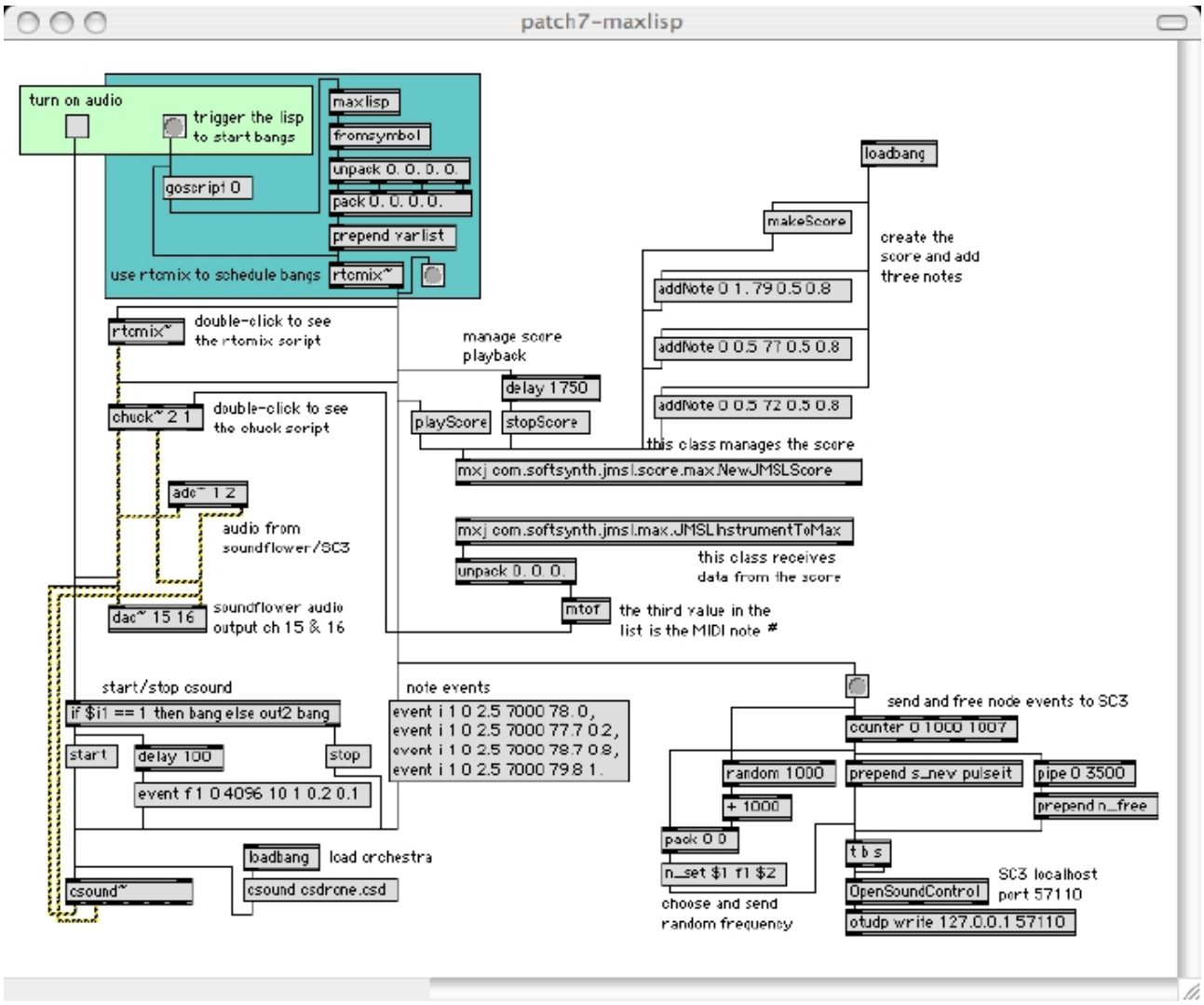
8.6 Conclusion? - why this alliance seems so natural

We saw the straightforward use of Lisp processing power in example [8.3], applied in communion with practical interfaces and control panel like behavior of MAX patches. [8.5] is a sketch of a far more comprehensive program. And still, it is a macroscopic view only of a functional full blown MC system of the evolutionary programming breed. This micro project handles, just like the one in [8.3], only pitch information, without enlightenment ("interference") from other musical dimensions, such as duration, phrasing, dynamics or timbre. We saw in both ch4 and ch5 how essential this multi-dimensionality is for the project of synthetically creating musical meaning. The structural analysis in such an approach ("melotonic", [ch3]) will therefore be artificially shallow and simplistic. But we may have realized that Lisp and MAX can be perfect partners, when they complement each other, and do what they are best in (using their specialities). So we might get the best of two worlds, after all?

Successful integration doesn't stop at this point, though. We recall that a considerable number of authors of recent MC systems concluded that an encouraging path to follow lies in the use of hybrid systems [ch7]. The person behind [maxlisp], Brad Garton is known for exactly this eclectic or pragmatic approach. His chase for opportunities to combine functionality from different sectors of computational music applications made him carry through an experiment. In 2007, Garton exposes the results in the article "Multi-language Max/MSP[550] where he goes for a "grand slam" of MC tools and paradigm integration. It is a test case[551] for parallel approaches within Max/MSP , that he describes as "computer music language extravaganza".

Garton starts with a script-based *Rtcmix*c **[I]** code [8.1] that generates random shifted clusters of sawtooth waveforms. Next, he stores real-time audio scripts in a *ChucK* [8.1] object, *[chuck]* **[II],** for filter processing of the sawtooth input. Then he uses *java-based JSML* code **[III]** that handles western music notation in a hierarchical scheduling environment. But instead of controlling the scheduling in *[mxj]* itself (what would be the logical choice, being a java object that accepts JSML), he chooses to use MAX' native *[metro]* **[IV]** object to control note and sound events coming from the *rtcmix~* and *chuck~* objects. Next in line, he opens a *[csound]* **[V]** object, the synthesis program mentioned in connection with CommonMusic [ch6], in order to add another instrument to the circuit. Garton brings pulse-width-modulation sounds to the patch, that he defines in the *SuperCollider* **[VI]** language. Its code resides outside of MAX (in an external application, i.e. not embedded in any MAX object), but is triggered and controlled via *OpenSoundControl (OSC)* **[V]**, an inter-application interface for music applications controllable from a regular MAX-object. After having done some peculiar audio patching, he ends the stack of heterogenous code in a crowning finale with *[maxlisp]* **[VI]** object that is "loaded" with fractal functions to control trigger times. Even so, Garton mentions a couple of languages that he left out, notably java and javasScript, and sums up the lesson like that:

> ... I believe that different languages do in fact structure our creative thinking differently ... the ability of Max/MSP to "glue" them all together is a real boon.[552]



Arne Eigenfeldt in “Managing Complex Patterns”[553] holds a different view. He believes that powerful programs are preferedly (or at least productively), made in exclusive-Max manner. He recommends to follow the wisdoms of structured programming, e.g. modularizing programs into subpatches (encapsulation). Other tips involve the use of debugging and text objects for inspection of messages following through the patch. Adapting to the simple rules of standard programming makes MAX a *homogenous* and *predictable* environment, according Eigenfeldt.

Garton looks into the future of the “Cooperating Computer Music Languages” (2007)[554] and delivers his vision of an “ultimate music technology”, where “ideas flow 'transparently' from conception to realization”. Since tools and languages will never be neutral in respect to their practical results (**[1.6]** for general programming languages in this respect)[555], Garton thinks combinations of approaches will expand the artistic expressiveness first and foremost. He hopes that developers will one day produce components and languages with “simple and transparent interconnectivities” to ensure that future users of MC tools will hardly notice these issues (somehow like with webpages displaying contents of heterogenous generative sources) when they choose tools from a menu of disparate tools and languages out of primarily musically and to a lesser degree technically motivated considerations **[1.8]**.[556]

> I also have to admit a bias towards imbedding in that it allows a very high degree of 'cooperation' between different environments without much need for data translation or reformatting... (Brad Garton, 2007).[557]

Chapter 9
Concluding remarks
Between Phytagorean dreams and LaMettriean temptations.
Machine intelligence and Creativity, Machine Music and Aesthetics.

1. Categories and Varieties of Machine Composition systems
2. Challenges and possible objections about Machine Composition
3. Towards a framework for Machine Music Aesthetics ?
4. Two theories that explain music (Meyer and Goodman)
5. Are there aesthetical traits that characterize (generalized) machine composition systems?
6. Aesthetics and Machine Music in a naturalistic and hence darwinistic perspective
7. Thought experiments: How could Machine Composition change the urban landscape?
8. Instead of epilogues: final remarks

"Man is so complicated a machine that it is impossible to get a clear idea of the machine beforehand, and hence impossible to define it. For this reason, all the investigations have been vain, which the greatest philosophers have made à priori, that is to to say, in so far as they use, as it were, the wings of the spirit. Thus it is only à posteriori or by trying to disentangle the soul from the organs of the body, so to speak, that one can reach the highest probability concerning man's own nature, even though one can not discover with certainty what his nature is, it is so complicated a machine that..."
Julien Offray de La Mettrie, Man a Machine, 1748

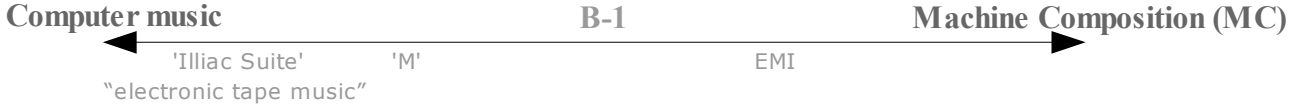
This *concluding* chapter is neither concluding nor exhaustive. It will at most express a series of remarks coming out of the succeeding chapters. It is open-ended in many senses. Therefore, the so-called "framework for Machine Composition Aesthetics"**[9.3]** is really not more than the inverse of a swiss cheese, its pretentious title notwithstanding. The phantasy in **[9.7]** is high-flying (if fortunate, less naïve as Ikaros'), but hopefully satisfying the curiosities of open-minded readers, perhaps telling more than a thousand honest words in an effort to stay respectable. Finally, **[9.8]** is a series of remarks *on* the remarks *about* the entire subject. In order of space, we say that this last chapter consists of a *positive* **[9.1]** , a *negative* **[9.2]** , a *constructive* **[9.3]** , a *theoretical* **[9.4]** , a *hypothetical* **[9.5]** , an *analogical* **[9.6]** , an *imaginative* **[9.7]** , and finally a *conversational* **[9.8]** collection of remarks about Machine Composition. These remarks are memes too, and as the cliché reverberates: if these memes invite other memes, *they* consider themselves a success.

9.1 Categories and Varieties of Machine Composition systems

Several attempts have been made to survey and categorize the many paradigms and systems in MC. I have proposed a preliminary system of two dimensions in **ch4**, based on Robert Rowe **[ch4].** Christopher Ariza[558] proposes seven descriptors of dimensions, such as micro/macro-structures, real-time/non-real-time, singular/plural idiomaticity, closed/open extensibility, generated/transformed event production, sound source and user environment. But in his discussion about the task of categorizing such systems he concludes that "the number and diversity of CAAC[559] systems, and the diversity of interfaces, platforms and licences, have made categorization elusive". He mentions some of the tentatives (Roads, Coy, Pennycoook, Pope a.o.) that are based on different, but overlapping and incompatible schemes of references and categories. We will therefore here have to content ourselves with references to the undertakings of others (especially *algorithmic.net)* and be happily spared from taking clear positions in these matters.
Instead of proposing well-defined categories or even metric scales, we illustrate the varieties of MCs using *prototypical* characterizations within a selection of *dimensions*. The resulting list of dimensions, as we will see, turns out to be neither exhaustive nor does it formulate metric spaces.

As this wouldn't be troubling enough, the name we will give to the entire domain of our subject, Machine Composition, ends up being a non-trivial act as well[560] (different names will emphasize unlike properties and perspectives). We therefore deal with rather intricate problems concerning categorizing actual systems of composing machines. The first here proposed three dimensions *(B1-B3)* are about a *system's behavior*, and the fourth is about its *interface (I-1)*; the last four dimensions clarify the *inner logic or functionality* of a system. We call the first four "*behavioristic* dimensions", and the last four "*cognitive dimensions*", reminding ourselves of the fact that these labels mostly serve as mnemonic devices.

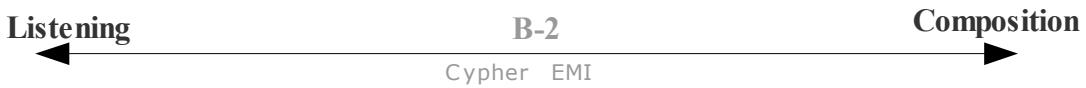
Computer music vs. Machine composition (B-1)
I have intended 'Computer music' to primarily refer to the compositions *generated* by computers in the early period of machine music experiments. Many of their creators were pioneers and often felt contented by affirming the generative powers of their computer algorithms, resulting in highly complex musical structures. Even though many of these pioneers tuned their systems (post-processing) to become more understandable to human listeners, communicational gaps with respect to non-specialized listeners persistently remained. Small circles of highly motivated or participating specialists were often the driving forces in these abstracting and formalizing subcultures that consciously put themselves in contrast to mainstream musical life.
On the other side, I defined 'Machine composition' in a way that includes the more recent systems (starting from around 1985). It does, in my opinion, include most[561] systems mentioned in this thesis, and excludes, by intention, many of the historical 'computer music' applications.[562] In practical terms, if sound-structures are much less meaningful to human listeners[563] than to their generating machines and human creators, I position them conceptually under the term 'computer music'. This puts us in practical disagreement with Roads, Cope and some others that prefer using 'computer music' in a very general and inclusive way. Material reasons for the positive discrimination towards MCs will be expounded in **9.3** as part of a normative proposal of criteria for successful composing with machines.



Concrete examples of computer music are the 'Illiac Suite' (too little directionality), many works of Xenakis (too high complexity) and many of the "tapes" coming out of the "Studios of electronic music" in the late 1950s and 1960s (too little relevance[564]). Along this axis, many of the mathematically inspired and knowledge-based systems will naturally populate the middle area in this dimension. 'M' **[4.4]** and 'EMI' (Cope, **[5]**) are prototypes that fit comfortly on the peripheral points of this conceptual axis.

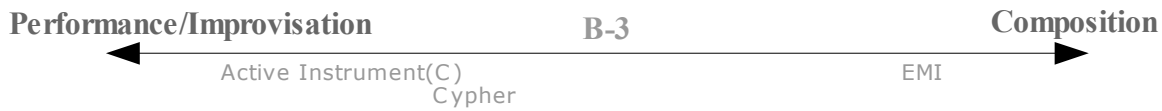
Listening vs. Composition (B-2)
A non-listening ("autistic") MCs may not be of much practical value, but serves well to illustrate a theoretical border case in this dimension. Such an "autistic" composing system, that neither needs nor wants input from humans, would probably not satisfy our basic expectations concerning relevance, meaning and communication (machine-man-relation). Therefore we conclude that any system of composition will have to listen *and* compose *somewhere* in its genesis. Listening, on its part, will include some degree of interpretation as well (see learning cycle, **2.2**). But, the listening part of MCs comes in various quantities and qualities, pretty much similar to the variances in "conventional" composing by "Sapiens". Some composers are conventional and follow a trend or culture. They will probably listen more (perhaps analyze less and relate more superficially) than the

prototype of an individualistic artist in search for his "own style"[565] and diverging so much from the "listened to" sources that his style in the aftermath remains elusive or opaque to his public. Then, the modes or ways of listening vary significantly as well. A MCs that accepts a graphical sketch as input may from this information alone be able to find qualitative dimensions (e.g. using Gärdenfors' conceptual spaces, **2.3/2.5**) and matching them to musical structures (in composing). Other systems may ask for "phoneme-like" melodic snatches of a song, before they start to process these patterns in composing.



We discover that both Cypher and EMI are tenacious listeners, but in their own ways and styles. While Cypher (tending towards improvisation, see below) reacts fluidly, faster and rather impulsive, EMI demonstrate perseverance under its collecting of signatures and stylistic features induced from idiomatic work collections in non-real time. In other terms, Cypher is more *responsive* and EMI more *reflective* in their modes of composing new structures. Both are learning, and since learning depends on input, both will have to *listen* to something. Less intricate MCs will ask for clearly defined input and thereby assure predictable generative processes (a type we will call "assisting MCs" in dimension C-4 below).
What about 'classification systems'? A system that specializes in listening and returns or outputs only the name of a category will occupy the ultimate point on the listening pole[566]. And just as we excluded "autistic" composing (non-listening composing), we may as well exclude the extremely "poor" "categorical composing" (non-composing listening) from our context of composition.

Performance/improvisation vs. Composition (B-3)
The miminum level of productive output is "performance" in terms of notation only. A system that returns no sound, but only a notational representation of it **[2.4]** will be a poor performer and hopefully compensating as a prosperous composer. For example, EMI **[4]** and Cmusic **[6]** output CMN[567]-notational representations of musical works. Other systems generate output in machine readable form, e.g. as MIDI files or other more "proprietary" standards. Yet, many MCs are more generous and supply sonic representations as well, i.e. demonstrating their creations in directly perceptable form. This requires performance modules that interprete or "enliven" the compositional structures. They are technically speaking *not* part of MC, since they pertain with micro-structure of sounds in a performance perspective.[568] Improvising MCs, either with other MCs or humans, will have to possess advanced functional properties of listening at various levels. Cypher is the prototype of improvisational MCs.



Actually some people find it more appropriate to exclude improvisational Cypher from our definition of MC altogether, thereby tightening the definition and particularizing the various stages in the learning cycle. This would shrink the range of this dimension by half. In fact, Rowe himself, prefers to call his system 'Interactive Music system", somehow avoiding 'composition' to become his main focus. On the other hand, Chapel's 'intelligent instrument', discussed in **ch7**, is a relevant example of a highly performance-oriented system of MC. With respect to its performing and improvising nature, Chapel's "active instrument" could be characterized as co-composer/co-player

relative to its human player/composer. In any case, I have chosen a liberal definition in all these cases and include them all in the "conceptual space"[569] of Machine Composition.

Closed vs. open architecture (I-1)
Closed systems were the rule rather than not in the earlier days of Machine music. This was in part a consequence of little flexible machine platforms and limited computing power, available at the time. This dimension relates to questions about:
a) what data/program or code/formats can enter machines during run-time mode,
b) how easy is it to import new modules of other programming languages or formats[570], and
c) what sound libraries/modules may be used and in what way (internal, exported, imported).

Generally, modern MC is taking advantages from increasing transparencies and their integrational forces. As mentioned in **ch7**, the reigning trends towards open-source platforms will further push these systems towards extensible functionalities and open architectures.

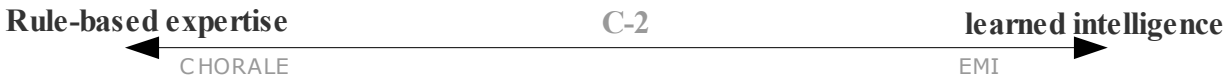
We know from **ch3** and **ch8** that both MAX and Lisp are rather open architectures. MAX and Cypher allow for real-time control (run-time). Nyquist **[ch6]** uses internal sound sources, athenaCL **[ch7]** calls Csound **[ch6]** as exported sound sources, and finally CommonMusic **[ch6]** uses CLM **[ch6]** as imported sound source.[571]

Sequenced/transformative techniques vs. generative/grammatical systems (C-1)
The following characterizations will to some degree overlap with the above dimensions. All the same, they make up a necessary contribution to the description especially of the *internal* workings of a system.
Sequenced, transformative and generative techniques, as we recall from earlier chapters (Rowe, **3.8, 4.2/p.75**), are the salient qualities of interactive systems. The former two are variating[572] bigger chunks of musical sequences, while generative techniques build up constructs from small elements (bottom-up). The grammatical apparatus, mostly rule-generating formalisms, ensures coherence and cohesion in the larger scale structures, typically entire pieces or works. Not surprisingly grammatical/generative systems supply *deep* structures, where the sequential/transforming systems stay on the superficie. Full scale MCs are expected to use more *generative* methods while improvisational systems mostly specialize in *transformative* methods. The crown example of a grammatical system is Lerdahl and Jackendoff's GTTM.[573] Its rules control both the horizontal and vertical order on different time-scales and in a specific idiomatic style (18/19.c music tonal idiom). This is done through an hierarchic system of wellformedness-rules and preference rules, that again rest on the so-called *Reduction Hypothesis*.[574] The authors describe its effects as "the pitch-events of a piece are heard in a hierarchy of relative importance; structurally less important events are heard as ornamentations of events of greater importance" (GTTM, 11.5 about Time-Span and Prolongational Reductions).

Cypher, as we may remember from **ch4**, uses a network of modules that "look at each other's states"

in order to ensure consistency in the overall systems performance. A consistency that must be weighted against what comes from its human "co-composer"[575], as well as its own prior and historical output.
As we may expect at this point, the richer and more elaborate these intra-relations of a musical piece are, the more will it gravitate towards the generative pole of this dimension.

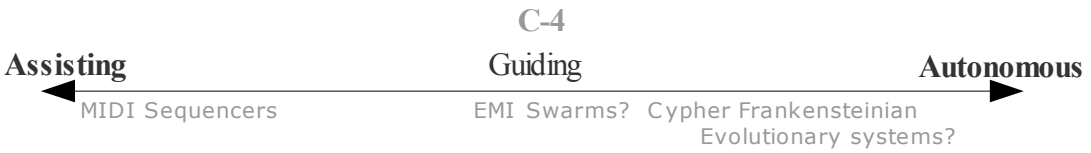
Rule-based expertise vs. learned intelligence (C-2)
This dimension is distinguishing MCs on the basis of their dynamics in relation to their environment. Rule-based expert systems are static (i.e. non-learning and changing); learning systems are dynamic. Learning systems adapt and adjust their own incrementing knowledge under the influence of the feedback coming from its environment (described in the AIMA-learning agent model in **2.2**).
Such adaptive properties are adequate descriptions of connectionist systems as well, where the weights pertaining to the links (between nodes) are continually adjusted (e.g. perceptrons[576]). But adaptation also denotes the internal 'critic' in symbolic systems, supervising the adaptations or changes of the inner structure (e.g. changing rules) to facilitate effective changes (i.e. improvements relative to assumed future exposures). This is very much in line with the musical learning cycle. The importance of musical intelligence (emerging in repeated "shapening" cycling around learning cycles) emphasizes the essential significance of this dimension for MCs. In effect, rule-based expert systems (manifesting expertise and not musicianship, **2.2**) exemplify what I called 'sector proficiency learning' in **2.2**, while full-blown learning systems are a logical equivalent to *full*-cycle-learning in musically abundant (variated) environments.



M **[ch3]** and Ebicioglu's *Chorale* system **[ch4]** are typical models of expertise, while both Cypher and EMI, as well as most systems encountered in **ch7**, reside naturally at the learning pole of this dimension.

Non-deterministic vs. deterministic decision-making (C-3)
This distinction will be discussed in **9.5** instead, for reasons that will be made more clear there.

Assisting vs. Autonomous Composing Machines Systems (C-4)
I believe this dimension to be the most central and relevant in distinguishing high level systems, manifesting intelligence *and* creativity from low level systems excelling as operators or calculators. I used a rather similar and preliminary scheme in **4.2** that placed systems from *tools to agents*, paralleling this updated version of C-4. The final discussion of creativity will be saved for **9.3** ; in the present dimension of different levels of machine composition intelligence, we will concentrate on the assumed gradient quality, ie. the variables that define the transition through levels of MC intelligence starting from a "mere" assisting role going thoroughly to the creative identity?



For further clarification, we "travel" along this axis, describing some exemplifying instances. Just as a first term student of counterpoint only applies (and knows about) the first or second 'species' of Fux' Parnassus,[577] a simple version of an *assisting* MCs solves counterpoint problems relative to standard 'cantus firmus' and a limited number of species. This is very much on the level of a calculator adding five numbers, or the returning of a Fibonacci sequel number for a specific sequence, e.g. 1,2,3,5,8,13,21.[578] We call this lowest level of MC intelligence *operator level.*
With higher forms of proficiency in well-defined areas we get to the *assistant level.* The next levels in the middle parts, will not only execute (repeat) musical operations, but vary material on the firm base of certain levels of rule complexity and context sensitivity. Actually, the full school of Fux' counterpoint applies five species/levels of rules.[579] On this level of proficiency, emphasis is given to strict rule application balanced with context considerations as well as increasing levels of idiomatizing enculturation.[580] What we have is *instrumentalized* intelligence, still not carrying "creative cargo". Chorale (Ebicioglu) uses around 400 rules, extending the conventional and explicit rules with new, data-mined, and finer discriminating rules. This makes Chorale an example of high level intelligence, but still limited to static knowledge. We may call this middle stratum of our dimension the *guiding level* of MCs. Guiding MCs will propose material and intelligently (knowing what they do) vary material that is selected or used as inspirational sources in human (or other machine's) compositional acts.
Moving on to the right-most section of this axis, we should expect increasing levels of explorative and experimentally derived solutions far from textbook expertise. Cypher scores high enough to appear on the right side off centered. EMI more in the middle, since it needs continual guiding from her mentor. But both systems are candidates for Turing tests with successful outcomes.[581]
The highest level of intelligent and creative MC is the *autonomous level*. Such systems will have learning and 'critic'-functionality or even do engage in the "grand drama of evolutionary forces" of Darwinism or memetics (below, **9.7**) in the contexts of Genetic algorithms or evolutionary programming **[ch7]**.
This level describes Otto Laske as "the artist using a computer as an *alter ego*".[582] We may further discriminate various levels of *ego* at the zenith in this hierarchical dimension. Composers adapt and learn their stylistic preferences from relevant stimuli in *given* environments. An *absolutely autonomous* MCs, on the other hand, will *not* be the autistic type described in B-2, but an agent that *chooses* h[is/er] *own inspirational and motivational input*, i.e. actively selects as well as applies models for further exploitation and development of h[is/er] own souvereign creative style. Having said so much, can a machine reach such a level? This will be the subject of **9.3** through **9.6**.
From the absolutely autonomously created styles and rule systems down to the lowest assistants functioning as "tonal calculators",[583] these are the outermost examples, more useful to fortify the explaining powers than serving practical purposes really. Even so, they reveal the astonishing "gradiental" properties involved in *upscaling* intelligence**s** (used to solve problems) and creativitie**s** (used to diverge the usual solutions). This is concretized in the similar staircase built with increasing levels of structural complexity, control, variation and contrast[584].

A less abstract and perhaps more pedagogical way to *depict* this dimension, calls for some figures from the universe of Walt Disney. They exemplify different "roles" or personalities of the strati of this dimension (C-4) and learning model in general. We start with *Gus Goose* ("Guffen") as low and little effective version of the *assistant level*. Gus needs to be told every detail of the task and he will not do more (often less) than he is told to. *Cousin Fethry* ("Klodrik") operates somehow higher up (on the *executive assistant level* perhaps). He executes certain tasks quite well, but his whimsical

variance is a low form of Darwinistic non-directed variation. In the middle of the spectrum, *Donald* as well as *Hue, Dewey* and *Louie*, are both conservative and innovative at the same time. The many tentatives of Donald (sometimes delegating tough nuts to the occasionally more reasonable nephews) to recombine old stuff into new functions[585] are often failing, sometimes due to hostile environments impersonated by neighbor Jones ("Jensen"), but more often than not due to a less than sufficient proficiency and experience in the actual domains. More rational is the rule-complying *and* effective counselor *'Helper'* (*guiding level*), that advises creative inventor and master *Gyro Gearloose* ("Petter Smart") at the highest level (*autonomous level)*. Gyro is approximating the criteria of an *absolute autonomous* inventor[586], mostly living secluded in his own "Leonardian world"[587], and only at rare occasions he responds to external problems of others. But then, Gyro does not live in our world, after all, other than that of childish peoples' imagination.

We have now circulated around several dimensions. Would it be possible to describe an actual MCs using a vector, consisting of values from each of these dimensions (and potential additional one's)? Could we e.g. characterize

Cypher: v1= [0.8; 0.4; 0.4; 0.3; 0.4; 0.6; 0.7; 0.6] and
EMI: v2= [0.7; 0.6; 0.8; 0.4; 0.6; 0.9; 0.8; 0.4] **?**
?: v_n= [0.1; 0.9; 1.0; 0.4; 0.0; 0.9; 0,5; 1.0]

And would it then be possible to represent Cypher, EMI and other systems as points in a three-dimensional coordinate system after having reduced the number of the notoriously fused dimensions down to three "compound dimensions", that ultimately describe actual systems as single points in an Euclidean space? And could we in continuation of such a project - model these features or properties of MCs as 'Voronai diagrams',[588] using similarity functions *and* prototypical examples only?

Cleaning up this garden of varieties?
Such questions are about the extensionality of the properties of MC systems and about the interpretations of massive opperlapping dimensions, especially between "behavioristic and cognitive dimensions" from above. It seems difficult to think of some rational and complete reductions into *better (*explicating) categories that distribute their members in orderly ways. One reason lies in the *intensionality of the use* of MCs. Many features of MCs describe *the way* we use MCs and *not* that what *happens* in a machine (such *behavior* of MCs alone might be possible to model rather precisely, see dimensions of C-4 of assisting vs. autonomous; appendix). The other reason is really more a symptom, namely the significant fact that nobody, according to Ariza's overview[589], has succeded in finding loopholes to get out of this state of relative confusion about categories. This very paper's structure, as a matter of fact, relies on some more informal grouping of MCs into chapters, with knowledge-based, learning, and evolutionary systems sorted into chapters **4, 5** and **7**, and the different types of models for systems are found as subsections of **ch7.**[590]

We will continue to discuss the relevance of some of these dimensions in **9.3** and **9.5**, where they will refigure under different names, "Pythagorean dreams" (B-1) and "LaMettriean temptations" (C-4), only redressed in other costumes. In the next section **[9.2]**, we listen to critical voices, arguments and questions as to the possibility of MC altogether. So then, what arguments could there be that have rational forces powerful enough to undermine the whole spirit and motivations in an ongoing evolution of Composing Machines?

9.2 Challenges and possible objections to Machine Composition

We now look at some common objections to MC, coming from several domains and perspectives. Unfortunately, there is not more space as is necessary to scratch some of the surfaces in these issues. We will therefore limit the discussion to indicative answers to the selected issues:

1. Issues from body and body movements
2. Issues from phenomenlism
3. Issues from intentionality and personhood
4. Issues from language and meaning
5. Issues about emotions
6. Issues about expression
7. Issues from morals and ethics
8. Issues from sociality and society
9. Issues about creativity, spontaneity etc.
10. Issues from miraculousity and souls
11. Issues from musicology

Q1: *How can machines compose music without having a body? Humans engaging in musical activities use their bodies in many ways. How can a machine do without it?*

A: Right, humans participating in musical activities have muscles and bodies and do often use them extensively.[591] Still, many music forms, e.g. listening to classical concerts in concert halls or churches implicitly ask for bodily restraint, both on behalf of listeners and the performing musicians. On the other hand, certain styles of popular music, especially where rhythmical grooves florid, will be uttermost physical and naturally stimulate moving bodies. But even though music *is* related in basic ways to dance and body movement, other forms of music are oriented at least as exclusively towards the mind alone (indian traditional music for instance). It is also true that the origin of music seems vitally related to dance, as investigated upon in **ch2**. Nonetheless, the gestural reality of many musical styles and expressions, coming from the singing 'cantilene' in melodies and dancing 'groove' in rhythms, may well be described successfully in machine models. Such topological or gestural representations of music are the subject of research today. All in all, MCs will not be able to move themselves,[592] just as little as human *composers* need to perform their ideas in bodily ways while they write them or present them. Even improvisational MCs' will not need to perform in sensori-motoric ways in order to include others' playing along with them.

Q2: *How can machines compose without experiencing sounds? Composers listen to sounds like listeners, machines will not really know what they do while composing sounds?*

A: Yes, true. The richness of phenomena and the experiencing of qualia (e.g. the immediate feeling of hearing a tuning fork in 440H[593]) is very much different from the information reaching sensors and their further processing in the software of a MCs. But what matters in being a composition is really the structural outfit, *not* its *realization* afterwards. (This may easily be confused with formalist positions. More on that in **9.4**). Therefore, systems like EMI produce note scores that are played by humans on a piano. The “impressional qualities” on this level are simply not what is at stake. Machine performance systems will have to answer to such questions more thoroughly. MCs are concerned not so much with single sounds, but with the relations between sounds in a macrostructure. Nevertheless, as we know from GTTM, effective musical structures will carry over meaning on many levels (see below) that is then experienced as “rich phenomena”. But such experientally complex functions are not *in* the composition but a *perceived* value in the human listeners.[594] The listening part of a MCs will just as well not hear more than relations only; but again, to the composition, all what really matters is its relational structure.

Q3: *How can a machine, that is not a person, but a mechanism, compose in the sense of intentionally communicating something to other persons?*

A: Well, you presuppose that musical works are objects of communicational content, or kind of messages. You may perhaps rather think of musical works as expressions *of something,* of an idea in itself. In this sense, machines that are programmed in certain ways may do very well. A work that is an instance of a musical style, doesn't necessarily need to communicate anything at all, other than itself as a structure. True, some composers like to think of their works as conveying some kind of cognitive content, e.g. a political or poetic statement; but most composers do not. (see also Q8). MCs will do ok in most of these cases. But this applies to non-autonomous MCs (C-4). Autonomous MCs may well exhibit both intentionality and personality. We saw that Cypher, at least in a limited sense, had self-awareness; remember how Cypher knew about its own inner states or conditions. Using a critic implies even a certain degree of self-reflectivity and possibly a sense of self[595] as well as individuality. Having a model of one's own particularity will in this context candidate the system for personhood.[596] Musically playing with different and seperate Cypher**s** will as a consequence be interpretable as playing with different musical personalities.[597] Dan Lloyd[598] has problematized such matters and argued for continui of representations and consciousness (reflections).[599] His model of mind is very much compatible with AI-methodology and is an example of early attempts to build a person or mind from simple persons or minds. Since humans with deficient higher brain functions are often denied personhood, it shouldn't be unthinkable to allow for machines to possess such functions? David Brin proposed identity, autonomy and agency to be the attributes of personhood, a definition that seems compatible with machine compositional agents in a future time. And philosophical definitions based on reasoning, and a conscious and persistent personal identity do not seem impossible to satisfy either. Having said that, MCs today are far from living up to such criteria, but neither did chess computers win over human master players only some few years ago.[600]

Q4: *How can machines compose without having neither language nor meaning, doing only calculations with numbers and digits?*

A: This applies actually to all representations of complex entities in computational terms. Anything can be computationally represented and manipulated, as long as there exists a translation or interface standard. Just like sounds are not really sounds in computers, but only digits or electrons, no languages or meanings are needed other than their represented ones. Concerning musical composing on computers, most sound complexes are handled as patterns of various kinds. We may well assume that also humans in some form process patterns **[2.4-2.6]** and such meanings are the result of the processes of com-posing relational objects **[9.3]**, and de-compositioning of these objects during listening (known as parsing). The aim of AI is to find computational solutions at appropriate levels to be able to model certain phenomena in adequate ways. There is no agreement about how this should be done in detail, but low-level representations[601] will co-exist with higher-level representations (gestural, geometrical[602]) and will ultimately be judged or selected by their success alone.

Q5: *Machines do* not *have emotions. Aren't emotions necessary conditions of music in its organismic understanding? If so, how can emotion-less machines produce relevant compositions?*

A: Right. Emotions and music have been in the continuous philosophical and even civilizational discourse. Platon viewed music with suspecting eyes. His "case against music" was based about "We are very conscious of her charms; but we may not on that account betray the truth".[603] These 'charms' have been object of continual debate up to our own times. Peter Kivy[604] makes the case that music first and foremost is a formal structure, but that its sonic patterns in "absolute music is a warm formalism....It has human warmth. Because it has human emotions as a perceptual part of its structure."[605] Kivy is very much opposing the view that music *expresses* what he calls the 'garden-variety emotions' (the conventional specific human feelings). In his view music does *possess* emotions, and that "there is a presence of emotive qualities in music"[606], but no content other than the syntactic and semantic relations guiding us in listening. He calls this theory *'enhanced formalism*' and according to it machines will therefore not need to have an understanding of those or any other particular emotions in order to construct sound structures, that in turn may excite emotions in human listeners.

Q6: *Doesn't music* express *some message or content from the composer or performer to the listener.*

A: Just like emotions felt during listening to music are *not part* of the music itself (Q5), but only phenomena that happen as a consequence of musical form, so do many expressed meanings in music arise due to listeners active and often conventionally associated interpretations. As a matter of fact, many people understand or hear very different things of *extra-musical* meaning, even in works of so-called 'program music' that are titled exactly to facilitate such expressive contents ("Fontani di Roma" or "The damnation of Faust" etc.). Often one confuses the messages of the text sung along with the music. But even then, people often do not understand the words. The idea of music as expressing something determinate is pretty much shaped during romanticism, where song cycles (Schubert, Wolf etc), opera (Verdi, Wagner) and program music florished alongside the equally radical idea of absolutism in music, proclaiming instrumental music to be the "queen" of musical expression of *intra-musical* meaning only (Hanslick, Schopenhauer). This polarization leads to several inconsistent and fragmented views about these issues. We will here just stress the *autonomy* of music *in principle*, i.e. without normative view, but support the contention that form alone (see Q7, and **9.4**) is sufficently rich in itself to produce those high levels of interest to listeners. As a consequence, composers may *want* to *mix* sounds with non-sound *associations*, but machines will do just fine without them.

Q7: *Human composers are moral and ethical beings and contribute in these domains to others by using intended musical meanings. How should machines be responsible and moral in doing such things?*

A: Some music is in fact composed by humans with the intention of making some moral point. Wagner's operas and some of Cages compositions are obvious examples. But also Beethoven's "revolutionary" symphonies, symbolizing the heroism of the French "freedom-fighters", or the music of national anthems and certainly a lot of rock and jazz music are all witnessing to this fact. Those musical examples and styles are used to pursuit *non-musical aims* of varying content and rigor. Machine music will need such functions only as much as the human co-composer actually lays the adequate premises and basic conditions for them to occur. Machines will execute such intentions on behalf of the co-composer but without a need to understand them. One might object, though, that people that follow such non-musical objectives, are *using* music and its sounds rather as a *tool* or in a functional way than as a goal in itself. We need therefore, as I see see it, not to apologize for "shortcomings" of content in relation to *any* imaginable wishes regarding the *potential* uses of musical communication.

Q8: *Music has social roles and contribute to societal issues. How is that possible with machines creating music?*

A: This seems to be related to the moral issues considered in Q8. In fact, many theoreticians stress this bond between music and society. Adorno stresses the socio-cultural situatedness and influence on people. He thinks further that music and cultural phenomena should advance critical consciousness, in contrast to the commodification of popular music today. Similarly thinks Jacques Attali[607] that music as noise can function like violence, and music as ritual means the subordination of individuality. When music becomes commodity and even decoration only, it will distance or alienate its users from their natural human agency. This positive view on independence in relation to critical listening (certainly not without relation to analytical listening as well) is somehow in disaccord with the social regulatory role of music, a role Stephen Mithen has expounded as well. But we may remind ourselves of the fact that many of the politically effective uses of music borrow their impact from text and other expressive channels than sounds alone. Bob Dylan, e.g. is often mistaken for being a musician, because of his underlining of texts with sonic patterns that may derive their effectiveness from the same sources than 'Hmmmmm' communication among Neanderthals. But neither the singing nor the guitar playing of Dylan is much worth mentioning in itself. What enters on stage is a personality with texts that fascinate some people. Again, as we saw with emotions, the extra-musical meaning mixed with sound is not integral, but seperate part really. And if people are increasing the potency of any kind of political or societal message with musical means, this is not a necessary condition for music to engage in. Curiously, Attali sees exactly composition, i.e. the active producing of music as the ideal relation

between music and citizens. And this is where MC may contribute one day. **[9.7]**

Q9: *How can Machines create novel things out of sponaneous acts from within the hidden treasures of the mind?*

A: This question will be answered more thouroughly and affirming in **9.3**. So much for now: Some acts of creatitvity seem to come like a flash out of the blue, like a spark. But sometimes we turn around a corner of a street, and might turn our heads. What do we see? Only the last few meters. Should we then wonder about miraculous acts of magic that placed us there without being able to see where we came from? I seem to get many ideas walking to and from the university. Does that mean that these arousing flashes of ideas come from my legs moving? Certainly not, and neither does a glass of wine contribute in other than peripheral ways to the fruits of minds being triggered by the temporary imbalances of my neuronal chemistry. Still, there are hidden treasures in our minds, but they are of this world only.

Q10: *How can a rational and shallow mechanism, like a machine, create the mystic and miraculous effects and deep meanings that are found in music created by human souls?*

A: Well, the fabulous and awe-inspiring qualities of some works of art are in reality the products of human minds, not souls, and those again are the products of evolutionary forces ("universal acids"[608]) and selections. Naturalized accounts of such souls and their products, such as the one known as 'Darwin's dangerous idea'[609], focus on continuities in the achievements of nature at all levels. The fact that artistic works might be the fruits of darwinistic engineering and not cultivated in mythic gardens of Eden or similar places, will not deride their sensual and cognitive qualities experienced by listeners of this world.

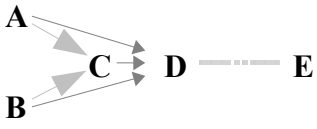
Q11: *The musicologist Nattiez claims that there are no musical universals, only perspectivism and relativism regarding the understanding of musical practice. How can one build musical machines without such a fundament of knowledge?*

A: Well, like we saw in **ch2**, representation is relative and without foundation really. But relativity of all musical knowledge, is no reason to become passified either. We saw that procedural and other naturalized epistemologies accept such a state of knowledge, but repeat that the truth is in the eating the pudding. So the "slipperiness and polyvalence of musical experience"[610] can be reconstructed in new and relevant terms. If this may not become the same reality than that of us humans, we may find other webs of musical meaning in them. But Nattiez is firmly right in claiming that an aesthetical objectivity is simply beyond reasonable bounds of imagination. This is a lesson we will have to remember when we try to express something relevant about Machine Aesthetical criteria.

9.3 Towards a framework for Machine Music Aesthetics?

In this heavyweight section we look at some intermingled issues and try to place Machine Composition into a wider context. There will be no definite answers, rather cautious approachments to a rowdy junction, during hours of traffic jam. In simple words, we try to put machines, music and the arts in a framework. Aesthetics is defined as either the empirical study of sensori-emotional *values*, or the normative *judgements* of sentiment and taste.[611]

To propose fundamentals in Machine Composition in an aesthetical perspective, we need:

x	***Arguments and aims***	***Functions* similar *to***	***Section***
A	first, argue that machines are inside the domain of Aesthetics, similar to how we witnessed music and machines share certain domains in earlier chapters, and	*Premise*	**9.3**
B	second, present *explanatory* theories and philosophies that then	*Premise*	**9.4**
C	permit us to construct tentative descriptions of *aesthetical traits* that music composition machines will have to satisfy in order to be labeled as *genuinely* artistic and creative entities.	*Induction (proposal)*	**9.5**
D	We will then try to relate Human Aesthetical (hAe) spheres to the Machine Aesthetical (mAe) spheres and specify the nature of this relation.	*Deduction*	**9.5**
E	Finally, we will put these results into a broader and darwinistic, hence naturalistic perspective.	*Analogy*	**9.6**
		*Graph*	-

First question: *Can machines be creative, intuitive, inspired and original?*

These questions have been treated in **ch1** and **ch5** already. Many human prototype "creationers", i.e. composers, exemplify especially two facts. First, the importance of hard work and dedication and secondly, the ubiquity of reordering or recombining of existing traditions.[612] Both seem implementable in machines (EMI, Cypher), but are they sufficient conditions? Many will mentalize features, often described like active shaping or struggling of the style-changing and personality-developing artists. So then: only humans are creative? Or is EMI, once introduced to learning and higher levels style-combining as well as exploring (trying out daring things or experimens), a worthy candidate to become a dynamic and in this sense "enlivened" form of learning and evolving "personality"?

Personalities have wills and likings (*intentionalities* and *preferences*). Hierarchical and non-hierarchical mind-modules, in the spirit of Minsky's 'Socities of mind', should be able to generate such functions. Already today, many MCs have propensities, i.e. tendencies to do things in certain ways rather than others. Such functional peculiarities will, as we may presume, eventually grow into full-size personalities implemented in machine architectures. Dennis Dutton[613] proposed seven universal signatures in human aesthetics (expertise/virtuosity, non-utilitarian pleasure, style, criticism, imitation, special focus, imagination). Except for the last, imagination, all seem pretty

much in the reach of machines. Imagination will be taken up again in the next question.
Robert W. Weisberg in “Genius and other myths”[614] systematically deconstructs and “psychologizes”[615] the by many cherished concepts standing close to 'creativity'. Weisberg writes: “... demonstrate that much of what we believe about creativitiy is not based on hard facts but is more or less folkore, passed down from generation to the next, as if it were the truth”.[616] His naturalizing project of “romantic” and superstitional conceptions of human capacities, unites creative actions from scientific and artistic domains. Weisberg explicates, in an impresssive account, some of the 'folk-psychological'[617] labels, including the 'aha', divergent thinking, genius, scientific discovery, artistic creativity and several others.
The seductive subtitle “What you, Mozart, Einstein and Picasso... have in common” tells us about his reconception of some historical events and phenomena, in terms that are compatible with modern knowledge, and especially cognitive psychology. His “reconstruction of Mozart” is very much compatible with the hints given in **ch2**, regarding Mozart's “unique” (here they come again!) learning cycle history and its consequence for the *naturalized Mozart*. And Weisberg's account of the genesis of the ninth symphony of Beethoven,[618] informs us about the hard work and dedicated efforts, and mentions little about “sparks from heaven hitting Beethovens's creative forces”.
Weisberg defines a 'creative solution' as one that is novel and effective.[619] Moreover, the problem or domain has to be *ill-defined* and the creative act requires the use of “imagination”. Imagination[620] occurs when “previous knowledge is brought to bear on a new problem due to similarities and analogies between the new and old situations.[621] Then he extends the subject:

> In addition to the creative *solution* to a problem, one can also talk about the *creative analysis* [my italizing] or formulation of a problem, which involves approaching a problem in a different way from the approaches taken by others. Such analyses are thought to play an important role in science and the arts. In science, creative formulation is often cited as the first step in solving a previously unsolvable problem. In the arts, such a formulation might involve the invention of a new art form that opened up a new avenue of expression.[622]

If this act of finding similarities or analogies turns out to be the key of applying novel solutions to ill-defined design problems, generalizing of various depth levels or levels of relationality may be a machine's *re*solution to the myths about creativity and its siblings.

Let us look at *intuition*[623] in this perspective. Intuition happens during problem solving when progression is slowing down or even halting. The “creative spirit” will then look *into itself* (in-tuitere[624]) i.e. into its experiential memories to find new recombinations at *deeper* levels than the current *operational level*, and hopes to detect missing links or detouring paths around the halting state of affairs. An example from composition: two typical situations lead to intuitional modes, first if problems seem impossible to solve at the current level of operation, and second, if one finds the obvious solutions boring or unattractive. In both of these situations, the creator may refocus and ground his search at deeper levels, in more distant experiential domains. When Bach found counterpointal dilemmas he might have used his ample experience as composer to rewire some connections at different places in his scheme or plan **[ch1]**. When Schönberg attempted to transcend his 'Verklärte Nacht' '[625], he may have found himself in an unbearable situation of “convention blockage”, i.e. there seemed to him no way away from tradition, at least from within tonal music practice. His “intuitional pump”[626] might then have brought him so much higher up in his abstractional premising, that he redefined, in traditional terms, the whole tonal system's order (twelve-tone-technique). A similar example is John Cages famous and infamous “4:33”, mentioned earlier. Cage seemed more concerned with issues *about* musical practice than musical *practice* and form.[627] The practice he questioned turned out to be the expectational nature of music as a whole. His questioning of the idea of “satisfying expectations” (see Meyer below) in music becomes the object of aesthetic content alone, and confused and virtually shocked most “listeners”. Could a machine come out with such an idea? If the modular architecture allows for varying degrees and

loci of reflection and reordering of premises, I do believe, that even '4:33' in some form could become the outcome of machines alone.[628]

Next is *inspiration*. Inspiration could actually be described as the very process of expanding experience with new information and often involves general perceptual stimulation, triggered or necessitated by creational impasses or "creational obstructions". Many artists (as well as scientists) listen e.g. to familiar music or take a "re-creational" walk in "inspirational" environments. Often, inspiration does not come from the new stimuli themselves, but rather from the refreshening[629] of a mind in a looping or tiring condition.
Both intuition and inspiration might actually be described as induction-styled reasoning **[ch5]** where information is supplied from the *inside* in the former case and from the *outside* in the latter case.[630]

When it comes to the assumed *originality* of persons, we should rather point to all the creative *acts* or *products* as original, instead of bundling them, associated to a person[631] (cognitive science). An original mind is one that has an experiental breadth and depth, presupposing causal histories that make the reconstruction of creative acts difficult and opaque to their publics. Often, minds that are original in the opinion of their contemporaries, can seem perfectly and naturally embedded in their context, in the perspective of future historians. So, originality seems to be a function of *lack* of knowledge (opaqueness[632]) about someone. An example are specialists, like craftsmen, solving problems in ways that *look* original only to the laymen, not acquainted with common practices. Finally, we mention Boden's distinction between P- and H-creativity, where the personal learning cycles are compared to the historical or compound learning cycles. This illuminates the distinction between personal and cultural styles or idioms as well.

Summing up, recombinations and their variations (i.e.creative processes[633]) are not mindless processes, but they involve the plans and intentional varieties of intelligence and personality (experientiel histories). This will help us to scare away the ghosts and myths of common-sense understandings of the historically shaped phenomena, literally created during romanticist memetic industrialization.

Second question:
Does an aesthetical entity need to have consciousness and 'holistic human-like properties'?

This second question is not about the nature of a creative process, but about the integrity of the creating person and the idea that a holistic entity is a necessary condition for aestheticity.
We know that *many* people *have* compound structures consisting of perception, intelligence, creativity, consciousness, phenomena of qualia, sensations, reasoning, introspection and so on, without ever to become aesthetically active, i.e. behaving artistically. So, the integrated nature of human existence, so to speak, seems *not* to be a *sufficient* condition after all. Is it a necessary condition, with the consequence of making machines become natural strangers in the aesthetical sphere? Phenomenalism and various forms of subjectivity-centered philosophies have made this case and believes body and mind, and hence movements and sensations (qualia), to be in principle impossible to disentangle or disjoint from each other without loosing the essential or whole picture. In other words, an algorithmic machine will never be able to become an aesthetical entity since it by assumption is not able to experience qualia and consciousness. Conferred to the domain of music, which comes precisely out of entangled origins of movement, sounds and holistic 'Hmmmmm' **[2.1]**, we might therefore quickly surrender and conclude that algorithms or machines will never compose music *like humans?*

This is the statement of weak AI at its maximum interpretation. We leave the weak/strong AI issue for another while, but suggest here only the following argument: the fact that music "comes from"

integrated phenomena, does not make it impossible to disjoint them later in practice and in the same time preserve the essentials of the equation.[634] For instance, the fact that humans have 'been fish' in some distant evolutionary past, and still pass this phase in utero, does not mean we have to be somehow "fishy" today, does it? In other words, things formerly integrated can be further modularized later. We may even assume (following common-sense) that by concentrating on smaller components, one *can* augment their specificity. This question is about reductability and particularization of phenomena, encountered already in the process of discretization in **ch2**.
We may here content ourselves with the observation, that humans today *do* musical activities without singing, without moving their bodies,[635] and without embedding its meanings in the totality of human experience.
Therefore we may say, that entities with only partially generalizable experiential areas (over sets of domains) will be able to engage in imaginational processes leading to creative acts. In Minsky's terms, it may be sufficient to build modules of analogy-seeking and similarity-detecting functionality to initiate the game of building consciousness bottom-up in incremental steps.[636] This will then allow for a concept of a conscious machine, however rudimentary it may be. In effect, many of the high-spirited and possibly somehow greedy proponents of strong AI, such as writer and theoretician Vernor Vinge and Ray Kurzweil believe in the future event of *singularity*, where AI is supposed to transform itself into becoming a *strong artificial intelligence*, that according to these authors will cause sudden and dramatic changes in societal realities. I doubt this to happen soon or even happen at all, but in any event will MC enter an era of relatively autonomous musical intelligence and creativity (see **9.1**, C-3).
All the same, there will remain domains of human creativity, where machines at least out of more practical reasons will be excluded from. The most evident fact of this line of argument lies in the *cross-experiential* or *inter-modal* qualities of the brain, especielly multiple sources exploited by the imagination,[637] e.g. the comparisons between philosophical ideas or food sensational features with sound features seem rather far (and unpractical) from even an imagined machine compositional practice. These extra-musical meanings and expressions of music are the subject of our attention in the next section.

9.4 Two theories that explain music

There are numerous theories of music looking for full-blown accounts of the nature of music and specifically about the power it exerts on humans. Bowman's "Philosophical Perspectives on Music" is the obvious place to look for a comprehensive survey today. We concentrate here on two approaches, which I believe to be of special relevance to issues of MC. Leonard Meyer's "Emotion and meaning in music" and Nelson Goodman's various descriptions of symbolic systems in artistic domains[638] and elsewhere. Both are cognitive theories and as such talking the mother tongue of AI as well. Both are therefore concerned with the cognitive mechanisms, like perception, memory, imagination, and in general interested in the meaningful constructions of the mind of various shades. Both are making theories about communication, Meyer more specific to music, Goodman in the most general sense thinkable. We start with a matrix of three notorious distinctions that are responsible for the never ending debate about music's innermost character. *Formalist* positions argue that meaning is the result of primarily intellectual understanding of the musical relationships found in the musical work. Expressionists will say that the same relationships are exciting emotions (feelings) in listeners. The *absolutist* view[639] is stating categorically that extra-musical meaning is non-relevant, musical meaning lies exclusively within the context of the work itself. Finally, the referentialist view, opens up for the whole extra-musical world of thoughts, feelings, concepts, personal characters and so on. We notice that the first and the last couple of concepts are opposite views, but belong to different dimensions (intra-musical/extra-musical, intellectual/emotional). Apparently, there seems to be no historical progress in music philosophy about these matters, since all these theoretical stances have been defended or criticised since antiquity, up to our times.[640]

	Meaning derives from form?	***Intra-musical meaning?***	***Meaning intellectual?***
Formalisms			
Expressionism			
"Emotion and meaning in music" (EM)			
Absolutism			
"Languages of Art" (LA)			
Referentialism			

According to Meyer's theory (EM), a person that listens to new sounds or music will always have knowledge about styles already ("more or less complex systems of sound relationships understood and used in common by a group of individuals"[641]) This background generates expectations in succesive events, that are either *confirmed* or *frustrated*. The unfolding stages of alternating experiences of uncertainty are the core engine of musical meaning in music. The resulting "embedded meaning"[642] is emotional in nature, because:

> The customary or expected progression of sounds can be considered as a norm, which from a stylistic point of view it is; and alteration in the expected progression can be considered a deviation. Hence deviations can be regarded as emotional or affective stimuli. [643]

This formal-emotive transition is grounded in Meyer's Inhibition thesis that "emotion or effect is aroused when a tendency to respond is arrested or inhibited".[644] Such feelings are undifferentiated and physiologically unspecified, in other words abstract emotions responsible for non-designating references. But they, in turn, arouse basic feelings of disorientation or suspense. e.g. non-familiar music will imply uncertainty or ignorance that causes feelings of "impotence", "apprehension and anxiety".[645] A listener will in the course of musical experience learn new patterns of antecedent-consequent-relations that expand pattern and style knowledge continuously. Since the mind hates uncertainty, it will contribute with its tendency to fill gaps ("The mind, for example expects structural gaps to be filled"[646]), and make use of innate processes like *grouping, closure,* and *good continuation.* Those processes are basically in line with Gestalt psychology, but Meyer puts them in a primarily information-theoretically perspective. In fact, he stresses the importance of probabilistic thinking in many places of EM.[647]

The theories of Nelson Goodman are both different and similar to EM in various respects. First of all, EM is a theory about music only, where Goodman's theory of symbol systems (LA[648]) is not only comprising the arts, but all human activities of cognition, including the sciences, language and philosophy. But both are cognitive and communicative theories about music. Both see themselves in opposition to the dichotomies between intellect and emotions. Goodman's theory may well be called one of the most radical theories in these domains today. His nominalist[649] stance, i.e. both irrealistic and anti-idealistic, and hence building on conventionalist and relativist foundations are at the core of his theories of symbols and symbol systems[650]. His detailed description and categories of symbolic modes and activities (referring in general) are extending to all domains of human life, but sometimes requiring from us to loosen or even relinquish concepts and understandings that are in conflict with his symbol-theoretical approach. His theory leads us to, what we may call, paradoxes about the definitions of *musical work , notation systems* and other issues in music. On the upside, his theory is unifiying all forms of experience and knowledge domains, including different art forms in one and only *symbolizing extensional domain*. In relation to music, this means e.g. that the notational system permits the precise characterizations of syntactic *and* semantic (determining the

form and range of symbol scheme and systems) aspects in terms of disjointness and finite differentiation for syntactical requirements, and in terms of unambiguous relations of 'characters' to their correlated 'fields of reference',[651] semantically disjointness and finitely differentiation as semantical requirements.[652] Symbol activities that satisfy both of these levels are notational *systems* and music is its prototypical example (technically termed 'allografic' contrasted with 'autografic, that are densely symbolized paintings e.g). As a consquence the level of *precision* in communication is *in principle* very high for music, according to Goodman's interpretation. This makes it *in theory* the most *extensionalistic theory* of music today. In general, Goodman distinguishes between denotation, and exemplification. Where language uses both, music uses various forms of exemplification. Goodman's account of metaphorical symbolizing is based on a kind of *referential dynamics*, where rearrangements in a field of reference are controlled by resisting and attracting forces of metaphors ("a metaphor is an affair between a predicate with a past and an object that yields while protesting"[653]).

In music, metaphorical use is not made with denotational labels, as we know it from language, but with the exemplifying symbols of music. This means that for example "sad music" will not mean 'sadness' in itself, but "only" *metaphorically exemplify* this feature by convention or symbol-interpreting use. This mode of symbolizing is called "expression" by Goodman, but in a theory-specific and from common-sense derailed way. Expression is commonly understood as an emotional communicational form, while it in LA is a technical term designating a cognitive form of exemplifying a feature metaphorically. We may call it G-expression to avoid misinterpretations. But how can something be exemplified (= possessing) only metaphorically (≠) ? Isn't this a classical contradiction in terms? We cannot follow his theory into the necessary detail in this place, but must content ourselves with some citations with illuminating powers:

> [The relationship between a symbol and what it symbolizes] is never absolute, universal or immutable.(LA,50) Goodman's applications of the notion of exemplication to art is in many ways enlightening, by allowing him to explain significance of certain features of artworks in referential terms... Goodmen's attempt to account for the nature, interpretation, and values of artwork fully extensionalistically (i.e. just in terms only of what they refer to)... [654]
>
> Representations, like metaphors, are creative. They classify objects and make relations between them. Our perception of nature depends on the ways the arts and sciences present it to us. Art and science, in that sense, create the world. (Daniel Cohnitz &Marcus Rossberg, *Nelson Goodman,* Montreal, McGill, 2006, 171)

This last citation reminds us of the importance of art and music as well as the ubiquity of creative and constructivist forces in all symbolic activities. In fact, to Goodman, there are no real worlds or descriptions of them, only different versions of world, without any particular world behind them. This radical constructivist approach is paired with his building of a very precise theory about symbol systems with sufficient formal power to explicate common-sense language, and perhaps musical sound events as well, in scientifically satisfying symbolic practices.

Summing up, Meyer's EM proposes a more static model of cognition, while Goodmans cognitive approach is wholeheartedly dynamic and "worldmaking". Both are contextual theories of cognitive processes, but Goodman stresses the active understanding and changing of the meanings of symbols and over time the meanings of entire symbol systems.

> Constructivism in psychology is the perspective that we all construct our own perspective of the world, through individual experiences and schema. Constructivism focuses on preparing the learner to problem solving in ambiguous situations (Schuman, 1996[655])

Goodman, but especially Meyer have been criticised for their *formalist* aspirations[656]. But understanding of their approaches in their own terms, such criticisms appear more like caricatures

of these theories. In perspective, looking at the "graphical table",[657] depicting music philosophical borderlines (above), both theorists acknowledge firmly that meaning/symbols derives from extra-musical and non-intellectual activities as well. Both are affirmative to questions as to whether meaning derives from form, but these relations transcend the simplistic and superficial layers of formal relations attributed to the formalists or non-existing puppet targets perhaps. "Greedy" formalists will be those that explain everything from atomistic elements all the way up.[658]

Both Meyer's dynamical and expectation-based relations and, even more so, the intricate and unlimited ranges of symbolic activities sketched by Goodman are clear examples of domain-transcending theories ending up with respectable records of relational richnesses.

9.5 Are there aesthetical traits that characterize (generalized) machine composition systems?

We have opened the pandora's box in **9.3** and described it in its most general terms available in **9.4**. What criteria can define the aesthetical nature of MCs, criteria we might use in assessing aesthetical value or at least judging particular MC systems to be aesthetical systems or not?
Would it be sufficient to find high complexity of their functions and results or would a high degree of compliance to norms and traditions be satisfying conditions? Neither of these are enough, but they certainly are not without relevance. So, let us now look again at information theory and generalized observations about communication:

> Only absolute *certainty* banishes entropy absolutely
> The most uncertain situation has the maximum *entropy.*
> *Information* relates to the breadth of what could be communicated.
> *Knowledge* is a distillation for the regularity and order arising from a communication.[659]

We may say that too much information is *aesthetically overkill*, and too much redundancy is aesthetically *killing bore.* During the entire music history, composers knew that music had to offer likeness and difference; or repetition and contrast; or similarity and deviation; or convergence and divergence. We can continue as long as we like, it seems a very first fundamental law of art and *interest*. Too much information or contrast/non-familiarity and we get disoriented or afraid; too much redundancy and we get disinterested and even asleep.[660] In brief, good music has *some* redundancy and *some* entropy in *right* portions and proportions. Proportion means here the right *way* to order them. The sonata form is a regulating mechanism to ensure that repetition, variation and contrast are taking place in the right portions and proportions/places. The sonata form bears even similarities with fundamental narrative structures, as exemplified in the travel or homecoming dynamics.

> If the entropy is too high, the music is too unpredictable and the listener eventually gets frustrated and stops listening...when entropy and redundancy sustain a fluid, dynamic balance, there is enough regularity to orient the listener in the music but also enough novelty to preserve interest. This suggests that in general, **Composing is about the manipulation of interest, affect, and attention**. If audition and memory are the engines that drive expectation in music, expectation itself is the beginning and end of music.[661]

This is exactly the resounding of the essence of Meyer's EM where he states that "Expectation is a prediction based on current and past experiences. Musical meaning is a function of expectation."[662] Actually, we see that variation possesses both novelty/contrast and familiarity/repetition at once, it occupies the "golden mean" position between repetition and contrast.

Now we go on with the "maximum non-linear repetition"[663] or less cryptical the *dynamics of contrast*. When it comes to proportional or mathematical *upscaling* of compositions, something one

finds in Bartok's methods with 'golden means' and Debussy's compositions [ch1] among many others, there will have to be assymetrical features as well, otherwise as it has been demonstrated, our musical minds see through the plots of algorithmic-like "balooning" of repetitive varieties of a subject. Fractal models [ch 7] may circumvent this ideational scarcity substituting it with gigantic variability. Unpredictability, derived from contrast, seems in any case to be a necessary feature of aesthetic information:

> Structural predictability can only be useful to a composer up to a point because music is designed to gain and maintain interest, and this requires a certain degree of structural ambiguity. Consequently, materials may be ordered , combined, disordered, recombined in a manner that defies easy analysis. (Loy, *Musimathics,* 350)
> ... some of the greatest music is great precisely because the composer has not feared to let his music tremble on the brink of chaos, thus inspiring the listener's awe, apprehension, and anxiety. (Meyer, *EM)*

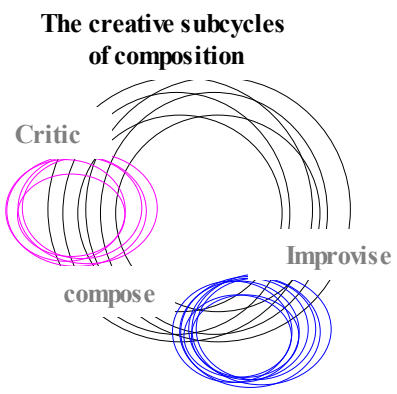
Does this mean that a touch of non-deterministic decisions[664] may be part of an effective composing algorithm? I do not think so, because even though there *are* occasional situations where minor break-downs of regularity or expectational regime *will* be an artistic means to some related end, in general non-deterministic methods, however refined with seeding, random sieve or constraining of their values, will ultimately contradict the essential needs of listeners matching the target sound *sequences* or *patterns* with their idiomatic style experiences. The dimension of C-3 from above in 9.1 will therefore preferably be densely populated on its determinist pole alone.[665]

> The efforts in this chapter to generate compositions by rule have so far shown no particular musical intelligence. Because all values chosen by the *Random()* function are strictly independent, the music created directly from it is unsatisfying; it lack the glue – redundancy – that binds music together.[666]

But then, what about "breaking the rules" and Cage (see '4:33' above)? The answer is clear by now, there is "no breaking rules" without alienating oneself from one's listeners or culture altogether.[667] To "meta-question" our musical practice, like Cage did with '4:33' and others do with similar experiments, and thereby putting a literal question mark after our 'expectations', effectively makes them to renounce from their public. A similar example from the practice of chess playing would be to refuse to "recombine the pawns and rooks", thereby stating simply one's opposition to competition altogether.[668] Competition is the silent premise of chess, so is "fulfilling" a sudoku board a silent aim [ch1]. If we stop putting numbers into a non-finished board of sudoku and intend to declare that filled boards are ugly or boring, we are in effect *not* breaking rules, but *making* new ones at more elevated and fundamental levels. Having generalized some steps, we may think again of Bach, Debussy and Bruckner from the first chapter. None of them broke rules, but they constructed new ones. As we saw above about creativity, rules are re-made, invented and discovered whenever we encounter problems. Ambiguities are sometimes solutions, sometimes problems. In Bach's case of "breaking rules"[669] in the *right way,* meant the hiding of dissonances in the most appropriate places.[670] Bach's aural sensitivities, ie. feedback and training, told him to look for places where todays music psychologists confirmed that "masking"[671] is occurring. It is a *creative solution*, as defined above, but not involving a genius and overruling, only a practical weighing of conflicting rules waiting to be placed in a bigger picture or simply stated an *craftly expanded rule system*. Similar arguments are applied to Debussy and Bruckner, as well as the vast majority of artistic composers. The description in ch1 of a numerophile as well as numerobsessive Bruckner, is that he struggled with his problematic entourage and his disciple-like and immature behavior, choosing to identify and act as loyal *follower* of his counterpoint teacher Sechter. All this does *not conflict* with his being one of the music history's greater composers. His rule-following and adapting is hence no disqualification. As we know, styles and idiom are the background against that all art will have to be understood, communicated and judged. This is in marvelous harmony with Goodman's general idea of the idea of "symbolically remaking of our worlds". Loy and Meyer are in no way different opinioned in this matter:

> Music is like a field, bordered on one side by order and regularity and on the other by surprise and irregularity, and the most effective musical domains lie in the middle ground between these borders. Redundant elements communicate a *sense of order* that is embodied , for example, in the regularities between various parts of a musical composition. *Taste* is reflected in the entropical elements, and *style* is revealed in the pattern of trade-offs made by the composer between order and taste. If we appreciate the sense of order, taste, and style in music, we appreciate the *intelligence* that informs the composer's work.[672]

So, did we find criteria or traits of aesthetical nature? Are the *trinity traits* of repetition, variation and contrast in adequate portions and appropriate proportionalities the key to test machines for aestheticity? Even if we may have stumbled in the right direction, those rules or traits are very fundamental and general, they say nothing about the precise nature of neither machine architectures and paradigms, nor about their products' characteristics. I think nevertheless that we are where we should be in these matters. Let us look again at the learning cycle in a composer perspective. The composer is not just creating out of the air, as we know. His hard work of adjusting old models and balancing aims and claims is more a reconception than a creation. Therefore a composer is able to be his own *imaginary critic* and his own *imaginary muse* by investing into listening, applying the aural sensibilities[673], improvising in search for the one golden recombination that he might trip into and in continuance exploit the reconstructive plans and acts that will mold new artifacts out of the dying old.



> In constructing a system, then, we seek to achieve an acceptable balance among competing claims. Both the character and their criteria for an acceptable resolution depend on a variety of factors including the range of presystematic sentences we expect our system to preserve, the goals we want it to realize, and the amenability of the domain (as we currently understand it) to the sort of system we seek to construct. Often alternative resolutions are equally reasonable. Pluralism results. A number of independently acceptable systems can be constructed , none of which has a claim to epistemological primacy.
> Symbol systems are artefacts. Their construction and their application are subject to constraints. The interconnected questions of what constraints are legitimate, what symbol systems are constructible, what worlds they define, and what sorts of understanding they yield are central to epistemology. [674]
> (Catherine Elgin in Goodman/Elgin, *Reconstructions,* 24/26)

The learning cycle with subcycles during creation implies that MCs will apply both material-*augmenting* modules (variation-producing) and material-*decrementing*[675] modules (critic-selecting successful material). David Temperly (“Music and Probability”, 2007) has looked at this dynamic process and the communicative pressures in the learning cycle continuation of composition (including players, listeners etc.), and uses computational models of *Bayesian probability constraints* in these processes. His probabilistic preference-rule approach is equally interesting to analysis than composition activities. His use of notions like *cross-entropy* that measure the “goodness of fit between a model and a body of data is hitting our theoretical “expectations”. And it shouldn't surprise us then, if Bayesian preference-rules, very much in the spirit of Meyer's expectational paradigm[676], would become a practical example of such implementations of the trinity-traits growing into the functionalities of the critic, both inside and outside of the MCs, in possibly multi-participant-models (multi-Cypher) or even in the Frankensteinian evolutionary environments from **ch7**, thereby copying nothing less than the primary creational process of evolving artificial creational systems:

> In a sense, the communicative pressure brings us full circle. We began our musical investigations by addressing problems of perception – meter-finding and key-finding. I suggested that these perceptual processes were guided by assumptions about the process whereby music was generated. I argued further, that listener's generative models are tuned to reflect the statistical properties of the music that they encounter; thus in an important way, perception is shaped by production. But now we see that the influence also flows the other way. **Composers (and performers and improvisers) wish for their intended structures to be perceived by listeners; but not every possible style is equally conducive to this. The desire to communicate exerts pressure on productive musical behaviors, and thus, on the evolution of musical styles**. Many questions remain unanswered here, and many avenues await further investigation. But I hope I have shown that the Bayesian approach provides a powerful set of tools for investigating music perception, production, and the complex ways in which they interact. [677] [my emphasis]

The trinity of traits will essentially be as good as its implementations, and this is how it should be, since our search for principled traits must come in the context of a *naturalized aesthetics* after all. The guiding thread in the theoretical chapters of this thesis are very much subsumed in this naturalizing project. Such a project is, if not identical, by all means compatible with the irrealism of Goodman's *worldmaking aesthetics*. Goodman has his own criteria to "judge" symbol activities as aesthetical. But the integration of human understanding makes it impossible to declare them as either necessary or sufficient conditions. The "real" world is always bigger than our symbol systems trying to encompass or embrace it. Therefore Goodmans aesthetical criteria are five *symptoms*, that collectively tell us, at best, about an *aesthetic gradient* from the non-aesthetic to the highly aesthetic, reminding ourselves again of the unity of all kinds of types of endeavor undertaken by humanity:[678]

I. syntactic density
II. semantic density
III. relative (syntactic) repleteness
IV. exemplificationality
V. multiple and complex reference

How can machine models be said to show these symptoms? They can in principle, because Goodmans's extensionalistic system goes hand in hand with absolutely ontological tolerance as its principle, in the same time that it asks for high levels of consistency, even with the preceding symbol systems and their labels. Density (I,II) and repleteness (III) are precisely defined logical terms, so are the extensionalistically defined exemplification in all its shades (IV) and the compound expressions of multiple and complex references (V).

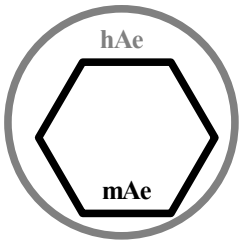
So, we might not exclude the possibility of a constructivist project where complex symbol systems are build from ground up, suitable for MCs, and allowing for the requested levels and relational richness that make up aesthetical symbol activities. In this setting, there are few principled obstacles left; even a referentialist conception of music (see table above), with G-expressions from musical symbol systems, that metaphorically exemplify features and objects from extra-musical domains, presupposing nothing (sic!) more than the existence of another Goodmanian symbol scheme or system describing exactly *those outside* symbolic domains. It seems there could be a bright future for "The Structure of Appearance" (Goodman[679]), once combined with the power of AI and their "machinist"[680] abilities and possibilities.

Consequences: How relate the Human Aesthetical sphere (hAe) to the Machine Aesthetical sphere (mAe)?

The standard answer, as we saw in several places in this text, is that machines and humans fundamentally pertain to different worlds, the technological and the biological or cultural. But is MC really that different from human composition (hC)? And should hC *entail* MC or be entirely separate in terms of its essence, or nature; what I here call sphere, informally defined as the domains and processes.[681]

Fig.9.5.2: mAe ∩ hAe = Ø

or



?

Fig.9.5.3: mAe ⊂ hAe

If we accept that MC and its operational domain or aesthetical sphere (mAe) operates within the aesthetical sphere of humans (hAe), we may say that mAe is entailed by hAe **(fig.9.5.3)**. This is probably the common sense view in reasonable circles, since they accept that parts of hAe can be done by MCs.
We may call both these views *traditional views*, *excluding* and *including* respectively.
Examples are the programs of limited intelligence, doing the repetitive parts of creative composing, the assisting level ("workers") of our C-4 dimension from **9.1**.[682] But is really every outcome from machines in mAe either compatible (entailed by the domain of hAe) or non-compositional in the aesthetical sense?

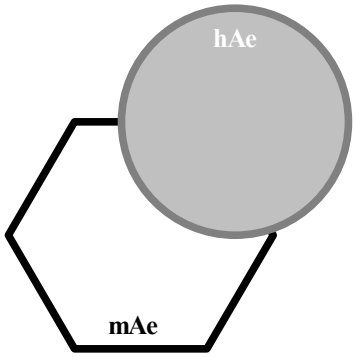


Fig 9.5.4: mAe ∩ hAe = assisting machine

This would be the model where the hAe is the *only* aesthetically *valid* extension (grey area). The part of mAe that overlaps (subset of hAe) is hence the only acceptable part of mAe, copying somehow human aesthetical activities, but not more. The white area of mAe (outside of hAe) would in this view entail machine musical projects of other interest than the aesthetical, e.g. information-theoretical experiments or music psychological tests or eventually a sphere where machines might communicate to other machines in music-like but not aesthetical ways. We will call it the *arrogant view.*

Next is the view that mAe is identical to the hAe. They entail each other. This is the strong AI version of MC, and I have already stated my reservations about it **[9.3]**. It is the contention or belief that machines will do composing *just as* (and not only just like) humans and at least at the same levels and the same way. This is effectively named as the *greedy view.* Some formalist composers may have hoped in the truth of this model.

Fig 9.5.5: mAe = hAe = assisting machine

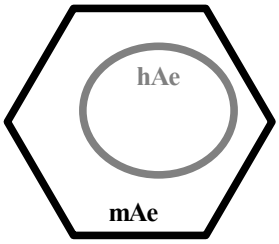
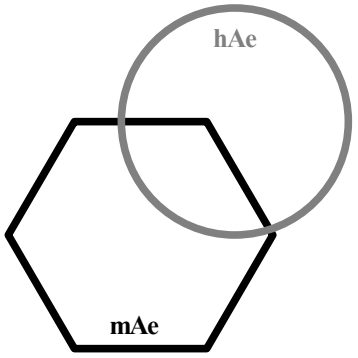
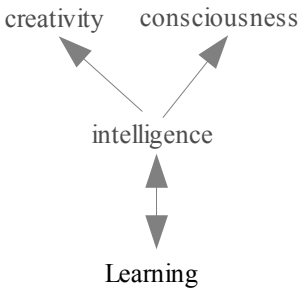


Fig.9.5.6: hAe ⊂ mAe

Then we may imagine the *eliminative view* ("extra-strong AI"), that states that all human compositional practices are done perfectly well by machines (greedy view) *and* machines will be expanding this aesthetical domain, a domain that today encloses human compositions with new areas of creative compositions into the *extended mAe*. That means hAe is a proper subset of mAe (hAe ⊂ hAe). The underlying premise seems to be that since humans basically are physical machines, there should be no reason, in principle,why humans not make a subset of a more general machine category. This seems too reductionistic to some, but such views are defended by futurists like Kurzweil, Wolfram and Hans Moravic that writes that as a consequence of hardware evolution, will the:

> "forth generation" robots exceed humans in all respects from running companies to writing novels. Such robots will evolve to such lofty cognitive heights that we will stand to them as single-cell organisms stand to us today.[683]





Finally, the view I will defend in this paper, a view I will call the *elimino-concessive view* in lack of a better name. It is yielding somewhat to the arrogant view regarding the superiority in kinds of consciousness-related functionalites of living brains. As I expounded this point in the answers to the second questions of **9.3**, I regard the cross-experiental or inter-modal features of brains, phenomena studied as consciousness-related processes as *difficult* to model computationally. This is not a concession to ontologically conclusions of any sort, but simply the acceptance that these features are too far from todays machine worlds as making it realistic to fable about it. As a matter of fact, I think that humankind has to lift a lot more stones in its search to understand itself, including the so-called mysteries of the mind, before pessimist voices should be given the microphones. But it *is* a clear concession to the fact that human composers, at least for some considerable time, will be specialists or experts in the connecting of meaningful information between extra-musical and intra-musical elements of sound structures.

We may therefore denotate the area of hAe (hAe – mAe) as the *exclusive part of human composition*, while sharing an intersection with mAe (hAe ∩ mAe) , that I want to call the *competing part* of human and machine composition. It is the aesthetical area of composition where both machines and humans are proficient and effective to varying degrees and qualities. This will be object to contention from both sides, somehow like non-negotiated borderlines between countries will have to await their final settlements. These borderlines on *both* sides will probably move in one or the other direction due to the evolutionary developments in both camps of composition, partly caused by their reciprocal stimulations and motivations as a consequence of machine-man-cooperational contexts. In addition, the intersection (hAe ∩ mAe) may be the locus for implementations of the trinity of traits in the style of GTTM or any other deep-level describing rules of Common

MusicPractice [ch1] today.
We now come to the more provocative part of this view, the eliminative claim is accepting the revolutionary nature of technology and AI in the creative domain. Not only does this imply that human intelligence can be superceded by machine intelligence, but it means even more so that human creativity is *not* ontologically tied to biological brains,[684] but may successfully be modeled on some kind of machine architectures (in the future or now, see ch2), and eventually gain enough ground to win this kind of battle *in this particular domain*. In fig.9.5.8, we encounter again the proposed "hierarchy" of the human mind. What I am saying, after all, is nothing more, than that human compositions may *specialize* along the *right path* (of consciousness or whatever[685]) and machines may *specialize* along the *left path* (of creativity). The common territory is intelligence, where machines of various types, such as 'Deep blue' (chess), databases (memory) and 2$-calculators successfully outperform humans already. Even the higher strati will be enlivened by both participating "ingroups", but succeeding there to only varying degrees.
As a matter of consequence, we see easily that this union of machines and humans leads us to a union of two spheres, the aesthetical spheres of human *and* machine composition. In other words, the entering of machines, is *not* the "march of the machines"[686] that drive humans *out* of the paradise, but an enlargement of the *total* aesthetical and hence experientiel sphere of humans. The resulting union ($hAe \cup mAe$) is also the union of the optimal capabilities and abilites of machine and man respectively. It seems a win-win situation more than a 'race to the bottom' competition.

But exactly how much of the hAe will machines be able to populate? Let us again take the case of the learning cycle, extended by the subcycles of specializing composers. They use what Loy calls their 'aural sensibilities'[687] to select their sketches and experiments on their way to the final drafted composition. This is not unlike the functionality of a 'critic' in a MCs. In many of the advanced examples of MC, such as Cypher and Frankensteinian systems,[688] multiple critics at different levels will cooperate to select material, algorithms and even learning, ie.change their own principles.

This triggers two observations. First, the modular architectures are found to be realities in the brain (neuro-scientific models of the brain; 2.1), and second, the modular methodology proposed by Marvin Minsky (Societies of mind) and the procedural methodology advised by Abelson and Sussman [ch2]. They all claim that higher level functionalities may arise as emergent properties from the modularity and complexity of relational structures. In a similar fashion, Meyer speaks of the probabilistic style-generations coming from lower level expectancies that again derive from antecendent-consequent relations. Today, many feel this to be a feasible task. Intelligent agents of AIMA are the 'abc' of AI. But putting agents, learning and certain environments together we obtain meta-agents and adaptation as emerging properties of complexity. John H. Holland was one of the pioneers of machine learning. In his "Hidden Order"(1995), Holland looks at such processes, where adaptation builds complexity out of building blocks, tags, internal models, and *properties* such as aggregation, nonlinearity, flows and diversity. Such studies of complexity and properties of living entities may lead us closer to the autonomous level of MCs, described in 9.1/C-4. Hollands adaptational processes do bear some similarity with the trinity of traits as described in the probabilistic theories of Meyer and Temperly. They certainly fit well into the picture of genetic algorithms and evolutionary programming. This will be taken on in the next section. The emergence of progressively higher critcs may well serve first and foremost the concrete selection of better designed MCs, but ultimately even function as generalized Critic (super-critic[689]) of aesthetical judging.[690] In this chapter we were out for a framework of MC aesthetics only. The analysis resulted in the trinity of traits and is exactly what we were looking for, a *frame* that will be the structure to *support* and *contain something*. With progressing levels of machine power, architectures, AI and adaption-based technologies, these frames will be filled and stuffed with aesthetic implementations in the same time as they will be judged by the emergent critics of critics until we reach the generality that we today call aesthetic norms and overall values. Those aesthetics may then open up

our conventional sensualities and aural sensibilities to the expanded norms and options thanks to the new territories in mAe. Whenever we should get visitors fram Mars or other places, they might get more excited and aroused by the expectancies deriving from machine made aesthetical norms than we humans strive to incorporate in our understanding. Some may certainly feel this as a Brave New World declaration. It shurely is, and still, we are not finished yet.

Until now we cannily and prudently remained out of the referentialist domain of extra-musical meaning. It was defined above as the integration of extra-musical meaning with overall musical meaning. We may call it the *Wagnerian position* with reference to his 'Gesamtkunstwerk' conception. Goodman, as well, seems to be appealed by multi-art-works (also called multimedia). Is there really no chance for MCs to develop (or learn) such extra-musical meanings? We may revisit the theory of symbol systems in the light of our model of aesthetic spheres once and for all. Let us first expand the formal model of the spheres with the non-aesthetical sphere surrounding it (see **fig.9.5.9**). In the figure we see now also subsets of spheres, that are called styles, and that may overlap each other or cross spheres. Styles are given names or property-labels ($P_1...P_x$). Let us tentatively make an interpretation of this model.[691] The whole domain of reference is C in the aural domain and T in the total domain of experiences. We may now split them into more discriminating areas, such as for example:

> $M_1=(((B - A) - D)- C) \cup F$; this would in model T of the total domain (fig.9.5.11) denotate the union of the aesthetical spheres of pure machine music and pure machine literary art.

But we will leave the further specifications and discussion to other places.

A: human aesthetical aural sphere;
B: machine aesthetical aural sphere
{x ∈ C: P_1 (x)}: P_1 = Elevator music
{x ∈ C: P_2(x)}: P_2= Shopping center music
{x ∈ C: P_3(x)}: P_3= Functional sounds?
{x ∈ C: P_4(x)}: P_4= new machine style
{x ∈ C: P_5(x)}: P_5= machine style with GTTM+
{x ∈ C: P_6(x)}: P_6= heterogenous-style
{x ∈ C: P_7(x)}: P_7= Mozart style
{x ∈ C: P_8(x)}: P_8= Bartok style
{x ∈ C: P_9(x)}: P_9= Jazz style X_1
{x ∈ C: P_x(x)}: P_x= Top 20 style?

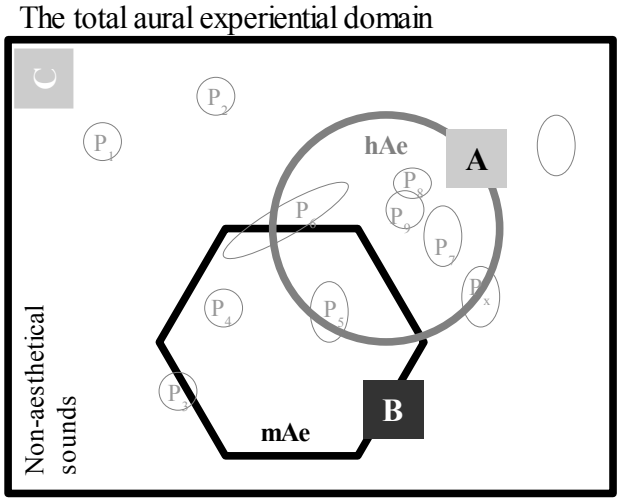
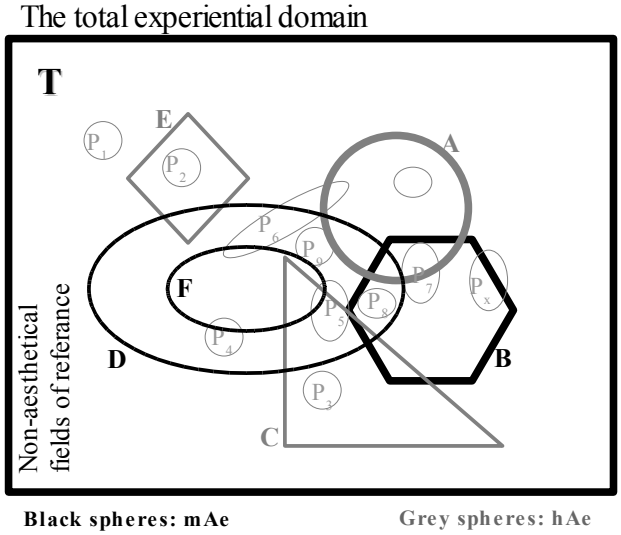


Fig.9.5.9: The total aural experiential domain : C = nAe+ (hAe ∪ mAe)

Could this open up some kind of bigger pandora's box of machine composition aesthetics? In any case it makes it possible to define certain conceptions that are describable in this theoretical conceptual apparatus. We continue with the all-encompassing model of referentialist or extra-musical meanings.
We assume like above that A and B are the sound/music-spheres of man and machine respectively. We expand them with intersecting spheres C and D denoting spatial representations/figurative arts of man and machine respectively, and finally E and F denoting literary/linguistic spheres of man and machine respectively. We may on this interpreted background further specify and denote subsets, proper subsets, intersections and unions as various activities (arts, sciences etc), various subactivities, styles and other recombinations of basically *features* and *fields* of *recombinations*.



A/B: sound; **C/D:** "spatial arts", **E/F:** literary arts [692]

{x ∈ T: P_1 (x)}: P_1 = Newtonion physics
{x ∈ T: P_2(x)}: P_2 = Moliere
{x ∈ T: P_3(x)}: P_3 = Rembrandt
{x ∈ T: P_4(x)}: P_4 = new machine sculpture style
{x ∈ T: P_5(x)}: P_5 = variant of P_4
{x ∈ T: P_6(x)}: P_6 = man-machine multimedial style
{x ∈ T: P_7(x)}: P_7 = machine style with GTTM+
{x ∈ T: P_8(x)}: P_8 = m. style of music/figurative arts
{x ∈ T: P_9(x)}: P_9 = m. style of figurative arts
{x ∈ T: P_x(x)}: P_x = Top 20-chart machine musical style OR machine-functional sounds to enhance sleep quality and alleviate upwaking process.

*Fig.9.5.11: The total **experiential** domain : T = nAe+ (hAe ∪ mAe)*

This perspective is in analogy to the spatial paradigm taken by Gärdenfors a.o., where conceptual spaces (or features of Goodman-symbols) may be compared qualitatively regardless of their experiential modal origin and dimension.
Another interesting part is that we now can describe G-expression, Goodman's most defining reference mode in the arts. G-expressions are metaphorical exemplifications, and the extensionalification of symbolic domains, that we for the sake of the argument assume to be part of our models of spheres (domains and activities), let us now specify an example of artistic expression or G-expression, that machines may be able to operate as long as their domains are specifying sufficient resolutions and ranges of features. I might for instance say that "*this* grand piano is like the cashier employee", and mean thereby to "connect" the non-existant (at the moment) feature of "out-of-tuning" of the actually tuned piano (only potentially being out of tuning) to refer metaphorically to the employees lack of concentration and habitatual behavior. In artistic or aesthetical terms, this will be an acceptable way to draw attention to very low level features from different domains (or same domains) in a symbolic relevant way. It is *not* metaphorical, since those are exemplifying *existing* features from different domains, but *G-expressive* because they metaphorically exemplify *non-existent* features from different domains. In Goodman's symbolic theory, that mode of reference pretty much "exemplifies" art as the symbolic activity it is. As a consequence, if those featural domains are explicated and specified in machine's ontological domains of reference, we may imagine ourselves not only intelligent symbol recombinations (rearrangements of symbols), but creative and constructive recombinations as well. This is as far as I can see an entrance ticket for machines to *our* aesthetical domains, as full members (in principle), and to the extensions machines may make to these domains as a result of their own evolutionary histories and emergences. If this becomes reality, machines would take part in the referentionalist modes of combining *new* and *creative* meanings from cross-experiential and inter-modal domains of experiences.

Without this premise, we may still argue that machines have potentially higher levels of intelligences and creativities in *specific and single* domains, but may not receive consciousness-related (or referentialist) acts of creativity. This view, that I called elimino-concessive above, because the right part **(fig. 9.5.8)** of creativity may have to be redefined, i.e. eliminated as their existing definitions or in other words our current terms of creativity **eliminated** (just as folk psychological terms due to eliminative materialism), while the left part (**fig. 9.5.8**) will be, as far as

we should permit us to think today, **concessed** to the human species of sapiens alone. Curiously, this is a version of the weak/strong AI debate for the musical domain. It is a weak **and** strong version of AI at the same time. It is *weak* regarding cross-experiential creativity and *strong* (and even eliminative) regarding *musical domain-specific creativity.* In informal terms, machines may well be better composers of music in the unified sphere (see above), but humans may still be the masters of the total domain of experience and their domain-transcending meanings and symbols.

On the other hand, this distinction does not say much about the level of autonomous MCs, they **may** be, as a general rule, more autonomous the more creative they are, but not out of logical necessity.

In Pearce and Wiggin's "Towards a Framework for the Evaluation of Machine Compositions", similar thoughts are expounded about what they call evaluation of Machine Compositions. It contains four components: "specifying the compositional aims, inducing a critic from a set of example musical phrases, composing music that satisfies the critic; and evaluating specific claims about the compositions in experiments using human subjects". In their discussion of the evaluation of musical creativity, they identify two levels, where the first is described as "quality" (what I called intelligence level) and the second is named literally "creativity". "Finally, the framework may be extended to evalute the musical creativity of machine composers." In their highlighting of issues, they mention the connection between composition and inferable underlying mechanisms as well as an extension of these aesthetic programmes to other creative systems ("for the generation of, for example, visual art, stories and jokes"),

Finally, there is the more practice-oriented approach of John Bower's Machine Music Aesthetics (in his "Improvising Machines Ethnographically Informed Design for Improvised Electro-Acoustic Music", 2003[693]). Bowers is concerned with improvisational MCs and their "manifold contingencies negotiated in shaping a musical practice"[694], a view that he sees in explicitly opposition to "the "methodological strictures of acousmatic composition" and the allied practices for reasoning about sound associated with Pierre Schaeffer (1966) and his followers." In contrast to our more epistemological perspective, Bowers stresses:

> I have already flirted with making a qualified aesthetic leap here. The open exhibition of the many and varied forms of interaction and engagement that can exist with such machines and materials *is* the aesthetic point of improvised electro-acoustic music...
> The image of improvised electro-acoustic music that I want to experiment with is one where these contingencies (of place, structure, technology, and the rest) are not seen as problematic obstructions to an idealised performance but are topicalised in performance itself.[695]

Much of this thinking may be derived from improvisational challenges mostly. Still, his "pattern of engagement as one of *initiation, delegation, supervision* and *intervention"* bears interesting similarities with our exposed trinity of traits in a more dynamic or action-scientific (Laske) enclothing. We will encounter such an entrenchment of machine and man in the thought experiment of **9.7** later. For now, we restrict ourselves to Bowers prophetic-sounding and concluding words:

> Rather, I am presenting electro-acoustic music as an arena where our varied relations to machines can be explored as indigenous to the music.[696]

The overdue closing of this rather far-fetching section ends with what I found, and find to be the nearest to wisdom in these matters. We let Loy conclude from *Musimathics* with the following words, that are his own last sentences of *Musimathics* as well:

> Methodological criticism, information theory, psychoacoustics, complexity theory, and other approaches discussed in this chapter are making important contributions to theoretical aesthetics that finally allow the

> dialogue about the nature of art to move beyond its fixation with Pythagorean proportionality.
> Perhaps the truest proportions in music are those that relate expectation, interest, entropy, and redundancy; perhaps the truest study of music structures requires understanding the non-linearities of our perceptual and nervous systems as well as the self-organizing principles of nature.[697]

We may well take these words in their bravest interpretation and link the whole enterprise of sound production, be it organised by humans alone, or in cooperations with machines or finally by autonomous machines alone to the so much bigger and encompassing project we label evolution or simply nature. This, as modest as it seems possible to remain, is the objective of the next section.

9.6 Aesthetics and Machine Music in a naturalistic and hence darwinistic perspective

Too much focus on the creating agent and his products (compositions) leads to the neglection of the systemic or ecological perspective in the subject. As we saw in several places **[ch2]**, many of the most "talented" composers of music history are very much a product of their interested and demanding publics. During the more recent music history[698] a minority of specialists proposed works to a majority of listeners, that then chose the best pieces/composers in ways that let a few continue and prosper and the less successful vanish and go "extinct". We are somehow back to the origin of music from **ch2**. What does that mean for our aesthetical and machine musical perspective? Finely discriminating[699] publics (critics) select certain composers out of the variating and replicating processes of artistic evolution. Having said so much already, could there be a more universal nature to this aesthetical nature? Ben-Ami Scharfstein[700] is convinced about the universality of art and writes in "Of Birds, Beasts, and Other Artists":

> Having justified the drawing of analogies between animals and humans, I go back to the analogies themselves, beginning with song. With respect to the song of birds, I should like to draw five broad analogies. The first is that birdsong and human art are both ways in which the individual self is made external or given an external form. The second is that birdsong, like human art, may come to have a value in and for itself. The third is that birdsong and human song have similar uses, to establish intimacy, to court, and to claim, warn, and war. The fourth is that birdsong, like human art, creates a relation of challenge and response, a kind of cooperative competition. The fifth and last is that birdsong, like human art, helps to create the interdependence of individuals and so to make a group cohesive.[701]

The third to fifth analogies are very much in accordance with Mithen's theory in **ch2**. The first two are disputable, but sapiens has a long record to downplay and deny other organisms' intelligence.[702] The unification of communication in 'Hmmmmm' **[2.1]** and the unification of all symbol systems by Goodman's theories **[9.4/9.6]** of symbolic communication as well as the unified approach to art by Scharfstein (cross-species and cross-cultural) may motivate our curiousities to ask the all-encompassing question as to whether all cultural phenomena are a subset of organismic or evolutionary phenomena of life? Is art and science in effect the outcome of the same laws of nature, namely the evolutionary rules named after Darwin?

Ray Kurzweil[703] hints in this direction in his description of Alife:

> The artist is thus like a deist god who creates a starting point and then unleashes a recursive process of repetitive re-creation. The artistic value of this technique should not be surprising. After all, real evolution has certainly created a myriad of beautiful forms. In fact, aesthetics itself is often considered to be rooted in the beauty of natural creation. The artificial-life approach to creating art is to imitate this *ultimate design force*.

Daniel Dennett in "Darwinizing Culture" sees culture as the shared attention and shared intelligences. Academia "creates structured networks of knowingly shared attention and mutual knowledge, so that more or less everybody is on the same page". This brings with it the competitive

and concerted effort that is defining scientific cultures. Artist cultures or networks are perhaps somehow less concerted, but at least equally prepared to compete. In “Darwin's Dangerous Idea”, Dennett uses this idea of the natural, but mindless and purposeless selection, to reconstruct the evolutionary processes that caused both musical intelligence and creativity and its newest mechanisms, composing machines. Both organisms and cultural products are *designed* by this one and only *algorithmic process,* that he calls the “universal acid” as well.[704] Such designs *emerge* from lower level order without need of “skyhooks” (helping hands from above somewhere), involving no more than the diligent working of cranes (products of strictly algorithmic processes until then, that now *amplify* the power of the existing algorithms as well). We may think of notational systems or the rule formalism of Fux's Gradus[705] as important crane technologies in musical evolution. In chapter six of “Darwin's Dangerous Idea”, Dennett likens the design building processes of evolution to the Research and Development, executed by natural selection and consisting of actual trajectories in the “Vast space of possibilities”. This search for good designs is by itself a consistent and unifying approach to the notions of creativity, originality, invention etc. The fact that we talk about a *single* encompassing space, not one for art, one for science and one for sonata forms, leaves us with a perfect continuum between organisms (and viruses) and artifacts of culture. In this grand picture of Dennett, biology as well as aesthetical spheres will be part of the engineering processes of evolution. Like 'nature' “studies” the functional mechanisms of organismic designs, their construction and operations, musicians study the functional forms of styles and works, their compositional principles and operations. Like adaptionism in nature, that can be described as the figuring out what “Mother Nature has in mind”,[706] and functions somehow like what we call 'reverse engineering',[707] so do musical composers similar research inside the scores of former composers' works in their hunt for proficient tricks and methods they can make into their own in *their* composition of stylistic traits and signatures[708].
In Dennett's entertaining “In Darwin's Wake, Where am I?”, he expounds the mechanisms behind so many peoples' wishes for skyhook-explanations in the artistic domain:

> The very idea that all the works of human genius can be understood *in the end* to be mechanistically generated products of a cascade of generate-and-test algorithms arouses deep revulsion in many otherwise quiete insightful, open-minded people.... Darwin's “strange inversion of reasoning” turns an ancient idea upside-down. The “top-down” perspective on creative intelligence supposes that it always takes a big, fancy, smart thing to create a lesser thing....
> There is no requirement in Darwin's vision that these R and D processes run everywhere and always at the same temper, with the same (in-)efficiency. Consider the unimaginably huge multi-dimensional space of all *possible* designed things – both natural and artificial. Every imaginable whale and unicorn, every automobile spaceship and robot, every poem and mathematical proof and symphony finds its place somewhere in this Design Space. If we think of design work or R and D as a sort of *lifting* in Design Space then we can see that the gradualistic, frequently back-sliding, maximally inefficient basic search process can on important occasions yield new conditions that speed up the process, permitting faster, more effective local lifting. Call any such product of earlier R and D a *crane,* and distinguish it from what Darwinism says does not happen: skyhooks. Skyhooks, like manna from heaven, would be miracles, and if we posit a skyhook anywhere in our “explanation” of creativity, we have in fact conceded defeat – ' Then a miracle occurs'.
> What, then, is a mind? The Darwinian answer is straightforward. A mind is a crane, made of cranes, made of cranes, a mechanism of not quite unimaginable complexity that can clamber through Design Space at a giddy – but not miraculously giddy – pace, thanks to all the earlier R and D, from all sources, that it exploits.

He likens Deep Blue to EMI and asks rhetorically, who beat Kasparov or who fooled experts and professors of music? It it is neither David Cope nor Murray Campbell and his people at IBM. Deep Blue, like Kasparov prunes the search trees by taking calculated risks. EMI, like Cope scan through the same Vast Space for design solutions that Mozart or Puccini had to traverse. In the sense that Cope is the designer of EMI, he may claim fathership to her (“Emmy”) solutions, but not anymore ownership or even apprehension of detail. They effectively split into different evolutionary roads, an interpretation I believe Dennett would accept.[709]

Dennett finds that the themes of art "all converge when the topic is creativity and authorship" in the genius' *own ,* its *irreducible* genius. In the end of the article Dennett thanks[710] all his mentioned "co-authors" featuring in the article, not for their actual presence, but for their ideational or "memetic" presence. Somehow like the painter Philip Guston formulated his flux between being a product, a producer and a nothingness (disappearing as an ego), just as we learned about the sense of boundary loss from the account of the distant origins of music and culture of the 'Hmmmmm' **[ch2]**:

> When I first come into the studio to work, there is this noisy crowd which follows me there ; it includes all of the important painters in history, all of my contemporaries, all the art critics, etc. As I become involved in the work, one by one, they all leave. If I'm lucky, every one of them will disappear. If I'm really lucky, I will too.

Definitions of art and culture are non-trivial affairs really. For examples does Goguen in his research about our "microscopical" qualia relate their capacity to anticipate events to the evolutionary frames of our existence:

> However, humans can anticipate, not just simple physical actions which animals have, but also the behavior of other humans. This capacity arose from evolution and natural selection to enhance survival; it rewards correct anticipation with pleasure, arouses curiosity when anticipation fails mildly, arouses doubt and uncertainty for greater failures, and arouses fear in case of significant failure in a dangerous situation. These instinctive responses are emotion, triggered by comparing anticipation with reality; this is the E (for Emotion)Hypothesis.[711]

Richard Dawkins, in his well-known "Selfish Gene"[712] likens all identifiable cultural units, such as tunes, and catch-phrases[713] to the genes of biological evolution. This cultural or memetic evolution of *memes* has according to Dawkins many parallels with the selection and replications of genes. Just as genes propagate in the gene pool, leaping from body to body, memes propagate in the meme pool, leaping from brain to brain.[714] His important "copernican revolution" is that not bodies or minds are the protagonists in this propagating population dynamics, but the selfish genes and selfish memes become the active units that *use* our bodies and brains only to replicate and diffuse *themselves* in our common bio-spheres. Is it possible that a tune, let's say McGyver's intro, is the "creative" force of culture and not our selecting and reasoning about semantic and pragmatic realities? Could it be our priggishness only that makes us believe in a Ptolomean world view of ourselves in the role of the chord-pulling free wills[715] ? The debates on the fundamental units (genes or organisms, memes or persons) in evolutionary dynamical systems are still going hot, and treat far from resolved issues, but the temptations to follow such "acid"[716] thought lines are difficult to block, once such memes have taken resort in us. Music is, in effect, a wonderful example, for meme theories. The whole music history may be re-stacked in new terms, where meme-industrial concepts will substitute the notions of forms, schools, styles and ultimately sounds etc. What is good music, after all, if effective musical memes (melodic subjects and harmonic devices) translate to "meme "A-A-A-F[717] spreads among people because "A-A-A-F" is a *good* replicator". In defence against invading memes, we are consciously and non-consciously raising our filtering weapons, to hinder non-welcomed melodic tunes e.g, to do damage in certain ways[718]. Such filter architectures are good arguments for a memetic reconception. Existing memes construct anti-meme-defenses (filters[719]) to continue a prosperous life in their unknowing and friendly hosts, namely us. Could really virus and memes be so similar? Again Daniel Dennett's words will resound in our mind for a while:

> What foundation, then, can we stand on as we struggle to keep our feet in the memestorm in which we are engulfed? If replicative might does not make right, what is to be the eternal ideal relative to which "we" will judge the truth of memes? We should note that the memes for normative concepts – for *ought* and *good* and *truth* and *beauty* are among the most entrenched denizens of our minds, and that among the memes that constitute us, they play a central role. Our existence as us, what we as thinkers are – not as what we as organisms are – is not independent for these memes...[720]

This should bring us back to our framework building of aesthetical spheres for machine composers. We found the trinity of traits to be a universal norm for music, but that makes it rather unspecified in detail. The details of any such normative theories of quality or "beauty" will have to go, for ever, notwithstanding their functioning as life buoy in storming waters of scientific revolutions:

> I submit that the meme's-eye view of what happened to the meme is quite obvious: "humanist" minds have set up a particulary aggressive set of filters against memes coming from "sociobiology", and once Dawkins was identified as a sociobiologist, this almost guaranteed rejection of whatever this interloper had to say about culture – not for good reasons, but just in a sort of immunological rejection.[721]

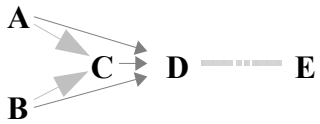
The only thing that will stay with us is the continual revising of a *naturalized aesthetics*, that in coalition with naturalistic cognitive sciences will contribute with knowledge about *us* and *"our" memes*, hence *naturalized epistemologies* as well. Is there nothing more eternal or stable in all this? Our answer must be a resounding *no,* just as "Neurath's boat"[722] is a perfect picture (or better *meme)* to capture the situation in art and its struggling for firm fundaments under the shifting disguises of aesthetical theories and their norms. A curious afterthought may uncover some of the surprising analogies (and possibly compatibilities) between Goodman's exemplifications and Meyers antecedent-consequent-relations on the side and the mechanical natured memetic industries of our communicational identities on the other side. Having compared the musical worlds with the biological world of evolution, many mechanisms of both "worlds" display the *fundamental tripartion* into = (≈ *replication*), ≈ (≈ *variation*), and ≠ (≈ *mutation*) in their respective domains, **O**. I will now add a very much speculative, and therefore tentative juxtaposition of some areas and subareas of the subjects (music, machines and evolution) that occupied our *"vectorial minds"* in the percourse of this thesis:

Context	**=**	**≈**	**≠**	**O**
Evolution	Replication	Variation	Mutation[723]	Evolution /selection
Design	Copy	R&D	Cranes	Engineering
MC aesthetics	Repetition	Variations	Contrast	Unity/work/style
MC systems C-4	Assisting	Guiding	Autonomous	Machine Aesthetics
MC-dimension C-1	Sequenced/transf.	Generative/grammatical	Complex rules	Machine Aesthetics
Creativity	Expectation	Interpretation	Reconception	Nature
Adaptation (CAS[724]) Complex adaptive s.	Internal models	Tags	Building blocks (Aggregation)	Nature
Physics/Prigogine	Equilibrium	Steady State	Complex systems	Systems
Language	Syntax	Semantics	(Radical?) Pragmatics	Bio-domain
"Machine Nature"[725]	Bio-inspiration	Adaptation	Complexity	
AI (Sipper[726])	Hardware	Evolware	Bioware	
Algorithms	Loop	Transformation	Recombination	computational
Genetical algorithms	Population	Crossbreeding	Fitness function	Evolutionary programming
CA (cellular aut.) Wolfram	Class 1/2a Constant/Repeats	Class 2a/3 Regular Fractal/Seething[727]	Class 4 Complex	Universal Matrix
Homeostasis	Strong Homeostasis	Weak Homeostasis	Structural Home	Ecological unit
Psychology[728]	Behaviorism	Cognitivism	Constructivism	Humans
Music cycle	Performer	Improviser	Composer	Music community

Context	**=**	**≈**	**≠**	**O**
Sonata Form	Expositions/ Recapitulation	Developments/ Transitions/Coda	Second subject Distant modulations	Sonata movement/ sonata/sonata circle
Fux counterpoint	Species 1-4	Species 5 – florid	"Bach"-counterpoint[729]	Fugue form
Pop songs[730] *(Verse -Chorus)*	Verse	Text [730]	Chorus (B) evt. Bridge(C)	ABACAB or AABA
Frankensteinian methods (evol.progr.)	Development	Diversiveness	Complexity/Chaos	Artistic computing
Adaptation	Learning	Intelligence	Creativity	
Set theory	$=, \cap, \subset$	$\cup$	$\in$ [731], $\notin, \not\subset$	domain
Logic	AND	OR	NOT[732]	
?				

There could be endless threads of memetic or symbolic combinations and recombinations in continuation of the implicit claims deriving from this heterogenous collocation[733], but we have to leave the baton to the more prosaic succeding section after a short recapitulation of what we may have achieved in the end.

Recapitulation of 9.3 to 9.6



We started out with answering to two questions as to whether machines in principle may be creative and conscious (in the sense of 'deep-' and 'infra-experiental'). We found arguments to naturalize both domains of human mental powers, and opened thereby the gates to machines and their becoming com-panions to us **[A/9.3]**. Then, we inspected existing theories (selected from the impossible many alternatives) and found expectations and meaning (EM), as well cross-referential symbolizing and domain-crossover meaning in explicationalistic symbol systems to be both *thinkable* and *formulizable* (LA) **[B/9.4]**. As a consequence, we *cannot deduct* formally any specific aesthetical conclusions for Machine Composition, *but* we may by *induction-like reasoning* find a base for arguing in favor of the *generality* (and hence at least its truth-likeness) of the *trinity of traits* (repetition, variation, contrast) and their foundation-*like* description or symptoms[734] for aesthetical machine composition. What we thereby want to *exclude*, are the sound-complexes that are too far from our expectational capacities[735]. What we want to *include*, though, are the establishments of new types and paradigms of styles that machines will propose to us and humans might (in the end) find understandable (and interesting/pleasing etc.), once they have *adapted* to these machine aesthetical spheres of understandability. **[C/9.4]**
Once we have opened, what I like to call the Pandora's box to MC **[C]** , we looked at several possible interpretations of the relationships between the aesthetical spheres of humans *(hAe)* and machines *(mAe)*. I proposed the *elimino-concessive* view **[D]** , which in simple terms states that along the creativity-dimension of art (see figure above), machines will *outdo* humans in respect to depth, breadth and consistency, while at the same time *conceding* to the human aesthetical sphere a certain autonomy regarding the dimensions of consciousness and their capacities to integrate meaning from overarching domains (cross-referential and multi-modal). In this sense we may propose that humans *understand* (listening) the integrated union of aesthetical spheres (hAe $\cup$ mAe), but not necessarily construct (compose) works in every part of it. In the opposite direction, machines will create or construct works in the mAe mostly, but will *not necessarily understand* the full meanings of complex symbol systems (Goodman), that are applied in the human aesthetical

sphere (hAe).**[D]**
Finally, we widened the domain of investigations and compared the "aesthetical dynamics" of symbols and their understanding to the units of evolutionary play **[E].** We found striking analogies between the forces of natural evolution and artistical evolution. A speculative table with equivalences to these three main motors (=,≈,≠) of both physical, cultural and musical spectacles are laid out with a modest attitude, inviting further speculations on these memes.

But we leave now this long theoretical part and change the "style" of presentation. In Greek mythology the muses brought inspiration and creativity to bare mortal humans. In the following, muses fused with machines have become *autonomous music machines* [736] evolving with each other, and in collegium with human players they do enter new stages of *human-machine artistic ecologies*. Marvin Minsky declared that machines some day will be so far superior to us that we will be lucky if they keep us around as household pets. In our case the question is whether they want to include us in their creational and recreational sonic activities. Will they find our playing interesting enough? And will our concepts relating to the creative domain be eliminated (elimino-concessive view, **9.5**) in their existing forms?

Is this, once again, one of history's many ridiculous LaMettrian phantasies or could this become a part of humanities' future realities?

9.7 Futuristic thought experiment: How could Machine Music change urban landscapes?

In 1624, English philospher and essayist, Francis Bacon described a scientific utopia in the *New Atlantis*. He predicted:

> "we have sound houses, where we practice and demonstrate all sounds, and their generation. We have harmonies which you have not, of quarter-sounds, and less slides of sounds"

Bacon's futuristic and utopian phantasy will now be "recombined" with pieces from the LaMettriean tradition (Man as Machine). The following conversation is taking place in a social club somewhere in the Underground Free Zones (UFZ) under the Oslo bay, near the Opera building. The year is 2056 and we listen to a couple of friends talking about music.

Hovard: What did you do, yesterday, Terje?
Terje: Nothing, just relaxing, ... I guess I was digesting this TT [TotalText[1]] about some moral issues, interesting thoughts really, ..
Hovard: What kind of morals, did... (*sees Ola coming and waits for him to sit down with them*).
Ola: Hi, there, what's up?
T: Just do some talking here. Hovard thought to play with the *muses*[2] ? What about you? Is your instrument with you?
O: Always with me, you never know when you want to play. Yesterday I played with the "Slottsgate-gang". We were around 10 people and lots of things happened. There was this guy for instance, that played this perfect tuba-sound, fast and precise, ... don't know how he did it, incredible really.
T: Are you shure, not one of the muses manipulated it, just for fun? Sometimes they are slowly starting to filter out all the 'subopts' ["sub-optimal errors"] without letting you know what happens. One time, I started having this groove and was high flying [happy], thinking I was becoming better than myself, wow.., until I realized that it had to be one of the muses making fun of me.
H: Really? sounds cool to me. It's called *fuse playing*, no? Why do you think he picked you, and not another one. Or were you the only one there?
T: It was one of the 'funk-hoppers' at the SAS-hotel, ... no, there were quite some people playing... May be he just

1 Total texts are multi-modal information entities that are read in integral ways of seeing 3D pics, 3D sounds and hyperanimated texts in integration. They are stimulating totally absorbing hyper-focus states, using a couple of immersion-techniques unknow to us today.
2 Muses (gr.): origins of creativity. Now stands for **Machine Unit sound evolution** (autonomous Machine Music)

didn't like me?

O: Or...he liked you a lot, and wanted exactly your attention. I like these guys myself. But it feels terrible when muses mimic the worst slips [playing mistake] of yours and use them in ironical ways to amplify them in the most embarassing places...They are the ones with these flashy colors, which one was it?

T: I'm not shure, I thought it was the blue one, at the time, but again, how can we know for sure. Do you hear what happens in Bogstadveien? They have received some muse-guy from Buenos Aires, for a while, some cultural exchange by the township I suppose, he plays these terribly complex latin rhythms, how can people in Argentina play along, or even dance to it?

H: Sounds funny, think I want to listen to those sometimes, did you make a *strape* [structure extracted recording[3]] of it?

T: No, unfortunately not, you never know what will happen to him, with the other muses around him playing all those *comfort stuff*. He might get out of training, or worse, take up too much of that Bogstad-style. Why could't they block learning cycling in such interesting muses, then we wouldn't loose those exotic things to social pressures [adaptions]?

H: Probably, they could just increase the ego-level of him, that would make him an alpha-muse that doesn't change easily and let the others learn from him instead. Did you hear about the next town contest?...No? it's the 3.Cope competition[4] and from what I read both Amsterdam and Bergen are going to send their "killer muses", all will compete with some of the best Oslo-muses. Wonder who will be picked to represent our town, could it be the 'Grorud-busters'?

T: Sounds awesome, ... Grorud-busters, they are f.. crazy, heard them some weeks ago, the style they play in, at the moment, is the most dense and strange I have heard, and still, they know how to do it, we get the message, it's understandable. Heard of that 50 years old Zorn music? Speedy and brutal, that's how *they* play, just more updated. Some people try to *tune in* [play with them], but most of them go *over the radar* [[5]]. These people [muses!] are not really trying to play with us, they are more like trying to exhaust each other.

O: Oh, I just heard about them, where do they play? they are not street-players?

T: No, they are invited to clubs and places. Their learning algorithms are hyper-evolutionary, they kind of jump over whole stages, cutting corners of learning, and still, we listeners follow them, they are kind of logical, after all.

H: Well, that's not fully true, I heard people that listened to them, and hated them, they called them snobs and other bad names, but then..., they probably just didn't get it, or ... they don't play themselves.

O: Ha! Who is it that is *not* playing today? I don't know anybody that does not cool down with playing music.

T: There was this time actually, when people didn't play, did you know? My great-grandfather, he is 107 now, told me that in his youth people did *not* play much music. They only listened to some music, and you know what, they played mostly the same pieces over and over....

H: Ha, that must be uttermost boring, why would they do that, had they no *variators*, *contrastors*[6], at least?

T: Well, he thinks it's strange too. Now he plays some of these virtual strings, *Rubber-go's*[737]. Well, he told my sister and me a lots of things about, say between 1960 to 2010. People then actually paid quite some money for the music that everyone else bought. And it gets worse, they chose their music from a list of only 20 pieces, played on those air-transmitting-stations, radios or something, all playing the same music, and the story doesn't end here, they bought them in some kind of non-writable storage medium, so that they couldn't even change it, think about *that*?

O: Strange people.Why buy music at all? And why did they all buy the same music to play it over and over?

T: Because, that's the best part of it, they wanted to be like everyone else. If other people bought this or that song, they wanted them too, may be they thought to be cooler if they were more alike, how do I know?
And they *had* to pay, since some of these multinationals declared absolute rights, to *own* this music so to speak, copyright was the name they used, so that nobody should be allowed to use that music without paying for it.

H: I have a story too, *my* great-grandfather was apparently playing this odd gigantic instrument, called pipes or sth ...

O: You mean organ, I suppose!

H: .... well whatever, they paid him for playing on this thing in those big churches, you know .. but, very few people were actually showing up... it must have been somehow like playing at home.

O: Why should people pay somebody for playing an instrument, at all?

H: Many people, gran'pa told us, were not playing, and some not even listening, they were *a-musical*, with complete empty learning cycle histories [**ch2**]. Somehow like *non-musical robots*.

O: My grandfather was like that, I was told, he tried to play a hammered piano as if it was a keyboard.

T: And my grandfather hated any sound altogether, he thought it to be most *brumm* [cozy] when it was total silence, no music, no playing at all, what about that?

---

3 Structural recordings are intelligently parsed essential patterns and structures, coming out of Machine Listening research.

4 Cope-competition, after David Cope (see ch5) uses rules similar to the prisoner's dilemma, both groups and individuals compete, and it is the algorithm that balances group and individual interest most effectively that turns out as winner.

5 expr: is cancelled by muses because under minimum accepted quality

6 Sound-transforming intelligent agents that allow the user to evolve patterns in related or new directions.

H: Yes, people must have been kind of strange, another thing I heard from gran'pa, was that they had this mystical belief in what they called "creativity", they had another name for it, "originality", meaning "being one self", Apparently many people didn't make their music themselves, and thought of this *composer-thing* as something really special.
O: I know, they believed in souls, too, or was that earlier then?
H: Yes, may be some odd people believed in vodoo stuff too etc, but we must understand that they hadn't the *tech's* and *the's* [technologies and theories] for creating what we learned in school, such as the hierarchic c-laws [*creationisms*[7]], of *rep-var-con* and so on. *Their* thinking was just so much different from ours, we really don"t know how they thought, do we?
H: They still had these mechanical instruments in wood, like Mozart and those classics. They liked these *materials* in themselves. No nano-tech-industry then, you know. Now that we have all those cool "Active instruments"[8] to play with, we can't imagine how it must have been in those "piano-days".
O: Oh yes, they couldn't play quarter tones for example, and scales were fixed, you couldn't change them in the middle of tune or piece. But what about this Vincento or so with this interesting keyboard?
H: I think you mean Nichola Vicentino[738], the inventor of the archcembalo with 31 steps to an octave, that was in the 1600s. And actually, there was this 'Musurgia Universalis', a mechanical device by Athanasius Kirchner[739], 1600s as well, that made music using numbers and arithmetics, the hard way really. He called it 'Arca Musarithmica', I think.
O: So this must have been an early *active instrument*, then? How do you know all these exotica?
H: Must have read it in this TT about the "Phytagorean and LaMettriean temptations", some old stuff. What instruments are you playing at the moment, by the way?
O: Me? Always the Jammer-horn? Don't change much the controllers, it's all about the software, anyway.
H: What version do you have?
O: Well, if you ask, I'm the proud owner of the newest B370-algositions. They help me in exercising and playing new stuff, just whatever. It comes straight from the Mumbay area [high-tech-zone]. That's where some *annoying* [=interesting] things are made now. Well and then certainly all that evolutionary and hyperlearning stuff coming from Hokiang and Sungkiang regions [China]. Expensive yes, but they are the most *neo-tech's* [creative technology] at the moment.
H: Lucky you, playing a 350+, that is actually doing what people 50 years ago called composing and they did most of it with pencils and ears.
T: But, Hovard said no-one really made music; that they just listened to predictable stuff.
H: Some few people were apparently specialized on making music for others, and got paid for it.
O: Would *you* like to go back in time, if it were possible, just to look around how they lived music at those days?
H: Not me.
T: Me neither, let's go instead and play with the muses in the sound house over the corridor, what's its name again?
O: That one? is the 'Bacon-house'. They serve good coffe too. Let's plug in our wirelesses and play instead of talking strange things....

9.8 Instead of an epilogue: Between LaMettriean temptations and Phytagorean dreams

We now take the time to collect some loose threads or ends before turning the last page of this thesis. If we were able to see the unity of creative forces in both evolution and musical cultures, we may not be too surprised to find another one in the present letters of this sentence. Writing a thesis is really very much like composing a musical piece. It is also the proposal of a replicated (repeating someone's ideas), variated (reformulate someone's arguments) and contrasted ensemble or work of ideas and consequences, meticulously put together in an as logically consistent and understandable way as possible. The planning of the chapters is the algorithm or compositional scheme, the filling in with content and arguments is the stylistic and craftly work of collective wisdom, and the "original ideas" nothing more than connections and recombinations of *hitherto non-combined* elements (memes) in new and fresh ways confined only by our common intellectual culture. The author's continual and incremental reading of manuscripts and error-prone versions of this thesis are under the control of the critics inside the author himself (and some of his guiding friends acting as intelligent agents[740]), shaping it into the final draft that will be inspected, critizised and evaluated by its highest judges, the censoring collegium, classifying the whole project and work into a metrical

7 Changed meaning in the 2030's
8 This name is taken from Chapel, see ch7.

scale between some selected letters. This whole intertwining of influences, environments and active agents is the vital nature of artistic as well as scientific cultures.

What was then, after all, the cohesive meme of this enterprise? We started out with the non-exactitude of 'analysis of algorithms'[741] **[1.3]**, and found the nature of sudoku *and* musical compositional "games" to be more in family with *puzzles*, than clear-cut calculations. We discovered that creativity, as much as our own organismic puzzles, are the consequences and contingencies of natural selection ("universal acid"), just as engineering and its products of humanity are the results of those universal design algorithms that unify the spheres of non-human and human nature. As a consequence, the nature of machines will have to be integrated into our own puzzles and compositions, just as persian carpets integrate a multitude of interesting patterns. The fact that patterns in symphony, under the organizing principles of our trinity of traits (repetition, variation and contrast) explain the expectational nature of musical sound carpetry (Kivy calls music "sonic wallpapers"), makes it seducive to dreams about *universal* harmony of patterns in all kinds of domains and depths of existence. But such Phytagorean dreams must be contained, just as much as their diametrical twins of LaMettrian temptations that emphasize the similarities between non-carbon machines and neuronal or non-siliconic machines. The greedy reductionists that uncritically equate us with computing matter will be as far from "the truth" than the arrogant holists that point to abysses between composing spirits and machines. There is no principly undividable nature of consiousness, that will make it logically impossible for machines to enter our aesthetical domains and even expand them for and together with us. We look at two final enlightening observations:

> Phenomenologists believe the experiential unity to be non-divisible (non-separable experiential dimensions, **2.5, 9.5**) We may permit us to disagree and make the following argument: people that use handsfree equipment to interface with their mobiles, look to us *as* they *are w*hole-heartedly speaking to their conversational partners. We may simply say, they *are,* with partners that stay another place altogether. They share the sounds of speech through bits of telecommunicating. They do not see each others body language or facial expressions. And still, they pretty much feel in communicational contact with the other, sharing not place but only time. This "experiment", repeated millions of times every day, proves that machines, too, will be able to understand and communicate patterns of genuine form *and* meaning, and therefore are genuine composers on our level. Phenomenology holds beliefs in the *untouchableness* of *wholes* that is a form of Phytagorean dream, one we must chase away to continue our evolutionary and "civilizationary" paths of humanity.

> Greedy reductionists, on the other hand, like we may find in Wolfram's theories of universal algorithms of cellular automata (110[742]), are telling us that this world, in all its shades and colors, is constructed out of the tiniest elements, that are not even part of our physicalist systems, and still they are thought to be responsible for all the "temples of reality", defined in such abstract terms as cells, grids, and matrixes. We will ask them as well, to refrain from such imperialist and somehow seemingly obsessive-compulsory propensities towards explaining the all with the nothing.

The middle ground that is left to us between these dreams and temptations are the engineering fields of human constructions or artifacts of all types, created by our designing faculties in interplay with our environmental partners and pressures. The tools of Machine Composition, exposed in **ch3** and **ch6**, are in such a perspective higher level agents themselves, in search for better platforms that eventually will come out with ever expanding complexities of recombinations satisfying our aesthetical curiosities and needs. The integration of Lisp and MAX into *maxlisp* is therefore only a very small footnote in the Darwinian play of machine composition. Combining forces, drawing on the best of all tools **[ch3, ch6]**, theories and methods **[ch4, ch5]**, and finally the astounding new paradigms of learning and evoluting technologies of Machine learning will together contribute to a MC evolution, within which we ultimately may find ourselves circumvented by armies of

competing and creative machines that may or may not include us in their agendas. This is the utopian phantasy of **9.7**, and we may well guard ourselves from excessive enthusiasm in these matters, since enthusiasm means being inspired by Gods; Gods which existence we want to deny, since our re-founded belief in natural causes and origins of both humans *and* music **[2.1]** may help us to abstain from such thoughts.

We conclude these epilogue-like afterthoughts by recalling some moments of the life of one of our presumed anti-heroes (Phytagoras' life is already well known to many): *Julien Offrey de la Mettrie*. His contribution to the history of philosophy is based on his book "Man a machine", and we should remind ourselves that this early radical materialist was well ahead of his time in many respects. While Descartes wanted his contemporaries to believe that animals were just mechanist machines, while human minds and souls were not, the couragious Julien discovered under his doctoral duties that body fluids and organs were much too alike the ones we find in animals to justify such clear boundaries between body and mind. Well, Julien was right in that claim, but failed miserably when he further deducted that Descartes' machines (animals) had to include ourselves *as machines* as a matter of consequence. Today, we are more inclined towards lesser divisional thinking in these living matters, but nevertheless many still want to believe that non-carbon entities (computing or "*com-put-it-in*" devices) are so endlessly different from the non-silicon entities (composing devices = composers) from among ourselves. This little grain of sand in the history of ideas should be a warning light, that fallacies and arrogances in thinking are not confined to our ancestors alone. The lesson, as I want to believe I have contributed to, is rather to be *generous on behalf of machines* and their accomplishments and to enter the communal spaces of inter-creative activities **[ch7, 9.5]** with open eyes and ears, to participate in the converging aesthetical spheres, something that Bowers in **9.5** called the "manifold contingencies negotiated in shaping a musical practice" between machine and human composers. Julien's radical books and his own presence became banned in both France and The Netherlands, before he found his new home under the protection of the "enlightenment-ed"[743] 'Friedrich der Große'. In one of his last deeds, Julien saved a life of one of his benevolent patients, that in his "prolonged life" decided to honour his saviour (Julien) with an exalted party. Julien, that preached gluttony among all the other epicurean pleasures, overstated those pleasure as well, and managed to die following the voluptious sensualities of an exclusive paté. The moral of this, as I want make you to believe in, is that the gardener's delights of mechanical models, composed of water fountains and moving parts of quasi-robots, very much in vogue in Julien's period (especially among the rich), was a bad guide to revolutionary machine ideas. Therefore we *should refrain* from such LaMettrian temptations (Machine **as** man) and their contemporary versions of 'Computer music' and other overstating Machine Compositional philosophies.

Lucky us then, that the voluptious sensualities of sounds don't kill their recepients, unlike excessive food, and that unlike the ever so exquisite qualities of patés that killed LaMettrie, due to his surrendering to his unlimited temptations, we, on the other hand, may consume the *endless patterns* and powers of moving air molecules (sound!) in whatever quantities and of whatever origin (machines or man) thay may come to us. As an aesthetical advice, we may therefore unionize more with Julien and his life-indulging proposals than with the Phytagorean demands for balance and harmony, exemplified in Aquina's eating postulates, that regulated *decent* eating manners with the following directives: eat not too *much*, not too *expensive*, not too *little*, not too *slow* and certainly not too *quick.*[744]

This entire excursion of Machine Composition might therefore be adequately reverberated by studying the example, as well as the *smile* of Julien Offrey de la Mettrie and his explicit and implicit statements in his life:[745]

genius: *ex nilii ? Ex dei ? ex singoli ?* ***ex evolutus!***
Julien Offrey de la Mettrie (1709-1751)[746]

Summary: Machine Composition – Between AI and Music

This paper consists of three complementary parts. First, there are the fundamental problems and challenges about the foundations [1,2,9] of Machine Composition (MC), especially in relation to human composition. Second, there is the background of existing theory and experimental results from main areas of Machine Compositional Research and their constructions of prototypes [4,5,7]. And finally, the most practical part introduces two important environments or tools [3,6,8] for the construction of machine compositional systems.

In the **first** chapter, a Sudoku algorithm is proposed and put into perspective with regard to musical and computational domains as a model for algorithmic as well as compositional activities. Many similarities are found to exist between the creative activities of computational problem solving and com-"positioning" in music. In the discussion of Bach, Bruckner and Debussy, interpretations of rule system design and expansion, instead of the conventional idea of the artist's "rule-breaking" are proposed as a better model for creativity. Recent computational trends attempt to "soften" programming cultures. A distinction between hard and soft algorithmic activities is proposed that accounts for both the similarities and differences between both activities. In chapter **two**, the origin of music is placed in the continuum of natural evolution, and this underlines the vital importance of musical learning and social culture. A musical learning cycle is employed to account for both prehistoric music learning and machine learning, and opens up a unified frame for man and machine. The objects that represent the parts of reality in computational as well as musicological domains are discussed and problematized. The many machine/AI paradigms and music systems choose radically different approaches to represent information and sound. The chapter concludes with the proposal of a procedural epistemology, based on pragmatic and empiricist epistemologies, both naturalizing and naturalized themselves, as exemplified in the ontic engineering and design for Machine Composition.

In chapter **four**, **five** and **seven**, the various paradigms of Machine Composition and their AI techniques are exhibited and discussed. The improvisational system of Cypher (Rowe) is a typical MC system where many levels and modules of knowledge interrelate to produce a responsive and flexible player paradigm. Cypher is *informed* by theory and knowledge from the musical domain. In contrast, EMI (Cope), staged in chapter 5, is a typical inductive learning system that extracts style features (signatures) from a sample collection of known composer styles. EMI's Turing-test-like "Game", where many people are failing to distinguish between machine and man creation, is discussed. In chapter 7, many of the more recent trends of MC are investigated. Especially mathematical and biologically oriented research is given attention. Music generated from fractals or chaos and music derived from evolutionary principles and adapted to synthetic environments are some of the newest methodologies. Todd and Werner's Frankensteinian methods for evolutionary composition are concluding the theoretical part of this paper.

In chapter **three** and **six**, MAX and Lisp/CommonMusic are introduced and discussed. MAX's graphical and eclectic user interface is contrasted with the very 'general purpose' programming language of CommonLisp. Both approaches have significant relative advantages and disadvantages and seem to work even better in integration. In chapter **eight, maxlisp,** the embedded version of CLisp for Max/MSP is examined**.** Using maxlisp, a user may choose to build MAX-patches in various styles of MAX, supported by lisp-objects (maxlisp-objects) doing the rigorous computational tasks. Arguments in favor of integration and hybrid methodologies are endorsed.

Finally, chapter **nine,** sums up some of the encountered notorious problems in and around MC. Questions concerning the classification of MC, questions of sceptical nature in general, interpretations about creativity and originality, as well as philosophical theories about music contribute to the proposal of a tentative framework of a machine compositional aesthetics, compatible with the evolutionary and Darwinistic theories about nature and culture alike.

1 Snow wrote a book in the 1950's about the overwhelming contrast between two kinds of scientific paradigms, also called the soft and the hard sciences
2 Leonardo da Vinci and his contemporaries would probably disagree with common sense of today in this matter
3 Algorithm is often taken as a primitive concept that can't be defined precisely. In RS1996 algorithms remain defined only in their analytical use. Robert Sedgewick, *Analysis of Algorihtms*, 1996.
4 Still,,robot muscle and sensor technology catch complex phenomena by reducing energy events and pictures to numeric representations for algorithmic manipulation. Connectionist or massively distributed approaches are exceptions to this rule, to some degree. Those will be treated later.
5 Examples are shrinking areas of "wild nature" or ecosystems with distributed control contrasted to human exploitation-based complex systems (rain forest, jungles), reducing of commons land, and especially the expanding urban or civilizing trends.
6 TM are abstract machines or basic symbol-manipulating devices that can emulate the logic of any computer type that exist or could be constructed. Theoretical computer science, algorithm analysis and complexity theory study abstract properties using TM for thought experiments about the limits of mechanical computation.
7 D, P, NP, NP-complete denote abstract or formal properties of algorithms in respect to Turing-Machine computers.
8 not considering general plans or pre-algorithmic sketches or ideas.
9 There are analogies in machine learning ; see more in ch5.
10 Robert Sedgewick, *Analysis of Algorihtms*, 1996, 25.
11 Ibid. 29.
12 Natural number systems have infinite domains that are defined recursively. There is always an element bigger than x, namely x+1.
13 Logicians and mathematicians try to prove some formal relations between these classes of problems, as e.g. P≠ NP. Similar problems of tractability related to infinitism and degrees of exponentiality are formulated for PSPACE problems that classify problems relative to the polynomial or exponential sizes of space needed in relation to the problem size. Many of these relations are hard to prove formally but are suspect to non-formal or intuitiv suspicions.
14 Grids may be filled with any types of values or combination of types: e.g. letters for grids of size 26*26 or pictures of fruits and so on. In other words, Sudoku has no inherent relation to mathematical number objects, they are just some convenient implementation of a puzzle.
15 If there are 4 empty cells of a structure (s), e.g. in a row, we can attribute three of these values in pairs to three of these empty cells, the fourth non-used value can be filled in in the last empty cell directly. We call this the reduction (or deletion from a logical argument) from four candidates to one because of these pair-placed constraints.
16 C_x : 14 means that cell x is a candidate cell for 1 and 4. If C_y :41, and 4 and 1 must be in either of these cells (i.e. cells submitting to constraints), then 4 and 1 cannot be in *other* cells.
17 This is somehow like seeing a house in the fog, with some easy spots that define the house instead of seeing the whole structure.
18 "Seemingly" means here that the unproductiveness is not demonstrated, nor proved, but indirectly assumed or deduced from assumptions about higher level properties and relations. Somehow like IQ-tests will use effective test tasks that are assumed to represent not only themselves, but a wide range of abstract problems altogether.
19 Equivalent to the rules of clever transformations in symbolic math, to shorten the path to the solution of a mathematical problem.
20 This will be the subject of further reflections in ch 9.
21 Donald E. Knuth calls his algorithm for Sudoku 'dancing links' that jump seemingly chaotic between strategies. (www.jalat.com/blogs/lisp?id=4)
22 See http://www.puzzle.ro/en/sudoku_mathematics.htm
23 Kostas Terzidis uses an algorithmic perspective to architecture (Algorithmic Architecture, 2006; Architectural Press)
24 Some so-called aleatoric compositional ideas claim that random processes create unintentional structures that may be recognized by listeners (see discussion of 4'33 by John Cage later). This may or or not be the case, but does in any case only transfer the creative role of pattern induction (see ch5) to the listener, in the same time reducing or even emptying the creative role of the "composer" himself.
25 A sextatonic is here defined as the equivalent to the pentatonic scale with 5 whole tone steps. A sextatonic is a chromatic sequence of six, hence with 5 semi-tone steps.
26 Strict 12-tone scale operates under the premise that every tone has "equal rights" and should be applied equally often and significant.
27 No voice (row) can have repetitions of tones and no chord (column) is aloud to double tones (repetitions) neither.
28 Soprano 1 and 2, alto, tenor, bass 1 and 2. Nothing special in this attribution, it is occasionally used in orchestra pieces with large choirs.
29 Dewey called creative problems 'indeterminate situations'.
30 Margaret L. Wilkins, *Creative Music Composition,* (Routledge, 2006),2.
"The term "free composition" implies creative work that is not stylistically based within historical eras. In practice, it has come to mean composing within a twentieth- and twenty-first century Western European style, open to individual interpretation."
31 See also civilizational thinking above.
32 Different programming languages and paradigms work at different levels of detail specified by the programmer that

results in consequences for ease of use and performance (or trade-off between time used in programming (programmers time) and time used in execution (computer time).

33 Modern musicology does indeed show a growing interest for the perception of compositions and the cultural conditions of music composers and styles. Still, these sociological and historical perspectives are often more practiced as a minority direction in musicology. The primacy of the composer and his works seems still valid.

34 Polyphony is seen as the sucessor and logical consequence of homophony and heterophony.

35 Interestingly, at this level of abstraction (a still higher level will be the subject of discussion in the last chapter), other arts like figurative expressions in two and three-dimensional space (painting, sculpture) and even the literary arts might be examples fitting into descriptions under these principles as well. If so, music is not so special than many aestheticians have presumed. This line of debate must be excluded out of reasons of limited time and space.

36 This is possibly a metaphorical comparison. Still, counterpointal progressions behave not too unlike logical inferences, and are captured in generative grammars (GTTM, below). This will be discussed in ch 9 more thoroughly.

37 Margaret L. Wilkins, *Creative Music Composition,* (Routledge, 2006),87.

38 Frescobaldi studied intensively Palestrina's state-of-the-art style, Bach studied equally intensively the works of Frescobaldi and numerous major composer of the 18.c studied Bach's composition as perfect solutions to counterpoint in any case.

39 The majority of Scarlatti's sonatas have two double symmetries. Tonalities are mirrored about a central point (double bar), and material repeats after the double bar. It fits well into a balanced repetition-variation-contrast analysis.

40 Haydn is often believed to be the originator or this form, but this is a matter of dispute. There are many precursors of this forming process, as there is usually with any other general form as well.

41 Griffiths, Paul, *The string quartet,*Thames and Hudson, 1983, 12.

42 A recurrent observation in this thesis: the integration of aspects or musical dimensions. In Ch8 some Lisp functions demonstrate a couple of techniques in a quasi-sonata-form generating function.

43 Maurer, John A., *A brief History of Algorithmic Composition,* 1999.

44 Margaret L. Wilkins, *Creative Music Composition,* 2006, 28.

45 Ibid., 42.

46 Debussy countered this label in a letter of 1908: “... what the imbeciles call 'impressionism' is a term which is as poorly used as possible, particularly by art critics”.(wikipedia:Debussy)

47 The background for this attribution may lie in some of the vogues of his days:
1. an infatuation in exotic cultures (that is found in Debussy's interest and use of javanese gamelan impressions)
2. the philosophical trends of phenomenology
3. the physical investigations into sensory qualities by Mach a.o.

48 Howat, Roy, *Debussy, Ravel, Bartók: Towards some new concepts of form,* Music&Letters, Vol.58, No.3 (pp.285-293)

49 Ibid., 291.

50 Debussy exploited dissonances without any formal resolution. Irregular and fragmented forms...Fluidity of Rhythm and color.

51 Howat, Roy, *Debussy, Ravel, Bartók,* 292-3.

52 Among the many as numerological described composers are Bach and Bartok.

53 Walter, Bruno, “Bruckner and Mahler”, *Chord and Dischord.*

54 Prokofiev, Britten and Shostakovich revived parts of sonata form, but the classical period might be concluded with Bruckner and Mahler.

55 This was actually a contract he accepted as condition to being instructed by his teacher of composition. Some believe this to be one of the reasons for Bruckner's “genius”, a repressed and then liberated creative mind.

56 Conductors producing Bruckner's works.

57 The conceptual allegiance with architectual “works” or constructs, underlines the commonalities in the con-structed and com-posed nature of such entities.

58 Bruckners endless refining of his compositions has been called revision-itis by critics. But may also be compared to this authors headaches as to whether ANN should be introduced in the first chapter or second, both alternatives bringing positive and negative consequences (dilemma). Bruckner may have, in this sense, speculated whether for example subject B' should be occur as preannouncement in the transition or not.

59 Bruckner came from a tiny little place in the high-alps, and carried this background with him for all his life.

60 Languages like Lisp, Simula, Smalltalk and Prolog had a hard time to penetrate the programming culture. More difficult even was the introduction of different computing paradigms, such as the connection machines (neuron-based, genetic or other very nonstandard models of computing).

61 Composers will often engage in discussions and debates about methods and style; their works start often out from very sketchy thoughts; checking of the auditory effects of constructional modules is done by playing it on a piano and the acceptance of daring innovation is part of the compositional modus; still, respect for tradition and continuance has, at least before the second half of 20c. been a obvious obligation to composers.

62 Feyerabend Paul, *Against Method,* 1975.

63 Newell,A. And S.K.Card: The prospects for psychological science in human-computer interaction (1985).

64 Flanagin and Holloway, *Patterns of creativity and Composition in Software Development, Music and Film,* 2002.

"...it is imperative that we refrain from indulging in the politics of defining what programmers do as either art or science and focus on the "compositional" and "creative" aspects inherent to the process... In fact, art and science are much the same."

65 It is difficult to study the musical composition *process* in works of Beethoven, Mozart and Schubert since they often covered their trails of struggle after completion. Beethoven's 5th symphony took eight years to complete, but all sketches were destroyed.

66 Liszt experienced during his life many changes to the keyboard and piano technology.

67 Flanagin and Holloway, *Patterns of creativity and Composition in Software Development,* 2002.

68 Metafont, tex, latex is an example of the knowledge acquisition and automatization of the typesetting and printing process. It is today done by typesetting engines like XML, tech and latex.

69 The mathematical philosopher Leibniz liked to talk of unconscious calculations in the head of musicians and composers.

70 This is in contrast to requirements of definiteness, expressed by Knuth, *Algorithms.*

71 For example robots playing games on behalf of humans and so on.

72 Spiegel, 1987.

73 Lakoff, G. and Johnson, M., *Metaphors we live by* ;University of Chicago Press, 1980.
Johnson-Laird, P.N., *Mental Models*, Cambridge, 1983.

74 Gärdenfors, Peter, *Conceptual Spaces – The Geometry of Thought,* Cambridge, MA: MIT Press, 2000.

75 Wikipedia: Creativity. Freud started this line of argument with his now classic theory of artistic creation as a re-directed suppressed sexual or other non-tolerated form of internal energy or forces. The difference between the creative artist and the mental patient is that the artist sublimated (modified mental conflict due to social norms) successfully what the "ill" repressed unsuccessfully.

76 Wikipedia: Citation of Paul Palnik: "To be creative means to become profoundly individualized thus separating one's self from the crowd"

77 Joseph Schumpeter's use of "creative destruction" or so-called "creative processes" in organizations.

78 There exist composers (artists) that intentionally break rules (e.g. do the opposite), but this can easily be reconstructed as a radical introduction of many new rules. We may think of serialism, but also of counterpoint-excercising of anti-fux ruling. As to the possibility of breaking all rules, Donald Davidson's analyses forbids even the thought of them.

79 Obvious examples are found in Bach's bold solutions to counterpoint that stretched the book of Fux.

80 Dewey's definition of a creative problem (indeterminate situation).

81 It is somehow ironic that the example of the staircases is taken from "Penrose staircases" but used to show another story altogether than the one Penrose propagates himself.

82 Hofstadter: Gödel, Escher, Bach, 1999. From the preface (in the 20th anniversary edition only)

83 Robert Penrose: "Quasicrystals and Kleenex"(Issue 16 of Plus)

84 Earlier sceptics of AI, like Weizenbaum and Searle have weakened their claims in the aftermath.

85 Dennett has critizised Penrose's view on algorithms and consciousness in several articles.

86 See Dennett in ch2,as well as Igor Aleksander: *Impossible minds*; Damasio: *Descarte's error* and others.

87 Weisberg citation?

88 Dennett, Daniel C., *Darwin's Dangerous Idea: Evolution and the meaning of life,* Gardners, 1996, 60.

89 Ibid., 59.

90 Both the perseverance of field ornithologists as listeners to birds and the active incorproations and reprocessing of bird themes into countless compositions by generations of composers (van Eyk's Nightingale variations, 16th c, Mozart, Beethoven, Messiaen among others, see Wikipedia s.b. Bird song) are illustrating this fascination.

91 Most bird species's have a characteristic commonality, that some believe might be hereditary based.

92 Biologist Peter Marler, 2000, cited in Mithen, 283. Bird calls are shorter and more intentional clear than songs.

93 Wikipedia s.b.Bird songs

94 Ibid.

95 Stephen Wolfram. *A new kind of science,*1180. But he doesn't want to exclude the more common-sensical explanation either: "Or it could be that humans find them pleasing because they are familiar from bird songs."

96 Mithen, Stephen, *The singing Neanderthal,* 2005, 285.

97 Charles Darwin. *The descent of Man, and Selection in relation to sex,* 1871.

98 Evolutionary psychologist Geoffrey Miller wrote even a modernized version in his "Evolution of human music through human selection",in N.W.Wallin (ed.).*The origin of music,* 1999, MIT Press.

99 The handicap principle, proposed by biologist Amotz Zahavi (1975) is even "upgrading" Darwins sexual selection principles by stating that "sexual ornaments should be costly to be attractive, so that they can accurately advertise biological fitness"(Wikipedia s.b.Handicap prinicple). In this theory, risk-taking behaviour and traits like the beautyful *and* alerting bird song and tails of peacocks, rock musicians lifestyles, bungee jumping and even education (economist Michael Spencer) in general are examples of signaling fitness but in addition to sexual succes will include other disadvantages and risks. The fact that they are risky are making them "honest signals" that can't be faked like less "exibitionistic" displays. Game theory has been applied to modelling the 'handicap principle'.

100 Mithen, Stephen, *The singing Neanderthal,* 2005, 284.

101Even octopus or squids are of interest in a more wider sense. Cephalopods, especially, squids are flashing with skin color patterns using their intracate biochemistrical bag of tricks to communicate with other squids in ways humans have not decoded . The patterns these highly intelligent and learning organisms “choose” to generate are so abstract and symmetrical that one is tempted to think of parallels with art. But these translocating pigments are well known in squids engaging in courtship rituals, in line with Darwin's theory of patterns of sound.Moynihan, M., *Communication and Noncommunication by Cephalopods.* Bloomington: Indiana University Press, 1985.

102Mithen, 2005, 284.

103Neurobiologist Timothy Holy. The results of his study were published by Eric Hand in the *St.Louis Post*, Nov.1,2005

104Apart from gibbons where male and female show extended duet singing to apparently affirm territorial and social status. Since these songs are fixed during liftetime, researchers believe them to be “biologically fixed at birth” Geissmann, 2000 in Mithen, 284.

105Similarly to how different 'eye designs' have been selected by very different scenarios and times in different species (see Daniel Dennett, *Darwin's Dangerous Idea*)

106i.e.unexplained at present and supposed difficult to explain.

107Rhythm abilities develops in human children after localization, pitch and tonality; only before style.(Seminar in music perception 838, Ohio University)

108in Peter Gärdenfors. *How homo became sapiens* (PG2003)

109Frans de Waal (1988) reported bonobos (the sexual and social very cohesive relative of chimpanzees) to emit staccato calls in choro.

110Another illustratration are the drum beats on slave ships in antiquity that “propagated” ships due to coordinated rowing.

111The idea is that forceful shouting and keeping time proves control of ones voices and hence their bodies. This may seem somehow ad-hoc though. (Gärdenfors, 2003,163-164)

112Ibid.

113Gärdenfors, Peter, *How Homo bacame Sapiens – On the evolution of thinking,* 2003, 164.

114Steven Pinker, *How the mind works,* 1997.

115Steven Pinker, *How the mind works,* 1997, 529-539, cited in Mithen, *The singing Neanderthal,* 2005.

116Mithen, *The singing Neanderthal,* 2005..

117Ibid.

118John Blacking. *How musical is Man?,* 1995; cited by Mithen,2005,5.

119Mithen, *The singing Neanderthal,* 2005, 266.

120“Changes in anatomy, foraging behavior and social life between the time of the common ancestor at 6 million years ago....My belief is that their vocal and gestural utterances remained holistic, in the sense that they were complete messages rather than words to be combined , and were employed to manipulate the behaviour of other rather than to tell them things about the world. (Mithen,*The singing Neanderthal,* 138)

121Homo ergaster is the “successor” of Australopithecines “taking over” around 1.8 million years ago.

122Mithen considers the elaborate hand-axes from hominid findings, possible examples of artificial (unnecessary) activity to attract the opposite sex, females, and thinks this sexy hand-axe theory can enlighten the function of sexy singing (Mithen, *The singing Neanderthal,* 2005,191). In our days, we might think of newest model mobile phones as flashing signals of being someone who matters.

123Mithen refers to Merlin Donalds, Origin of the Modern Mind,1991, that assumed 'mimetic culture' (mimesis, mimicry, imitiation) being the sucessor of modern communication.

124This is especially true for the human species' that had to live through adverse conditions, like ice ages.

125Evolutionary theory has for long been entrenched in an individual-centered model, making it difficult to understand how cooperative behavior might have evolved at all. See prisoners dilemma. Hmmmmm communication may be a missing link in this type of explanation.

126“Wray uses the term 'segmentation' to describe the process whereby humans began to break up holistic phrases into separate units, each of which had its own referential meaning and could then be recombined with units from other utterances to create an infinite array of new utterances. (Mithen,*The singing Neanderthal,* 253)

127I interpret the figure 1 of Hominid phylogeny, such that the transition from Heidelbergensis to Homo sapiens could have involved a evolutionary bottleneck effect?

128Mithen, *The singing Neanderthal,* 2005.

129Ibid.

130As described in ch1, using recombination as central idea. See also Dennett about evolution in general.

131As a consquence developing civilization, i.e. greater scale control and co-existing based on hierarchic structures and power behavior.

132Altruistic means here not metaphysically sacrifying one self, but trusting the groups greater value compared to an individualistic lifestyle.

133Mithen, *The singing Neanderthal,* 2005,63.

134Dysfunctional understood relative to a cultural majority. These uncommon states are sometimes so strange and exotic (see “The man who mistook his hat for his wife” e.g.)

135Mithen, *The singing Neanderthal,* 2005,76.
136Not too unlike the abilities of newborns to swim instinctively. It too, vanishes probably because of little usefulness in typical lives of young Sapiens today.
137IDS: Infant directed speech
138Mithen, *The singing Neanderthal,* 2005, 72, 84.
139A very funny finding states that woman standing in the surroundings of babies without the possibility of body contact will show even stronger IDS behavior, interpreted as a compensating method to reach out and get in contact.
140Laughter is a very old phylogenetic adaptation to relationship-building.
141Mithen sees this effect not so much in the light increasing intellence, instrumental or otherwise, but as a technology to induce short-living states of happiness. This is explained in the fact that music is a "language" of emotions that is hard-wired into us and satisfies deep-seated needs of grouping and emotional stimuli. (Mithen, *The singing Neanderthal,* 2005,100)
142Including Gärdenfors and Pinker.
143Mithen, *The singing Neanderthal,* 2005, 7.
144My own hypothesis is that the ability to instrumentalize must be matched by our ability to feel appropriate emotions, or what we now may call morals. Is e.g. the pilot of an atom bomb carrying plane morally fit to turn the switch? See also Konrad Lorenz that thought the hostility between human groups to be the most dangerous form of aggresion in humans. (Leslie Stevenson: *Seven theories of human nature.* 2.ed. UOP, 1987,174 [norwegian edition])
145An example is the repeated and prosody-enriched version of "du og jag" (You and me) from Emil (swedish tv child series by Astrid Lindgren)
146Hominids and probably Neanderthals had higher positioned larynxes, leading to such assumptions. See also higher voiced IDS.
147With the possible exception of Pinker's auditory cheesecake hypothesis
148"Emotions are deeply entwined with the functioning of human cognition and physiology; they are a control system for body and mind. It is most unlikely, therefore, that our deepest emotions would be so easily and profoundly stirred by music if it were no more than a recent human intervention... In fact, even when we sit still, the motor areas of our brain are activated by music." (Mithen, *The singing Neanderthal,* 2005,25)
149Mithen does not mention the theories of language by Vygotsky that support his musilanguage theory further.
150Strictly figuratively speaking. But there are theories that try to update the static character of Kant's categories into a dynamic frame system. (Wikipedia: categories of Mind). Still, theories of Chomsky and Jackendoff are more related to Kantian thinking than the proposed learning model.
151Merlin Donald's acccount of mimesis states that children imitate freely (improvising) over the themes of their family members; may be not to unlike teenagers finding a somehow different way through life than their parents.
152Evt. even reacting to her own output (auto-feedback?)
153See 'Hmmmmm' as a holistic communication and activity above (Mithen).
154His teacher.
155Not too unlike the informal and auditory learning in the more primi-tive cultures we know of today.
156Statistical learning theory about undiscriminated acoustic input in Mithen.
157Dennett's idea of the intelligent design in terms of crane's engineering power.
158i.e.the advanced 'Hmmmmm' and Music.
159Robert Weisberg: *Genius and other myths.*1987. Its subtitle "What you, Mozart, Einstein, Picasso have in common", says it all, for now. The rest will have to wait until ch9.
160Ironically, Merlin thinks of mimesis as the oldest and most primitive form of literal copying practiced by hominids.
161Music academies, as well as music schools deliberately structure their activities into seperate paths, leaving enriched learning of multiple instruments e.g. to "lower" forms of culture, such as gypsies and africans.
162With others but also with one-self, playing privately and solo, in expectancy of potential future social meanderings.
163Especially western, where the newspaper and concert Hall has taken the place of the narrative and ceremonial local context and inclusive contexts. It may be an alienating factor nonetheless.
164See 2.1 for model mentioned above by Isabelle Peretz. In a provocative way, we may ask if modern sector proficiency is somehow related to certain forms of stronger versions we call dysfunctions?
165Virtuosos acting like circus artists, soloists that cannot accompany others, only rhythm playing professionals etc.
166Ethnomusicologist Bruno Nettl in Mithen, *The singing Neanderthal,* 2005,11.
167AIMA,90 [figure 3.16].
168Related to the Turing test mentioned in ch1.
169"One might think that this statistical learning capability [of eigth month old children] is part of some dedicated mental machinery for language acquisition. But Saffran and her colleagues repeated the experiment with musical tones...The results were equivalent to those for syllables, suggesting that at that age we have a general ability to extract patterns from auditory stimuli.... If that i correct, then this ability must relate to how we get the most general handle on the world at a very young age." (Mithen, *The singing Neanderthal,* 2005,76)
170AIMA, 526.
171AIMA, 529.
172AIMA, 531ff, e.g. inductive and statistical learning.
173AIMA, 527.

174Clasical concerts are situations of unsupervised learning. The contentness of the public is not shown underways, but only after ending a musical work. This makes it difficult for a performer to know if tempo/mood etc. are well adapted to this particular public. The jazz session is an example of reinforcement learning (public is clapping and shouting under playing). Supervised learning is typical of lessons where most good teachers use mimesis to "explain" the right version. Explaining words are somewhere in between supervised and reinforcement learning.

175That singing in community is a strong emotional experience is now easier to understand. The loosing of one's boundaries with the group is fundamentally a return to advanced 'Hmmmmm' type communication of tribal strength of a community.

176"He [John Blacking] favored the idea that there was no such thing as an unmusical human being, and noted that the existence of a Bach or a Beethoven was only possible because of the presence of a discriminating audience."Mithen, *The singing Neanderthal,* 2005,13ff.

177How can we tell someone to look at a banana as if it were coconut? Or telling children that x stands for anything between 1 and 1000? We might begin with generalizing and understanding 'banana' as 'food' and x as any natural number. Perplexities of representational concepts are further discussed in 2.5 and to some extent in ch.9

178We remember from 1.1 that also the 'Hmmmmm' referenced states of interest, but in a very limited way and precision. We recall that it was manipulative and holistic. Covering one's face would mean some kind of *self shaming*, (see sociologist Thomas Scheff) but couldn't point to the exact matter of affairs.

179'Hmmmmm' communication is not machine-like enough for calling it a machine, neither is the advanced music-like 'Hmmmmm' of the Neanderthalian culture. A reason for distinguishing here, is the sustained stasis of culture by Neanderthals, while human "lingua-machines" may be evolutionary necessary steps to the "Machiavellian mind or intelligence theory" also known as the social brain hypothesis in evolutionary sciences. (Robin I.M.Dunbar: *The Social Brain Hypothesis*. Evolutionary Anthropology.

180Abelson/Sussman,1975, 4.

181Willard van Quine: *Word and Object,* Cambridge, MA: MIT Press, 1960.

182Ibid.

183Ibid. 80. fig.2.1

184Daniel Dennett, Consiousness explained. The intentional stance uses functional higher levels to abstract from non-relevant details (abstraction barriers below).

185Is this actually the same process of 'Hmmmmm'-mimetics, just higher up?

186Even the areal attributions are mostly questioned in some ways.

187In problems of discrimination, generalization, induction, learning among others, and generally where knowledge domains are insufficiently explored, i.e. where there does not exist a formal apparatus and highly segmented representation (syntax/semantics), artificial networks are often the choice of AI engineers.

188George Lakoff, Mark Johnson: *Metaphors we live by.* Chicago Press, 1980. The authors were early contenders of the theory that natural languages, unlike formal and computational/symbolic ones, are inherently fluid and changing. This view on language is somehow in conflict with Mithens polar description of music and language. "We shall argue that, on the contrary, *human thought processes*are largely metaphorical", 6.

189Peter Gärdenfors: *Conceptual spaces. The geometry of thought.*MIT Press, 2000.

190Subconceptual is used by Gärdenfors to clarify the relation between conceptual, symbolic and subconceptual modes.

191Gärdenfors, Peter, *Conceptual Spaces – The Geometry of Thought,* Cambridge, MA: MIT Press, 2000, 60.

192This is the common-sensical expression of spatially non-continuos and non-symmtrical 'spaces' or structures.

193This analysis is still dependent on versions of evolutionary realism (Mithen, *The singing Neanderthal,* 2005,82), but since Gärdenfors declares himself as instrumentalist he is not claiming more than the existence of evolutionary grown structures of perception and concepts in cognition. (p31)

194Peter Gärdenfors: *Reasoning about Categories in Conceptual Spaces.* Proceedings of the 14.International Joint Conference of AI, Morgen Kaufmann, 385-392, 2001,386.

195See morphological concepts in Godøy, 1993.

196Ibid., 388.

197Conceptual spaces seem therefore to explain the possibility of sharing knowledge with other "conceptual spaces", e.g. persons.

198Gärdenfors.*Conceptual spaces.30.*

199Or other emotional machines.

200Affective Computing Group, MIT Media Lab.

201Http://affect.media.mit.edu.

202Rosalind Picard in an interview with FM at www.firstmonday.org/issues/issue3_4/picard/.

203"Indeed, it is coping with the demands of living in large social groups that provides the most likely explanation for the origin of human intelligence. And so it is not surprising that our more complex emotions relate directly to our social relationships. Without such emotions we would be socially inept; in fact, we would be unaware of the complexities and subtleties of the social world around us, and would fail entirely in our social relationships." Mithen, *The singing Neanderthal,* 2005,87.

204Picard, R.W. (in press), *Toward Machines with Emotional Intelligence*, Oxford UP, 2007.

205www.firstmonday.org/issues/issue3_4/picard/.

206Project at Affective Computing Group (IP-adress above).
207Such an interpretation-free and idealized level presupposes a brain of a newborn that has never reflected or experienced any auditory information. And hence nothing to relate to. But newborns have significant musical experience aquired indirectly during pregnancy.
208Virtual fundamental is a partial (the missing fundamental) we hear in a complex tone, even without it being there. The paradox is due to our brains benevolent and reasonable interpretation of a tone or chord in context. Virtual beats are the beats we anticipate and respond to that should have been there, provided a complete rhythmic actualization.
209AIMA, 153.
210Taking an instrumentalist stance, phenomenologists and post-modernist “interpretationalists” will not think so.
211The Vienna Circle's manifest of unified science in neutral logico-symbolic terms.
212If we record sound on a magnetic tape or any other modality-preserving physical structure and play it back afterwards, we still preserve only certain essential parts of the information, not everything, since the location and physical properties (compression) by of the microphone filters out some of the sonic richness that was caused by the acoustic complexity of the room. Still, a tape copy of the original sonic events, reproduces and copies therefore quite successfully without too much of representing involved. A tape does not look like sonic events, but is nonetheless able to recreates them in certain favorable conditions (a tape player etc).
213Loy: *Musicmathics,* 2006, 3.
214Ibid.
215Called the measurement theory in experimental sciences. Absolute borders between observation and theory are widely seen as impossible, see the analytic-synthethic-principle and W.v.O.Quine.
216Loy, Gareth, *Musimathics; the mathematical foundations of music,* Cambridge, MA: MIT Press, 2006, 11.
217Loy speaks of a note as a tone placed in a particular temporal context. (Gareth Loy: *Musimathics*,12)
218This is analogous to the intentional stance followed by Daniel Dennetts approach of defining intelligence.
219The oldest known recorded songs are around 4000 years old: Assyrian cuneiform artifacts.
220The oldest known written records date over 4000 years back in time (cuneiform tablet). (Wikipedia: Music notation)
221Denoting primarily monophonic melody with other properties written above and below with gestural signs. See Wikipedia, Kepatihan notation.
222This is somehow analogous to the sorting algorithms of 1.3, where optimality is relative to assumptions or knowledge of the distribution of data. No sorting algorithm is the fastest for all sets of unsorted data.
223Instrument attribution to a stave in a partitura may be written in the start. But this makes it for instance difficult to notate voices that start with violin sound that gradually, in non-linear fashion transform into trumpet. We might add such information by coloring notes with shades of blue-ish to red-ish to signify trumpetish to violinish sound change
224Dipartimento of infomatica musicale, Milano. Alberto Sametti: “*ScoreSynth: A system for the synthesis of Music Scores based on petri nets and a Music Algebra”,* 1991 in Gareth Loy, Musimathics, 2006.
225Loy: *Musicmathics,* 2006, 396.
226Haus and Sametti, 1991, 10; citated in Loy, *Musimathics*, 2006, 394.
227Or may be: “Oh, I heard something and I like it!”. Music critcs writing in papers will often display the relative poverty of natural languages in describing auditory or sensory experience altogether. Concert critics are often permeated with praisings to the skies and sequences of cliché-laden adjectives in shortage of better describing and musically relevant terms.
228Loy, *Musimathics, 13.*
229In recording sessions, recording engineers will use both conceptual systems to communicate with man and machines.
230Ibid.
231Loy, Musimathics, 2006, 197. Adapted from John M. Grey, 1975, *An exploration of musical timbre, Ph.D. Diss., Stanford University.*
232With some of them not even diatonical by definition, since their downward movement is *not* diatonical.
233Mithen, *The singing Neanderthal,* 2005,100-10.
234Various publications of Patrik Juslin and Klaus Scherer.
235Curtis Roads: The computer music tutorial, 1996.
236Richard Rorty, *Philosophy of the Mirror of Nature.*(Princeton: UP, 1979), 105-106.
Rene Descartes defended a version of strict dualism proclaiming wholly distinct body and mind, or completely separating the absolute and objective powers of rationality from the contrasting subjective 'passions of the soul'. ('Principiae philosophiae', 1644; 'Les Passions de l'ame')
237Advanced 'Hmmmmm' of the Neanderthals was probably non-compositional as well, regarding the relative stasis of their cultural development as evidence.
238Tones are often segmented in ADSR: attack/decay/systain/release; Loy, *Musimathics,36.*
239Cook, Nicholas, *A guide to music analysis,* Oxford: Oxford University Press, 1987/1994, 230.
240Ibid., 233.
241Rolf I. Godøy, *Formalization and Epistemology,*1993, 171. Section 6.5 Monstration
242Musical Instrument Digital Interface: used in commercial electronic instruments as well as research/academics
243“Hanging tones”.
244Nevertheless, MIDI data has been analyzed on higher levels to capture such information, such as in the analysis of

multivariate MIDI that has been used for statistical exploration of musical kinematics (M.Das, D.M.Howard, S.L.Smith:*Motion Curves in Music: The Statistical Analysis of Midi Data.* At http://doi.ieeecomputersociety.org/10.1109/EURMIC.1999.794756)

245Rolf I. Godøy, *Formalization and Epistemology,* 1993.

246Ibid.,150.

247Godøy stresses not only the problematic sides of certain *reductionisms*, but also the importance to representant at *"middle" or* levels (such as the morpological or spatial levels), i.e. not exclusively tone-centered approaches of representation. This is an important point, as well, in respect ot the sub-tonal realities and representations (of overtones), presupposed in practices of "overtone singing" (see www.harmonicpresence.org) and other similar projects.

248Rolf I. Godøy, *Formalization and Epistemology,* 1993, 163.

249"Singing and Dancing together" in Steven Mithen, *The singing Neanderthal,* 208, 157.
Musicial writer Manoff defines 'musical style' as the characteristic manner in which a basic musical impulse is expressed. *Dance is but one of these impulses*."(Tom Manoff, *Music, a living language,* 1982,1*)*

250Gärdenfors Williams,*Reasoning about Categories in Conceptual Spaces,*2004.

251See also Heterophenomenology Reconsidered (on-line-reprint)

252Daniel Dennett, Wikipedia:Heterophenomenology.

253Joseph Goguen, "Musical Qualia, Context, Time, and Emotion"*,* in *Journal of Consciousness Studies 11, 2004, 117-147.* internet: www-cse.ucsd.edu/~goguen/pps/mq.pdf. Subjects are also treated in:
Joseph Goguen, Ryoko Goguen, *Time, Structure and Emotion in Music,* 2003,
internet: www.cs.ucsd.edu/~goguen/projs/ars.html.

254John Bickle, *Rethinking Reductionism,* 2006.

255Godøy seems to admit this fact; Godøy, Formalization and Epistemology, 171(footnote 19).

256Achille C. Varzi, Basic Problems of Mereotopology, 1998, published in N.Guarino (ed.), *Formal Ontology in Information Systems, Amsterdam: IOS Press, 1998, 29-38.*
Achille C. Varzi, Roberto Casati. *Parts and Places: The Structures of Spatial Representation,* 1999.

257Ibid. "All entities are treated as Individuals, i.e. as entities of the lowest logical type. See: Nelson, Goodman, *Structures of Appearance*, Harvard University Press, 1951/1977.

258Rudolf Carnap, Logical syntax, 1937. This principle was that proposed elaborate "calculi for scientific languages

259'Rudolf Carnap: Logical syntax, 1937.

260Loy, *Musimathics, 13.*

261Cook, *A guide to music analysis,* 233.

262Wilkins, *Creative Music Compostion,*11.

263See ch1 and comparison between logical connectives and the trinity of expression.

264Dennett, *Darwin's Dangerous Idea,* 1995.

265Moshe Sipper: *Machine Nature, (*NewYork:McGraw-Hill, 2002).prologue, see also 9.6.

266Due to complexity or evolutionary open systems (environmental shaping).

267Quine's answer to epistemological questions.

268Abelson, Sussman: Structure and interpretation of computer programs, 1985, xvi.

269Miller Puckette, Max at seventeen, 2002, 13.

270Winkler, Todd, *Composing Interactive Music,* 2.ed.,Cambridge, MA: MIT Press, 2001, 12. About Morton Subotnik.

271Ibid., 49.

272This example shows how important the order of message timing or execution is for meaningful behavior of patches. If the right inlet receives context-dependant or local values for delay after a note value is reveived in the left inlet, the note transmitted will have to use an eventual fixed or default value from the objects definition.

273Pipelines in Unix are set of processes chained by streams from one pipe to the next. Pipe scheduling is solved by *buffering,* data is held in queues waiting for the next 100 bytes or so. This pipe system avoids problems of multitasking, such as deadlocks. Pipes are the equivalent of objects in MAX.

274Lists can contain both numerical and symbol data, but its first value must be a numerical.

275 e.g. *[if $i4 > 48 && $i1 > 80 then out1 $i2]* sends the value at the second inlet to the first outlet if values at inlet 4 and 1 are under 48 and over 80. Conditional decisions are user defined with if. To make such decisions even more flexible and complex, one can use the expression object *expr* as part of conditional definitions with *expr* evaluating a mathematical expression only.

276Sic *Goto* in Basic programming language

277Winkler, Todd, *Composing Interactive Music,* 2.ed.,Cambridge, MA: MIT Press, 2001, 74.

278Ibid., 76.

279Philosophical questions about mind and machine and the nature of creativity belong to these questions that I posed in ch1 and try to illuminate in the conclusive chapter 9. See also the use of 'agent' in AIMA and debates about singularity in AI, hard and soft AI, as well as similar debates.

280In interactive composing MC it is most naturally to think of machines as composing assistants that fill in details according to constraints leading to consistent pieces under less laborious conditions. The opposite, where a

computer program decides the grand scheme or form asking its human assistant to fill out technicalities would counter our most fundamental intentions and aspirations with machine composition, but illustrate quite well some unconscious presuppositions in the field.

281Todd Winkler, *Composing Interactive Music,* 2.ed.,Cambridge, MA: MIT Press, 2001, 176.

282Ibid., 177.

283Melodoy and 1/f fractal noise generation; (Chapel, 2003).

284Mathematically minded departures are often exposed to critique of lack of human relevance, deriding computer music as composing for machines, not humans. See also 3.2 (Early computer music).

285Todd Winkler, *Composing Interactive Music,* 2.ed.,Cambridge, MA: MIT Press, 2001, 214.

2860,1,333, 5,666, 16, 33,333 ? [$F(x)=(x-1/3)*x^2$ behaves like this for x=1,2,3,4,5].

287the 1986 ICMC in Netherlands, discussions about abstract problem of scheduling real-time computer music performances.

288Puckette, Miller, *Max at Seventeen,* 2002.

289 In Pd objects from these domains share even the look, inviting further to confusion or puzzlement.

290Kyma or CM are musical composition language that complies with OOP-requirements.

291Puckette, Miller, *Max at Seventeen,* 2002, 12.

292Ibid.

293In Max/MSP the relative screen position of objects determines the execution order (right to left), i.e. the screen appearance shows the message flow but also meaning that “tidying up” patch windows must be made prudently having the functionality in mind.

294 “...software hegemonists' concern is not letting people do their own thing, and their products will always impede, rather than encourage process”. MP 2002, p12; this does not imply a critique of Zicarelli and Cycling 74.com since he refines the matter by: “The existence of software companies is an enabling one, as long as they don't engage in predatory practices”, where he may think of Gibson inc.'s rather predatory dissolving of Opcode.com after bankruptcy in the early 90s.

295Puckette, Miller, *Max at Seventeen,* 2002, p13.

296Rowe, Robert, *Machine Musicianship,* Cambridge, MA: MIT Press, 2001.

297Method-guided instruction is rule-guided while immersing teaching is typically the practice of demonstrating by example or engaging in ensemble playing.

298ANN is also called parallel distributed processing after Rumelhart and McClelland, 1986

299However, as computer music is so often accused of leading us to a day when machines will listen only to machines, I feel compelled to observe that many of us are motivated by a much different vision of the computer's potential connection to the community of human musicians. Rowe, *Machine Musicianship,* 2001.

300Doorbells that react to wind changes (fluctuations in air pressure) are automatons that are not musically interesting for other than philosophical discussion (see “LaMettrian” robotic gardens with moving sculptures and fountains, ch9.8)

301AIMA, 31.

302Ibid, 32. Figure 2.1.

303AIMA, 33.

304Rowe, Robert, *Interactive Music systems Machine listening and composing,* Cambridge, MA: MIT Press, 1993.

305Rowe himself admits that such distinctions are “fine one, and tends to blur easily, as do many of the classification metrics we have been discussing. Still,it seems to capture a noticeable difference in the way composers approach algorithmic methods: either the machine is changing something it hears or is generating its own material from stored data and procedures.” (Rowe, *IMS,* 186).

Rowe proposes a distinction between transforming and generative methods relative to whether they presuppose live input or not. A better criterium seems to me to lie in the size of the output relative to the size of the input. Generative methods are most productive, transformative methods more than triggering methods that often produce mostly timing schemes for playing them in a live context.

306Accompanying when referring to improvisational systems ('interactive music systems' in Rowe's terminology).

307See for example Roger Dannenberg: Dynamic programming fro interactive music systems in Miranda, Eduardo Reck (ed.), *Readings in Music and Artificial Intelligence,*Harwood A P, 2000.

308There is a weak analogy to negative and positive patch definitions (ch3) here.

309AIMA, 51.

310Corelli relied on sequences to achieve clear tonal organization. Whether constructed diatonically within one key or modulating downward in the circle of fifths, the sequence is a powerful agent for establishing tonality. (Russano Haning: Concise history of western music, 1998, p239)

311This example very much fits the reflex-agent with internal state from AIMA,43. It finds a rule, in this case the cof-function that matches the current situation. The internal state somehow describes the current scale/key to identify a situation as functional dominant relative to a tonic.

312A MCs that doesn't need information about the current state of the musical world would still have to relate to the overall structure of the output (its constructed “world”) while evaluating choices during the process. A more straightforward use of world or environment would be the inclusion of references to extra-musical states.

313Greek: spectacle (a public sight), comtemplation, consideration; this original meaning fits very well with music theory's somehow less strict use than the one practiced in “mathematized” theories in natural sciences today. The abstract principles embodied in music and its sounds.

314Analysis (gr.): a loosing or releasing [of bindings between parts].

315In more recent tone organizing systems as the serial or twelve-tone-system have more rigid rules than tonal music. Other inventions of “syntaxes” for composing or other cultures tone organizing systems are examples for non-tonal music systems.

316major: 2-2-1-2-2-2-1, minor as 2-1-2-2-2-2-1 and 2-2-1-2-2-1-2 moving up-and down-wards respectively, where numbers represent semi-tones from last to next tone in scale.

317In antique history periods, emotions, mathematics and music were united in encompassing explanations. They even anticipated modern norm-discussions and possibly thoughts in music therapy about the consequences on character of musical consumption (citation Plato), 9.2.

318Going back to William James and prominently illustrated in behaviorist theories at Meyer's time.

319Meyer's theory of meaning in music would have probably been rejected by classic critics like Hanslick and Stravinsky that excluded emotions as relevant principles for analysis in favor for intellectual studies (aesthetic formalism). Meyer and other expressionists like Susanne Langer's theory of meaning did not imply referentialists or extra-musical meaning typical of progam music. They might be absolutist about musical meaning, i.e. believing that it stays within the context of the work.

320Lerdahl wrote another book that expands on GTTM.
Lerdahl, Fred, *Tonal Pitch Space,* Oxford: Oxford University Press, 2001.

321Lerdahl, Fred, *Tonal Pitch Space,* Oxford: Oxford University Press, 2001.
Tsougras followed up on 'Tonal pitch space' with applying its general theory to modal systems of medieval period in his *Modal Pitch Space – A theoretical and analytical study,* 2003.

322Cook, Nicholas, *A guide to music analysis,* Oxford: Oxford University Press, 1987/1994.

323Eventually reduced to a period of his life.

324Rowe, Robert, *Machine Musicianship,* Cambridge, MA: MIT Press, 200, 60.

325Key is a group of related notes belonging to either a major or minor scale, plus the chords formed from these notes, and the hierarchy of relationships among those chords. The first note is called the tonic or keynote and is the focal point towards which all notes and chords gravitate. ('characteristic final note' in modal theory). The larger organizational system embracing keys, key relationships, chord relationships and harmonic goals is called 'tonality'. (adapted fr. Encyclopedia Brittanica/'Harmony').

326Rowe, Robert, *Machine Musicianship,* Cambridge, MA: MIT Press, 2001, 36.

327The composer Allen Forte defines pc sets more narrowly as equivalent when related by transposition or inversion followed by transposition (Rowe, *Machine Musicianship,* 2001,21), resulting in only 12 three-note-pitch-class sets.

32838,47,59: major triads; 37,49,58: minor triads; 36,39,69,48 dim.and augm triads

329Rowe, Robert, *Machine Musicianship,* Cambridge, MA: MIT Press, 2001, 36.

330Rowe, *Machine Musicianship,* 2001.

331The reason there is a final or absolute number of theoretical or possible chords lies in the presumption that only pitch classes are considered. So there are only 12 possible 11-tone-chords. The greatest number of chords is in the middle (six-tone-chords or sextads of which there are 924). See Rowe, *Machine Musicianship,*36. Table 2.4 for the symmetrical distribution of chords around sextads as axis.)

332Rowe, Robert, *Machine Musicianship,* Cambridge, MA: MIT Press, 2001, 19.

333“The choice between a Gb and F# minor chord should be made based on the surrounding chords and, ultimately, the key in which the sonority is embedded” (Rowe, *Machine Musicianship,* 2001,43). This actually presupposes that structural and functional consistency overrules pragmatic considerations like A is easier to read than Bbb in Gb-Bbb-Db!

334“The line of fifth is similar to the circle of fifths except that it extends infinitely in either direction” , Rowe, *Machine Musicianship,* 2001,44. In contrast with the circle of fifths circling around a kind of center.

335Rowe, *Machine Musicianship,* 2001,46.

336The line of fifths distance is related to preference rules of 'Pitch variance' and 'Harmonic variance' e.g.

337RR demonstrate this algorithm through an implementation in C++ (Rowe, *Machine Musicianship,* 2001,52ff). Rowe discusses also results from examples (Beethoven sonatas) that weakly diverge from common, ie non-computational analysis showing that these originate typically in more ambiguous chord progressions and are therefore more predicted by RP's model

338If a major/minor third above the computed root is present we call the chord major/minor deriving names of member tones thereafter.(RRMM59) If both type-determining major and minor are present there is a tonal conflict or inconsistency that is excluded by theory. If none of them is present we might presume a virtual chord type to present relative to neighboring chord types and progressions.

339Rowe, Robert, *Machine Musicianship,* Cambridge, MA: MIT Press, 2001,66.

340Interestingly, Rowe found that at least from the perspective of algorithmic effectiveness (quickness), weight reductions or candidate exclusions based on certain negative, ie.inconsistent influences. Rowe, *Machine Musicianship,* 2001,62-66.

341Skyhooks of Dennett...see also Marc Leman's Music and Schema Theory citated in Rowe, *Machine Musicianship,*

2001.

342Rowe, *Machine Musicianship,* 2001, 66.

343Primacy factor is the privileged status of first and possibly last events, somehow related to the common sense idea of heightened significance of first (and last?) impressions . This idea is also present in compositional practice that consciously formulates rules for “efficient” composition (esp. evident in schools of popular music that are restrictive about length of subjects etc)

344Rowe, *Machine Musicianship,* 2001,70.

345Atari, Amiga, PC. M was revived in 1997 (v2.5 in 1999) after distribution and development was interrupted since 1991.

346 *Upbeat* was software specialized for generation of flexible drum patterns (without pitch). Other similar programs were *OvalTune, Realtime* including some algorithmic features in OpCodes early *Sequencer.* Another influence was the universal adoption of real-time editing in music sequencer programs generally. (www.cycling74.com)

347 In a letter to CMJ entitled “A short History of Intelligent Instruments” Laurie Spiegel rebutts the idea that the programs created in the later 80's are of historical importance. She stresses the historical continuity of these family of programs and gives examples of historical precedents to IM's programs (among them Dr.T's, Marvin Minsky's Muse (early 70's) and Moog's modular analog synthesizers).

348Laurie Spiegel, *Brief History of Intelligent Instruments,* Computer Music Journal, Vol. 11, #3, 1987.

349 Rowe mentions that a timbral analysis could be called level 0, but since he uses MIDI as representational level, this information is abstracted away in Cypher.

350 L1-Features: All, line, oct1, oct2, wide low, mdlo mdi, high, slow, mslo, mfas, fast, soft loud, shrt long, C, C#, D,D#... L1-Methods: all, phrs, fort, acnt, bkwd decl quie thin , flat, accl, invt, tite, swng, tril loop arpg, chrd, strc, bass, trns, trem obli, orna, glis, solo,...

351Rowe, *Interactive Music systems*,1993,121.

352 This reminds me of Ed Thorpe's method of “counting” the odds or probabilities for winning against Blackjack in Casino's by simplifying the exact prob value counting using -1 for cards 2 to 7, 0 for cards 8 to J and +1 for card Q to A. This made it possible for “head-counting” the probabilities to beat the Card dealer, even if the odds against the player from the outset were negative. (Ed Thorpe: Beating Blackjack, 1962).

353“A thing or idea seems meaningful only when we have several different ways to represent it – different perspectives and different associations”, Minsky, Marvin, *Society of minds,*New York: Simon and Schuster*,* 1986, 640.

354 “That is, musical structures should not be analyzed as consisting of levels systematically stacked like blocks... but rather as intertwined, reticulated complexes – as integrated nonuniform hiararchies. Unity would then be a result of the interlocking connections that occur when implications are realized between parts rather than as a result of relationships determined by the assumption of a preexisting whole”; (EG1977,97).

355Rowe, *Interactive Music systems*,1993,106.

356Rowe, *Interactive Music systems*,1993, 103-109.

357 A machine-learning technique using back-propagation would have a good potential of finding hidden knowledge.

358 Illustrating in some respect the controversy in the philosophy of science between deductivists and inductivists (or Popperians vs. Carnapians).

359“Beat tracking is the process of finding a duration to represent the perceived interval of a beat, described as that level of temporal periodicity in music to which a listener [or performer]would tap a foot, or a conductor move a baton.” (Chung, 1989).

360Cooper/Meyer (1960) divide into surface pulse, medium-level meter and ultimately patterns of strong and weak beats called rhythm.

361Rowe, *Interactive Music systems*,1993, 145.

362 Smaller tempo contours resulting in “imprecise” placing is detected in C with a “zeroing in-technique”, (Rowe, *Interactive Music systems*,1993, 149) Factorialization amounts to continuing mapping surface arrivals to deep structure theories calculated by divisions and multiplications.(Longuet-Higgins/Lee, in: 'Mental processes', 1984)

363Rowe, *Interactive Music systems*,1993, 155.
Special weights are added when harmonic features like tonic or dominant notes are present.

364 Messages may be *xform* and *mutate,* while arguments change according to the message and object type.

365 “For example, the accelerando module can be made to accelerate its input more or less than it already does as a function of the speed feature found by the listener and the accelerando rate currently active”; Rowe, *Interactive Music systems*,1993,196.

366Rowe, *Interactive Music systems*,1993, 200.

367Ibid., section 2.5.

368Other more elementary examples can be found in ch3 above and especially in Winkler, Todd, *Composing Interactive Music,* 2.ed.,Cambridge, MA: MIT Press, 2001.

369Jaxitron (Pseudonym), *Cybernetic Music ,*TAB, 1985.

370 APL stand for A programming language and was specializing in flexible array and list representation and processing.APL's most obious attribute is succinctness.

371 While it may be more difficult to program, an algorithm founded on musical logic rather than permutations will be easier for a composer to control and will lead to far more satisfying musical results. (Jaxitron , *Cybernetic Music,* 1985, 269)

372M ← -3 + 1 1 1 DVLP 1 2 2 2], where the right argument sets up the kernel for a scale; the left (111) a featured motif, and the -3 adjusts the output to a vocal range.
373Jaxitron , *Cybernetic Music,* 1985, 51.
374Ibid., 60.
375Ibid.,193-4.
376Given melody, harmony, and voiced chords, our ultimate goal is to produce something resembling a musical score. While there are countless problems of musical logic to be solved ... (Jaxitron , *Cybernetic Music,* 1985, 273)
377Jaxitron , *Cybernetic Music,* 1985, 280.
378APL uses many “strange” symbols (quad, caret, geometric shapes and so on) and rather “dense definitions” producing difficult to read code for novices at the same time easing overview and efficiency for an informed user.
379Balaban, Mira a.o (ed.), *Understanding Music with AI: Perspectives on Music Cognition,* AAAI Press 1992, 304.
380Minsky, Marvin, *Society of minds,*New York: Simon and Schuster, 1986.
381AIMA, 5.
382Minsky, Marvin, *Society of minds,*New York: Simon and Schuster, 1986, 326.
383Rowe, *Interactive Music systems*,1993, 124/156-7.
384Ibid., 212.
385Ibid., 216.
386Ibid., 200/220.
387Ibid., 154.
388Ibid., 227.
389Movie-like 'event calculus' calls events with sub-events for intervals (AIMA, 234)
390AIMA, 298.
391AIMA, 73.
392AIMA, 84.
393To appreciate the numeric magnitudes involved here, let's consider one highly “ordered” category. A commonly acceptable sequence might well be one that combines just one or two changes for each of the seven pairs of tonalities. If on the average there are 30 ways to have a single change of pitch and 80 ways to alter two pitches between successive structures, then the number of eight-chord sequences that have six single-note changes and one change of two notes would be $252*80*(30)^6*7$ or over 100 trillion! (...) This refinement puts our hypothesis on a reasonable statistical basis but undermines any hope for testing or practical use.
(Jaxitron , *Cybernetic Music,* 1985, 229)
394AIMA, 105.
395AIMA, 114.
396Rowe, *Interactive Music systems*,1993, 225.
397Rowe, *Interactive Music systems*,1993, 221.
Schwanauer, Stephan M., and Levitt, David A. (ed.), *Machine Models of Music,* Cambridge, MA: MIT Press, 1993.
398Downward and upward properties in solutions means that every abstract solution in the planning process can be decomposed into more primitive solutions, and that inconsistent abstract plans have no primitive solution.
399AIMA, 376.
400Rowe, *Interactive Music systems*,1993, 163.
401AIMA, 335.
402Rowe, *Interactive Music systems*,1993.
403Musical hermeneutics is held by Jacques Nattiez, Bowman, Wayne D., *Philospical Perspectives on Music* ,New York: Oxford University Press, 1998.
404Cope, David, *Virtual Music – Computer Synthesis of style*, Cambridge, MA: MIT Press, 2001, 92.
405It looks for phrases, beats, harmonies and so on, but only in a certain time frame, I called sonic now.
406Also called Emmy (Hofstaedter a.o.)
407Using cooking as a model we might discern the creation of a meal by varying an established recipee (Cope) and the free form cooking based on more general rules of the chemistry and tastes of food components (Rowe). Recipees stand here for styles while more general rules ensure successful improvisations
408More examples of similar systems with relevance for EMI are found in ch1 of Virtual music (David Cope, *Virtual Music,* 2001)
409Association nets are similar to neural nets, but they lack backpropagation and may have unlimited interconnections between nodes.
410Resistant as well, since local patterns are eliminated incrementally.
411Rules of EMI.
412We might imagine an alien believing humans preference for food to be plants, since he landed in a poor or progressive community. Projection of data (Goodman) and selection of data is vital to find deep or fundamental knowledge that in the case of humans energy supply would be fat and carbohydrates.
413After Hofstaedter in David Cope, *Virtual Music,* 2001,44.
414'Virtual' because the numbering of it as 10^{th} presuppose a logical continuation of a composers work in the clothes of EMI.
415David Cope, *Virtual Music,* 2001, 50.

416See also Leonard Meyer in Emotion and meaning (1956) for a more traditional analysis of expectation based structures.
417Leonard Meyer, Emotion and meaning in music in ch 9.4.
418Cope, David, *Virtual Music,* 2001,48-49.
419Ibid., 137.
420Styles that implicit favor certain instrumental flavours or sounds will therefore be poorly represented in EMI. Likewise will styles that depend on dynamic contours in performance have weak representations.
421Schwanauer and Levitt (ed.), *Machine Models of Music,* 1993, 405.
422David Cope, *Virtual Music,* 234.
423Ibid., 261.
424Ibid., 280.
425David Cope, *Virtual Music,* 315.
426Ibid., 216.
427Ibid. 310.
428Ibid., 312.
429Daniel Dennett, *Darwin's Dangerous Idea*, 1995.
430In this perspective Cages 4'33 is the exclusive inclusion of noises alltogether, whatever this might mean for its status as piece of art.
431Memes will be taken up again in ch 9 (9.6).
432Dennett, *Darwin's Dangerous Idea,* 288.
433Ibid. 72.
434Robert Rowe, *Interactive Music systems,* 1993, 240.
435David Cope, *Virtual Music,* 422.
436Ibid. 312.
437Rolf Inge Godøy, *Formalization and Epistemology,* 1993, 140-141.
438Gareth Loy, *Musimathics*, 2006, 403.
439Ibid.
440Ibid. 405.
441Such thoughts are very much in line with the problems of ethical nature in medicine e.g. where machine diagnosis happens to outperform human competence. Should we accept the machine diagnosis and on what terms?
442Rudi Cilibrasi, Paul Vitanyi, Ronald de Wolf, *Algorithmic Clustering of Music,* 2003.
443You tell me which pieces you like now, I will tell you which other pieces, unknown to you, you will like as well.
444We might call them todays undisputed leaders of data-mined knowledge about the popular tastes in overall society.
445Dynamic typing is typical for Lisp. This means that variables and structures do not have to be specified before compile-time
446This is related to ill-defined domains, see also in relation to the concept of creativity in 9.6.
447Artists often use their trained intuitions to formulate aesthetic proposals, that they test on themselves or others. The artists visual or aural feedback will generate new proposals until the artist contents himself with a result that is then called work (of art).
448We may think of the choosing of programming languages somehow like choosing a specific natural language or dialect etc. in certain situations. It is on the pragmatical level of finding effective tools, not on the semantical level of what can be expressed.
449Output from CM may be generating sounds in Csound, CommonLispMusic, Open Sound Control, SuperCollider, MIDI, Midishare or Portmidi or display them graphically as FOMUS notation, CommonMusic notation, Plotter (data visualization).
450CM is released under LLGPL (Lisp Lesser General Public License).
451All syntactic expressions in Lisp are expressions, therefore the liberty to use functions as arguments etc.
452In this example, Pluck-Poly is the C-sound instrument that generates “plucked” instrumental sounds with polyphonic resources.
453These restrictments of using chordal, i.e. simultanous sound events or even multitimbrality in one part, are probably rests of outdated limitations of behalf of machine architecture but are still part of the definitional apparatus of CM.
454PluckPoly that is used in the examples is part of the MusiKit of the now historical NeXT machine and has slots holding vales for the following parameters: amp, ampRel, bearing, decay, freq, keyNum, lowestFreq, pickNoise and sustain-related parameters.
455Item stream data types are items, notes, pitches, rhythms, degrees, intervals, steps, voicings, series and amplitudes.
456Patterns are cycle, sequence, heap, random, palindrome and function.
457In traditional notation a note will always have both length and pitch. There is no abstraction like in the item-stream notation of MC.
458Sound output modules can be MIDI-modules if soundscore is written in MIDI (as MIDI-file), the Music kit on a NeXT machine or user-defined virtual instruments on any machine with DSP if soundscore is the result of programming in the “Sound kit language” (CLM; CommonLispMusic)
459CommonMusicNotation is a subset of CommonMusic .
460Lisp constructs lists by cons and extracts lists by car. CM uses x**s** (item**s**) to construct streams and x (item) to extract

or read a stream element. X may stand for rhythm values, note values, number values without musical meaning and so on.

461 Such a rather unhistorical claim will be easily disputable. In a more philosophical perspective, that I find very much attractive, any tool will by its nature, necessarily favor certain and exclude certain other options and possibilities in favor of others etc. Nonetheless, it seeems reasonable to talk of a relative openess or relative generality of both MAX and CM in respect to other reference tools, known from the history of MC.

462 Common Lisp Object-oriented System. Library for CL to make object-oriented programming part of CL.

463 Superficial is here meant in a pure descriptive sense. Patterns or X in EMI are somehow more on a morphological level of structure, in contrast to Cypher and other systems that embody deep structural knowledge like generative or grammatical theories of meaning. (GTTM, 1983)

464 Cope numbers some of his synthetic style works in continuation of the original composers system. This is a kind of post-mortem continuation of a composers creative life in the lists and recursions of EMI.

465 Like Stockhausen, Boulez (Curtis Roads, CM tutorial, 1996, p830). This mindset contributed even to the New York school of 'open form music' where the performer is asked to contribute more to the compositional work than what is usual. It takes its model to some degree from jazz improvisational practice.

466 Fascination with such objects goes back to Albrecht Dürer in the visual arts and his “The Painter's manual” from 1525 and to Leibniz in mathematics and philosophy that found straight lines bo be defined recursively self-similar.

467 Other examples from wikipedia: clouds,snow flakes, mountains, river networks, broccoli and blood and pulmonary vessels in organisms.

468 See Daniel Dennett's “Darwin's Dangerous Idea”, 1995.

469 Like the mentioned Mandelbrot set, Burning ship, Lyapunov (escape-time fractals); Cantor set, Sierpinski carpet, Sierpinski gasket, Peano curve, Koch Snowflake, Harter-Heighway dragon curve, T-square (iterated functions fractals); Brownian motion, Brownian tree, Lévy flight, fractal landsscapes (random fractals) and many others.

470 This depends on the size of the problem as well. As we recall from 1.3, classes of problems in relation to computability will always depend on the size of the problem as well, not only its type,e.g. linear models or systems.

471 Additivity means satisfaction of the equation: $f(x + y) = f(x) + f(y)$; Homogeneity means satisfaction of the equation $f(\alpha x) = \alpha f(x)$

472 We may classify algorithms and even computational problems as natural by now.

473 Wikipedia:chaos theory.

474 Chapel, Ruben Hinojosa, *Realtime Algorithmic Music Systems from Fractals and Chaotic Functions: Towards an Active Musical Instrument* ,Doct.Diss., 2003.

475 Dabby, D, *Musical variations from a chaotic mapping,* Ph.D. Dissertation, Cambridge, MA: MIT Press, 1995

476 Maurer, John A., *A brief History of Algorithmic Composition, 1999.*

477 Rene Descartes (1596-1650): Les Passion de l'ame.

478 The baroque idea of 'elocutio' led to lists of devices that musical rhetors could use to delight and stir audiences. Examples of such devices adopted by compositonal theory are: Anaphor, parallelism, metaphor. Important to remember that these affects were not particular but generalized or abstract feelings that were com-posed into forms of music. They believed that the mind's responses to stimuli could be caracterized into objective and discrete states whose intentional arousal was the principle objective of music.

479 An overview and timeline of the history of music psychology is found at http://www.musiccog.ohio-state.edu/Music829F/timeline.html.

480 Leonard Meyer, *Music and meaning*, 1956/1989.

481 Diana Deutsch,*The psychology of music*, 1982.

482 GTTM, 1983.

483 DT2001, DT2007 expands this systems further with a probabilistic framework

484 Temperly believes the clearest antecedent for preferenc rules is to be found in the Gestalt rules of perception from the 1920s; see ch3 Temperly, 2001.

485 Temperly, David, *The cognition of basic musical structures,*MIT Press, 2001, 13-4.

486 Ibid., chapter 8.

487 Ibid., 291.

488 Temperly discusses differences in PR systems for what he calls common-practice music, rock and african music in several chapters going into considerable detail, and finds in ch11 that interesting differences may be conceptualized using the framework of Prs. (Temperly, *The cognition of basic musical structures,* 2001, 296-7).

489 Temperly, *The cognition of basic musical structures,* 2001, 299.

490 Temperly, *The cognition of basic musical structures,* 2001, 314. Figure 11.11. Musical examples are to be found in section 11.15.

491 Lerdahl describes tonal tension even more comprehensive than Temperly in his Tonal Pitch Space (Lerdahl, Fred, *Tonal Pitch Space,* Oxford: Oxford University Press, 2001.).

492 “The meaning of something in a piece of music is simply its function, its role in the effect and impact of the entire piece” (Temperly, *The cognition of basic musical structures,* 2001, 326)

493 Gareth Loy, *Musimathics,* 2006.

494 Temperly's recent book proposes computational models for perception of key and meter using Bayesian probabilistic methods and techniques, before expanding this approach to phrase and pattern perception, harmony and

musical styles a.o.

495 Clarke expands the theory of musical perception to encompass physical and social contexts in listening.

496 Huron sees musical expectation as an instance of human expectation in general, drawing on statistical learning and evolutionary theories and serving the goal of pleasure, mostly. Emotions evoked by expectations are related to five distinct response systems: defensive reaction responses, stress-inducing tension responses, rewarding prediction responses, imagination responses leading to deferred gratification and consciousness involving appraisal responses. Those different response mechanisms usuallly cooperate in generating complex feelings based on multiple expectation driven systems.

497 The authors of this volume aim at laying a foundation for a cognitive neuroscience of music, but concentrate on perception and performance without dedicating composition a central role.

498 Lerdahl extends the GTTM theories. It is a "multidimensional model of diatonic space that quantifies listeners' intuitions of the relative distances of pitches, chords, and keys from a given tonic. The model is employed to assign prolongational structure, represent paths through the space, and compute patterns of tension and attraction as musical events unfold, thereby providing a partial basis for understanding musical narration, expectation, and expression" (book description, Amazon.com)

499 The author contributes to "contemporary musicologist's toolkit", including from "computational musicology".

500 He shows how our ability to follow musical meter is simply a specific instance of our more general ability to synchonize our attention to regular recurring events in our environment. (from amazon)

501 Represents a start for en "evolutionary biomusicology" with search for musical universals. See also ch2

502 Application of humanistic principles to cognitive science and music (humanistic music theory)?

503 Cypher: synchronous multi-level analysis and synthesis for harmonic and rhythmic structures
EMI:detection of larger scale hierarchical structure (*unifications*) and pattern analysis for *superficial signatures* in the supplied examples.

504 Both GTTM, Rowe and Temperly are somehow working under common paradigm.

505 Especially lines from GTTM and statistical theories of expectation.

506 Xenakis, Iannis, 'Determinacy and indeterminacy'. *Organized Sound*, 1996.

507 Braitenberg Valentino, *Vehicles-Experiments in synthetic psychology,* Cambridge, MA: MIT Press, 1984,

508 Braitenberg, *Vehicles,* 1984, 19.

509 Other examples are flocks, herds, families, societies etc. "Swarm music" is discussed elsewhere

510 Beyls, Peter, "A molecular collision model of musical interaction",2005.

511 In more complex instances of this general model, particles may have more variables that even remember former actions (memory). Beyls writes: "Consider for instance, people meeting and interacting in a public space. Social behavioral patterns will surface spontaneously." (Beyls, *A molecular collision model,* 2005, 2) Likewise Beyls hopes to "synthesize families of waves of variable coherence and periodicity. The complexity of the waves follows a law and the user has only limited and indirect instrumental control over the quality of the wave propagation process."

512 Beyls, *A molecular collision model,* 2005:
The present paper formulates a generative system featuring native internal dynamics open to disturbance by unpredictable actions from an external user.

513 Beyls, *A molecular collision model,* 2005.

514 Dittrich, Ziegler & Banzhaf. Artificial chemistries, A review, 2001.

515 Prigogine, I., and Stengers, I., *Order out of chaos,* Bantam books, 1984.

516 Score following?

517 from Tim Blackwell's website.

518 Barton&Hugh (see below): "The beauty and harmony of the universe is the 'real' music and the job of musicians is to tune tinto some small part of that preexisting universal harmony and translate it into the more usual type of music, made with human voices and instruments. 'Music of the Genome' is one small part of that universal harmony that has existed within us for ages, yet has never before been turned into audible music."

519 Wired: DNA the way to San Jose? (from wired.com)

520 John Dunn.

521 Jarvelainen calls this the "sonification of other natural processes" (Jarvelainen, Hanna, *Algorithmic musical composition,* 2000).

522 AIMA: "it converges to a local optimal solution, and has been used with some success in a variety of applications. As with hill-climbing techniques [from AI search methodology], however there is no guarantee that it will find a global solution. (593) and discussion on (583-584).

523 Todd, *Musical Networks,* 1989.

524 Loy, *Musimathics,* 2006, 388; Loy reports about HARMONET that harmonizes chorale melodies in Bach-style. But since it is a hybrid model with both symbolic and connctionist modules, it is difficult to know how much of its succes can be subscribed to connectionist methods per se.

525 It is often seen as similar to reinforcement learning, a subfield of machine learning, because it has to rely on occasional rewards only, AIMA, 619.

526 Both on paper with pencil and rubber, and in the head with imagination and decisions/selections

527 In: Griffith, Niall, and Todd, Peter, *Musical Networks; Parallel distributed perception and performance ,* Cambridge, MA: MIT Press, 1999.

528Spector L. and Alpern A., Induction and recapitulation of deep musical structure; IJCAI Workshop on AI and Music, 1995.
529Todd, Peter M, and Werner, Gregory, M., *Frankensteinian Methods for Evolutionary Composition, i*n NG1999,11.
530Ibid.
531Ibid.,17.
532A speciation process that happens when both female preferences for particular traits and male traits themselves coevolve into new specieses that divide the original population into subpopulations of different traits and preferences.
533Ibid.,21.
534John Dewey called problems in his pragmatist philosophy for "indeterminate situations", a description that fits well in unified perspective on algorithms and artistic plans.
535Dennett, *Dangerious Idea*,1995.
536Hromkovič, Juraj, *Algorithmics for Hard Problems,* Berlin: Springer, 2001, 464.
537This is not the same as the different machine architectures discussed in ch2. Hromkovic means with parallel solution the application of different algorithmic strategies or techniques, i.e. a higher level than distributing data to compute functions on them.
538Ibid., 476-7.
539Ariza, Christopher, *An open Design for Computer-Aided Algorithmic Music Composition,* Dissertation, Amazon.com, 2005.
540Thomas Kuhn's analysis of science applies here to MC as well, where normal, stable and conventional activities are compared to the revolutionary, unsettled and contrast-rich activities of young, not yet "normalized" practices.
541S-expressions (symbolic expressions), i.e. elements are delimited syntactically by parantheses only for all types and instances. Thus, such a regular syntax alouds metaprogramming.
542"Lisp is a programmable programming language", citation attributed to John Fodaro
543[maxlisp] is designating the particular CLips encapsulating *object.*
544Brad Garton; chuck.cs.princeton.edu. Chuck is an audio programming languag for real-time synthesis, composition and performance. It is strongly-timed in a concurrent programming model.
545Brad Garton; www.rtcmix.org/
546ELIZA is a classic example of rule-based expertise (of early AI) that may fool some humans to believe in its humanness, only on rather superficial and cliché-like conversational abilities.
547Andrew Benson, *Making connections,* internet: www.cycling74.com.
548Ibid.
549Peter Beyl, *A molecular collision model of musical interaction,* internet, 2005.
550Brad Garton, *Multi-language Max/MSP,* 2007, internet:www.cycling74.com.
551It is therefor therefore limited to only Mac OSX. But he is confident that many of these tools will be prepared for other platforms in time.(confirmed in e-mail from Brad Garton).
552Brad Garton, *Multi-language Max/MSP,* 2007, internet:www.cycling74.com.
553Arne Eigenfeldt, *Managing Complex Patches in Max,* 2007, internet: www.cycling74.com.
554Brad Garton, *Cooperating Computer Music Languages,*2007, internet: www.cycling74.com.
555This is valid even for Turing-Machine-languages such as Lisp, since they steer (not coerce!) users into certain directions.
556I feel reasonably confident in making the assumption that both the Feyerabend project **[1.8]** and Flanagan and Holloway **[1.8]** would applaud these attitudes and programmative values manifested in Garton's activities without much restraint.
557Brad Garton, *Cooperating Computer Music Languages,*2007, internet: www.cycling74.com.
558Christopher Ariza, *Navigating the Landscape of Computer Aided Algorithmic Composition Systems: A Definition, Seven Descriptors, and a Lexicon of Systems and Research,* internet: algorithmic.net.
559Computer Aided Algorithmic Composition (Ariza).
560Ibid. Ariza calls MC for CAAC, standing for Computer Aided Algorithmic Composition Systems. I found this definition and some other proposals to restricting and narrowly defining and ended up with Machine Composition, interpreted as liberal or including as possible.
561Excluding 'M' perhaps.
562Assuming the definition of music or better musicianship in the sense of generating rich meaningful relations sense, as formulated in 2.2 as musical learning cycles.
563Listeners thought to be at least a significant portion of general music life, and certainly not limited to small creating circles of similar sound-worlds.
564Many of these "experimental" compositions were placing themselves in the micro-sound (Roads, 2.5) dimension of music or in the direction of timbral investations typical of these decades' interests.
565"Find a style" has implications of discovery other than invention. We have no space to engage on this line of thought here, but is an illumating symptom about the way we look at arts and sciences.
566These systems are described already in 5.7.
567CommonPracticeNotation (ch 2.4).
568'Computer music' in my restricted understanding was very much concerned with timbral or tone qualitative shaping

and processing (see Csound, ch6 e.g.). The last century in whole (from Wagner, Mahler to synth-pop and hip-hop) may be termed the "*timbral century*" in music history, since many styles and investigations were in important ways tied to the manipulative possibilities of 'imprinting' technologies to *control sound quality* , and therefore these projects often overlap with com-positional intentions. Nonetheless, I think it is effective to limit the com-positional domain to the macro-sound and its combinations (in the definition of Roads, ch2.5).

569This may be a relevant application of this term, that has been explored already in 2.5.

570Language, batch, or interactive user environments

571More examples can be found in Ariza, above and Papadoupoulos/Wiggins below.
For historical systems see John A Maurer, *"A Brief History of Algorithmic Composition"*, 1999.

572Sequenced methods are more timing cues, and responding to certain patterns or sequences, see Rowe, ch4.

573Generative theory of Tonal Music. See ch4.

574This high level organisation was first discovered by music theorist Schenker, mentioned in ch2.

575Possibly both are called 'players', but that would essentially raise this the performance definition higher in complexity terms than we have done consistently in this paper.

576AIMA, 573. A perceptron is an idealized single-layer, feed forward network.

577Gradus ad Parnassum is the title of a most influential teaching treatise of Johan Fux (1725), where levels of learning counterpoint, called 'species' consist of rules grouped into ascending species' of growing difficulty translating into incrementing style proficiency on the part of the student. An interesting experiment might be to create a reverse-engineering experiment where a learning MCs extracts the Fux-rules progressing by increasing numbers of examples and of increasing counterpointal complexity.
Mont Parnassus was the home of the Muses according to Greek Mythology.

578A Fibonacci recursive algorithm may be generated from this formal function: $f_n = f_{n-1} + f_{n-2}$, $f_1=f_2=1$. It produces numbers, known as the Fibonacci sequence 1,2,3,5,8,13,21,...with initial condition=2.

579The last species is called the florid, with connotations of beauty and fluidity in style.

580This is the exposure to a rich history of environments/experience. It is also related to muliple caused creativity. More on this later.

581"Emmy" (EMI) does consistently make listeners, even of experienced levels, err in distinguishing EMI-generated from original pieces in a certain idiomatic style (ch5, 9.3)

582Bernard Bel, "Portrait of an Extra-Terrestrian", in Tabor, Jerry (ed.): *Navigating musical Horizons,* Westport, Conn.: Greenwood Press. 1999, 81-91.

583e.g. finding the next scale tone given a certain scale.

584A very much probationary illustration of such ideas is supplied as appendix.

585Studies of "Donaldism" stress the innovationism inherent to many of his actions. Even prolungated periods with doing nothing (suspense), may be interpreted as his "intuitional pump" generating practical creativity. His sceptic attitude towards social conventions and the working place is further witnessing to the artistic-like vein in his personality.

586Using a magic inventor *hat* when he need turbocreativity to solve the most difficult cases. This hat is a nice metaphor to describe thinking machines (computers) that help us to do some number crunching etc.

587Leonardo da Vinci, prototypical inventor of all times.

588See ch2.3 and 2.6 about conceptual spaces (Gärdenfors).

589George Papadoupoulos, Geraint Wiggins, *AI Methods for Algorithmic Composition: A Survey, a Critical View and Future Prospects,* 1999. internet: www.doc.gold.ac.uk/~goguen/pps/mq.pdf.
They categorize MCs "on their most prominent feature": Mathematical Models, Knowledge based systems, Grammars, Evolutionary methods, Systems which learn, and Hybrid systems.

590mathematics, psychology, physics and biology

591The kinesthetic theory of rhythm states states that there is a connection between restricted movement of physical handicapped and their limited rhythmic abilities. (see Moog, 1978).

592Musical robots will eventually prove this wrong, but this does certainly not infere with this papers fundamental attitudes.

593The most common-sensical understanding lies in the examples of "redness". Redness is a prototypical qualia.

594See also Peter Kivy's viewpoint on musical meaning.

595A sense of self over time is simply the conservation of the accumulating knowledge of its environments, i.e. the behavior of its co-player/co-composers. As long as Cypher's memory is not flashed, his sense of self is pretty much stable over time.

596Wikipedia: Person.

597Interesting philosophical puzzles are illustrated in these cases. Are different stages and therefore settings and preferences of the *same* Cypher, different persons or the same person in different stages of his life (based on causal continuity)? Similar problems were encountered by David Hume.

598Dan Lloyd, *Simple minds,* MIT press, 1989.

599Ibid., 182, 231.

600The point here is not to liken chess with music as an activitiy, but to remind of the fact that many not believed this to be possible before Deep Blue proved them wrong.

601This applies to many mathematical and physical models of MCs. (ch7)

602Godøy states that “a non-recognition of the unique organizational principles manifest only at higher levels of any particular objet or phenomenon” is his statement of non-belief in productive low-level discretization of musical events. Godøy, *Formalization and Discretization,* 1993, 45.
603from Republic, cited in Bowman, Wayne D., *Philospical Perspectives on Music* ,New York: Oxford University Press, 1998, 40-41.
604Peter Kivy, *Introduction to Philosophy of Music,*(Oxford: OUP, 2002).
605Ibid.,92.
606Ibid.,92.
607Jacques Attali, Noise: The political Economy of Music, 1985 (translated).
608Dennett, *Dangerious Idea,* 1995.
609Ibid.
610Bowman, Wayne D., *Philospical Perspectives on Music* ,New York: Oxford University Press, 1998, 251.
611Wikipedia: Aesthetics.
612Mozart, Debussy, Bruckner, “Emmy” (EMI, programmed by David Cope, ch5)
613Wikipedia: Aesthetics.
614Robert W. Weisberg, *Creativity, Genius and other myths,* (New York: Freeman, 2003).
615What I like to call 'psycho-plaining'
616Ibid., 3.
617Churchland Paul, Matter and Consciousness.
618Weisberg, *Creativity*, 1986, 128-130.
619Effective means here to actually solve problems.
620i.e. the formation of a mental image or view of things not present.
621Ibid. 142.
622Ibid. 142
623Related to insight, a concept we have to relegate due to limited space.
624Gr. in-tuitere: looking into one-self. Or in modern cognitive-like terms, “pushing”one's neurons to come up with better states.
625Verklärte Nacht is known to be one of the most progression-dissolving pieces in tonal music tradition.'
626Methodologal device used in Daniel Dennett in “Quining Qualia” to rearrange conceptual order of common-sense.
627In logical lingo (after Carnap) this means to ask external questions related to some system or practice.
628Having said that, I do not believe a machine may do this re-evaluating of premises AND in the same time decide on Cages title or label “4:33”. But neither would another human composer , that might have stumbled upon the “same” idea of a radical reorientation of music culture, e.g. towards indeterministic nature of events. An important feature that machines will have to incorporate, to be able to make such intellectual work. It is a model of motivation (or what some might call free will, see Dennett's *Freedom evolves).*
629This might also be described as a loosening up of neurons and their networks. In its worst form, electro-shock therapy, in-spiration of electrical currents from the outside, are thought to rewire the neurons to make an individual more normal and adapted. In its lesser drastic form, intoxination of either drugs like LSD or alcool will do similar re-wiring or loosening up of neuronal connections to aloud for new and exciting ideas on behalf of artists and wannabe-artists.
630Sometimes inpiration is used ambiguosly for both sources, but in-spirare – breathing in (latin) - indicates this in-out-distinction as archetypical.
631This is the well-known “Humean moral” in relation to psychological terminology.
632Similar to the pervasive problems with indeterminism in physics, that by some are taken as an opportunity to breathe life into ghosts or geniuses outside of the quantum world as well. We remind of Penrose in this question (ch1).
633Once again, creative processes are in this theses explicated as combinations and recombinations of existing elements. But, we also claim that personality architectures are not necessary ingredients of AI-based technology.
634Following this picture somehow, the function f(music)= body + imagination + sound, could be split in f(music)= sound without exiting music's definitional domain. This is what I propose to call the *disjointability* of music.
635Some movements may rest as relics.
636More on greedy methodologies in 9.5/9.8.
637See Gärdenfors' conceptual spaces in ch2.
638Nelson Goodman, *Languages of Art,* 2.ed.*,* Indiana: Hacking, 1975; Reconceptions of philosphy a.o.
639Absolutist views are most famous for the historical disputes in late romanticism about programme music's legitimacy. (Hanslick).
640Aristoxenus (4cent.BC) argued for absolutist and non-referentialist views, Plato made a case against music because of its perceived referentialist nature. (see Bowman).
641 Leonard Meyer, *Emotion and meaning in music,* 1956*, 45.*
642Ibid., 35
643Ibid., 32
644Ibid., 14
645Ibid., 27

646Ibid., 44
647David Temperly in his *Music and Probability* (2007) characterizes Meyer's theory as his major predecessor.
648LA: Languages of Art, 2.ed 1975.
649Nominalism: denial of the reality of essentialist meanings (such as intensional properties) and objective realities.
650Related to his *Fact, Fiction and Forecast*, where he interprets scientific categories as conventionlized as well.
651This is Goodman's way of speaking of something that functions like a denotational object, without that he has to admit any ontological defeats. His nominalist position is in disaccord with any firmly gluing presuppositions about "reality's" features. Features are labels, but their denotation are referance *fields* only.
652Disjointness and finitely differentiation are features of symbols and symbol systems, hence they are meta-features. They serve to reward systems that have logically appealing and rewarding traits related to precision and explicatibility.
653Goodman, Nelson, *Languages of Art,* Indianapolis: Hackett, 1975, 69.
654Stanford Encyclopedia of Philosophy: *Goodman's Aesthetics,* internet: *plato.stanford.edu/entries/goodman-aesthetics/*
655Brenda Mergel, *Instructional Design & Learning theory,* 1998. Mentions with this citation: Schuman, L., *Perspectives on instruction,* 1996. internet: *edweb.sdsu.edu/courses/edtec540/Perspectives/Perspectives.html*
656Another prominent music philosopher, Peter Kivy (9.2:Q5) is defending an enhanced version of formalism. Music is according to Kivy without conten. Emotions are possessed, but do *not* denote (Goodman!) emotions. This emotive cognitivism ("Emotions are not felt, but cognised", (Kivy,*Introduction,*2002,109) and "The emotive tone, so to speak comes with the territory , comes along for the ride".
657Hence, something both 'allographic' and weakly 'autographic' in the same time, following Goodman.
658This is in analogy to Dennett's definition of "greedy reductionists".
659Gareth Loy, *Musimathics,* 347
660Children's songs for falling asleep.
661Loy, *Musimathics,*348.
662Leonard Meyer, *Emotion and Meaning in music,*1956.
663Variation might be described as the result of a linear function of a pattern or repetition. Contrast in this sense would the result a non-linear function of a pattern or repetition. But this is somehow playing with metaphorical means.
664Non-deterministic means here from the perspective of the composer. A listener will always experience non-deterministic based on subjective knowledge. A composer will always, as I want to argue for, choose the best solution (a deterministic *felt* solution) and not deferring decisions to objectively non-deterministic methods like playing dices, that in turn generate "pockets of anti-structure". This may be an aesthetical statement of formalist predicament and Western biased as well; Buddhist or other cultures idea may be more concessive to objective indetermism in sounds.
665This is the point at which I want to state a philosophical argument in favor of determinism. Non-deterministic methods are, as I see it, "cheep" ways to generate contrast. Well-planned contrast will create the unpredictable responses and feelings of listeners, but may still become in later or other ways part a web of meaning relating it to the non-contrasting materials. In other words, listeners that think they perceive chaotic structures, may restructure or reconceptualize in certain ways to make sense of differentness as well. With non-determinist contrast, a listener will, somehow like Cage's philosophical intentions, feel a more thoroughly break-down of expectationalist norms. This *may* be an aesthetical intention, and not only a way to produce materials in convenient modes.
666Loy, *Musimathics,*350.
667This could be misinterpreted as a statement for aesthetical conservatism. It is rather, like I see it, a re-definition of creational processes. It doesn't interfere with the dynamics or speed of artistic change or recombinational rate relative to time. It *is* the claim that any change is best interpreted in constructivistic terms and not in romanticist ideas of the individual transcending his culture by consciously *breaking* existing rules.
668In case a person refuses to play for practical reasons, the act is non-aesthetical and non-relevant here.
669See also the example of a Fux' Gradus dilemma described in Loy, *Musimathics*, 362.
670I.e. behind and besides the most prominent or attention-claiming sonic events. This is called masking in psycho-physical theories and is a valid principle for all informational channels of perception.
671Masking is the aural phenomenon of shadow. In short, when sharp light masks the information from the dark area behind, masking takes place. There are many known cases of acoustical masking, described in Stephen Handel, *Listening , An Introduction to the Perception of Auditory Events*,(MIT Press, 1989).
672Loy, *Musimathics,*350.
673Aural sensibilities is a concept taken from Gareth Loy, *Musimathics.*
674Catherine Elgin in Goodman/Elgin, *Reconstructions in philosophy & Other Arts and Sciences,* Indianapolis: Hackett, 1988, 24/26.
675This parallels my pedagogical approach in ch3 in positive and negative patches.
676"The organization of sound terms into a system of probability relationships ... are all common characteristics of musical language." (Meyer, EM, 63).
677Temperly, David, *Music and Probability,* 2007,207.
678This is a daring statement, Goodman writes that even symbol systems that do not satisfy *any* of these symptoms may nevertheless function aesthetically. He calls them also clues that indicate but without guarantees. I assume here,

nonetheless, that the more of these symptoms are satisfied, the higher will the probability be that their practical use reflects these aesthetical symptoms. The relation between listening and composing in relation to a sphere is relevant, but must be put off to a later occasion.

679Goodman's major constructivist approach to formulate a logical and nominalistic system of structures of appearances.

680"Machinist" is the operator of a machine, here used metaphorically denoting.

681Much lies in this understanding of the domain or "universe of discourse" (Carnap, 1954). I here think of it as the set of sounds and processes that manipulate them is possible to think of. This is problematic, but only necessary to accept as an intuitive idea to span the higher levels around. For now.

682Another example are the tools in the preliminary dimension between tool and agent in 4.2.

683Cited from Hans Moravec, *Robot*: *Mere Machine to Transcendent Mind*, Oxford: Oxford U Press, 2000. Citated in Selmer Bringsjord, *Is it possible to build Dramatically Compelling Interactive Digital Entertainment?,* intenet: www.gamestudios.org/0101/bringsjord/

684John Searle, in his famous and unfamous 'Chinese room' thought experiment as argument against the possbility in principle of 'strong AI'

685Or whatever it may become or may be called when cognitive and eliminative sciences of mind and brain have reordered our conceptual garden in this notorious engine for philosophical paradoxes and speculations.

686Kevin Warwick, *The march of the machines,* 1997. Warwik thinks the domination of AI machines is the inevitable outcome of future history.

687Loy, *Musicmathics,* 403.

688Ch7: evolutionary systems with different levels *and* stages of selection.

689Highest level in a hierarchy of critics.

690Aesthetical judgements are today the area of sustainely disputed human opinion. Is it possible to think of machines taking over this area of dynamical expertise? I actually think so.

691Logically speaking, the model is *an interpretation* of the formal and empty, non-interpreted sets and relations. See Carnap a.o.

692The attentive reader may reognize somehow disturbing or entertaining peculiar consequences of this model. It is not proposed as a material model of their domains, but only as an indicative way to depict the indeterminate *situation* in our argument.

693John Bowers, *Improvising Machines*, in Advanced Resarch in Aesthetics in the Digital Arts, no.4, may 2003.

694Ibid., 28

695Ibid.,29,41.

696Ibid., 43.

697*Loy, Musicmathics,* 407.

698Probably right from the beginnings of investigations of musicology, i.e. from around 2000 BCE.

699This is paralled by Goodmans first two symptoms of artistic symbol systems requiring density of both syntactic and semantic fields of reference. This is somehow like complexity or better fine-graining resoluted domains.

700Scharfstein finds a number of artistic expressions common to all kinds of cultures of humans and non-humans: spontaneity, tradition, sense of self, oscillations between extremes, social fusion among others.

701Ben-Ami Scharfstein, *Of Birds, Beasts, and Other Artists*, (New York, NY UP, 1988), 47-48.

702Examples are many. The ability of fish to feel pain (or possibly worms as well) is just one of the many myths of such denials. I myself have observed a woodpecker that used public lamp covers as a resonnating body, and testing three different lamp covers in sequence (50m apart) for their "nice"? drumming sounds? This is certainly only pure speculation, but so is the other hypothesis that they just do it for finding a tree to make a hole into. In lack of a better explanation, it illustrates our tendency to be sceptical to such accounts that takes birds seriously.

703Raymond Kurzweil, *The Age of Intelligent Machines,* 1990. Chapter 9, The science of art (page 6 from internet).

704Dennett, *Dangerous Idea*, 1995. Ch3.

705One may ask rethorically, who would Bach, and hence Brahms, Schönberg, GTTM or all the other composers have been without their systematic and algorithmic cranes of Western musical evolutionary history.

706Dennett, DDI, ch8.

707Reverse engineering is really the art of learning to compose; especially in the last five centuries, composers learned induction from their model composers' works, studying them diligently and copying their traits (memes, signatures).

708See ch5, Cope's EMI use of signatures. This reverse engineering is effective because they most often use model composers that they believe to be effective models for their own public's tastes.

709"David Cope can no more claim to be the composer of EMI's symphonies and motets and arts songs than Murray Compbell can claim to have beaten Kasparov in Chess.", Daniel Dennett, *In Darwin's Wake, Where am I,* 2000, 6. internet: http://ase:tufts.edu/cogstud/papers/apapresadd.htm.

710Ibid.

711Joseph Goguen and Ryoko Goguen, *Time, Structure and Emotion in Music,* 2003. ch2, ch5.

712Richard Dawkins, *The selfish Gene,* Oxford: Oxford University Press, 1976, 206.

713Public toilets, especially at universities, are typically meme fabrics, i.e. inscriptions are presented and recombined, in tagging-like fashions. One I remember well is : "You are all loosers, you have to dy!", which some meme-combining human changed to "You are all **idol**-loosers, you have to (be) **baptised**! [norw.: dø**pes**]. In this moment

this meme is transcending from person A to B to C (that's me) to D (you reading this!).

714 Richard Dawkins, *The selfish Gene,* Oxford: Oxford University Press, 1976, 206.

715 Daniel Dennett, *Freedom evolve,* Viking Adult, 2004.

716 Selection is called 'universal acid' by Dennett. See above.

717 Beethovens 5th symphony's most propagated motive or in our today's terms, most *famous* (or known) motive.

718 Think of our tendency to pick up intro's from tv-soap-operas that eventually may be embarassing to repeat (replicate), thereby telling others that we have seen (and immersively accepted) memes with a negative social value.

719 Filters are for example: this book has this editor, so I don't buy it, or this music is liked by my awful neighbor, so no way!

720 Daniel Dennett, *Memes and the Exploitation of Imagination,* Journal of Aesthetics and Art Criticism, 48, 127-35, Spring 1990, 9.

721 Ibid.,10.

722 Neurath's boat is a famous expression of the opionions, especially in logical positivism and analytical philosophy, that there is no Cartesian and doubtfree fundament, we can use to build scientific bricks from bottom to top, but that we have to revise scientific (and artist) truths under sailing in Neurath's boat. (Otto Neurath was the proposing vector of this effective meme!)

723 Eventually, genotype effecting recombination or even success-based criteria related to positive selection.

724 Holland John H., *Hidden Order,* New York: Basic books, 1995, 6-39. I remain unsure about the placed categories, since Holland makes a very much intertwined model, but leave the revision of this proposal to a later time.

725 Sipper, Moshe, *Machine Nature,* New York: McGraw-Hill, 2002.

726 Ibid.

727 Seething is pseudo random, therefore constraint variation.

728 Schuman, 1996??. Behaviorism: behavioral patterns are repeated until automatized. Cognitivism: learning the mechanisms of the world (problem solving). Constructivism: building own perspectives and symbol systems through experience and schema, preparing learners and cognitizers to solve problems in ambigious or creativity demanding situations.

729 Or any other counterpoint developing Fux or Palestrina style into some more complex and creative than rule-governed harmony alone. Seeger's "dissonant counterpoint is an interesting example of changing the rules by "multiplying with (-1)". In a similar fashion to "4:33" of Cage, Seeger, out of pedagogical motivations designed the anti-rules of Fux' Gradus. None of them breaks rules, but applies them in creative new ways. Whether it is effective and hence creative in ill-defined domains remains to be answered elsewhere.

730 There are varieties of forms in popular music, I just present one of the prototypes fitting well in here. Pop songs that use more repetition than variation or contrast, do that often in order to avoid to interfere with the textual and dominant information mode of many songs. In those songs contrast and variation takes place *in* the text, while music stands for repetition mostly. See also guitar singers, what I would call for poetry first of all.

731 Compliment of the sets in the domain.

732 This could also be described as complex logical statements, ie. longer statements or recombining statements.

733 Other systems with such a structure are cybernetics and many other dynamical tripartions in biology etc.

734 Symptoms is used by Goodman for the specification of artistic versus non-artistic symbol systems. But we may follow his conceptual choice here, hoping not to cause confusion.

735 Again, much of 'computer music' should here be excluded from the aesthetic domain or spheres altogether.

736 See dimension C-4 in 9.1.

737 String type virtual controller of gestural type of instrument.

738 Don Nichola Vicentino, invented archicembalo, with six keyboards, thirtyone steps to an octave.

739 Athanasius Kirchner (1600s) MusurgiaUniversalis, a mechanical device that composed music. He used number and arithmetic-number relationships to represent scale, rhythm, and temp relations, he called Arca Musarithmica.

740 Or the Helper of Petter Smart that guides him in his troubling moments (no comparison intended elsewhere!).

741 We hereby do not mean the inexactitude of the field (AA) itself, but its own perception of approximatization of comparisons of algorithms. AA does compare the effectiveness etc. and does not measure it with exactitude.

742 Wolfram's theory is built on automata with a simple rule with special status. It is called 110 and decides the transition of steps or transitions of states. Wolfram analyzed 256 elementary CAs and other more complex CAs and found that they could be categorized in four classes (see table above). Paul Reiners, *Cellular automata and music,* 2004.

743 As well as hosting rebels from rivaling France.

744 Wikipedia:Gluttony. Paepropere, Laute, Nimis, Ardenter, Studiose.

745 In the words of Friedrich that composed his eulogy after Juliens sudden death: "LaMettrie was born with a fund of natural and inexhaustible gaiety, he head a quick mind, and such a fertile imagination that it made flowers grow in the field of medicine." Frank Magill, *World Philosophy,* Volume 3, New Jersey: Salem Press, 1030.
With such qualities he would have liked, I feel confident to speculate, to explore the compositional pleasures of *quick* machines and their fractal-like *imaginations*. But this is, again, the wildest speculation really.

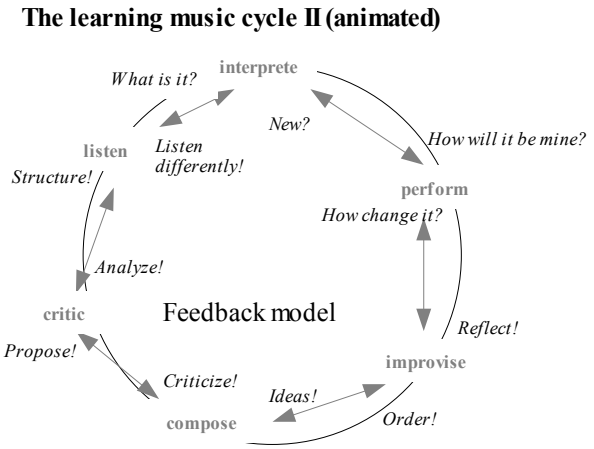
746 Ibid, 1024. "Only the physicians have a right to speak **on this subject.** Theologians have the least right.

Appendix

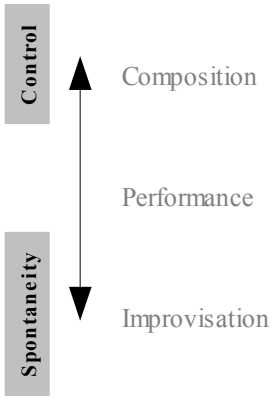
Figures and graphs

1. The learning music cycle II (animated)
2. Sponaneity vs Control

A.1 The learning music cycle (animated)



A 2.Spontaneity vs. control



	Patterns P()	**Frames F()**	**Patterns P (f)**	**Variator V(p)**	**Variator² V(v), V(v,p)**	**Contrastor C(p), evt. C(f)**
Lisp	List	Template	Template	Function	2.order function	Function
Input	Data	Data	Data	Data	Function	Data
Musical objects	Note sequences	Scales, chords	Scales, chords	Variations of Note sequences, scale types , chord types		
M Creator						
M Selector						
H Creator						
H Selector						

	Contrastor² C(c)	**Rule-creator R(r1...rn)**	**Form-creator F(f1...fn)**	**Style-creator**		
Lisp	2.order function	Functions	Functions	functions & lists		
Input						
Musical objects						
M Creator						
M Selector						
H Creator						
H Selector						

Machine Composition – Bibliography

Ariza, Christopher, *An open Design for Computer-Aided Algorithmic Music Composition,* Dissertation, Amazon.com, 2005.

Balaban, Mira a.o (ed.), *Understanding Music with AI: Perspectives on Music Cognition,* AAAI Press 1992.

Bernstein, Robert Scott., *Music, Creativity and Scientific Thinking,* Leonardo 34 nr.1,2001,63-68.

Beyls, Peter, "A molecular collision model of musical interaction", in Celestino Soddu, (ed.), *Proceedings of the 8th International Conference on Generative Art* (GA2005), Milan, 2005, 375–386.

Bidlack, R, *Music from Chaos: Nonlinear Dynamical systems as generators of musical material,* PhD Dissertation – San Diego: University of California (UCSD), 1990.

Blackwell, P. & Bentley,P, "Improvised Music with Swarms", *in Proceedings of the 2002 Congress on Evolutionary Computation,* volume 2, Piscataway, NJ:IEEE Press, 2002, 1462–1467.

Boden, Margaret A., *The creative mind: myths and mechanisms*,London: Routledge, 2004.

Bowman, Wayne D., *Philospical Perspectives on Music* ,New York: Oxford University Press, 1998.

Braitenberg Valentino, *Vehicles-Experiments in synthetic psychology,* Cambridge, MA: MIT Press, 1984.

Burns, Kristine H., *Algorithmic Composition ,* Ph.D. Dissertation, Ball State University, 1993.

Chapel, Ruben Hinojosa, *Realtime Algorithmic Music Systems from Fractals and Chaotic Functions: Towards an Active Musical Instrument* ,Doct.Diss., 2003, internet: *www.iua.upf.es/mtg/publications/DEA-2003-RubenHinojosa.pdf.*

Cook, Nicholas, *A guide to music analysis,* Oxford: Oxford University Press, 1987/1994.

Cope, David, *Computers and Musical Style,*E-R Editions 1991.

Cope, David, *Virtual Music – Computer Synthesis of style*, Cambridge, MA: MIT Press, 2001.

Crocker, Richard, *A History of Musical Style* ,McGraw-Hill, 1966.

Dabby, D., *Musical variations from a chaotic mapping,* Ph.D. Dissertation, Cambridge, MA: MIT Press, 1995.

Dennett, Daniel C., *Breaking the spell: Religion as a Natural Phenomenon,* Viking Adult, 2006.

Dennett, Daniel C., *Consciousness explained,* Cambridge, MA: MIT Press, 1981.

Dennett, Daniel C., *Freedom evolves,*Viking Adult, 2004.

Dennett, Daniel C., *Darwin's Dangerous Idea: Evolution and the meaning of life,* Gardners, 1996.

Dewey, John, *Art as experience,* new edition 1980, New York: Penguin, 1934.

Dobrian, Christopher, *Music and Artificial Intelligence,*1993, internet: http://music.arts.uci.edu/dobrian/CD.music.ai.htm.

Dobrian, Christopher, *Algorithmic Generation of Temporal Forms: Hierarchical Organization of Stasis and Transition,* 1993, internet: http://music.arts.uci.edu/dobrian/ICMC.Form.htm.

Dunbar, Robin I.M, "The Social Brain Hypothesis", *Evolution of the social brain. Science, 302(5648), 1160-1161,* internet: Anthropology.http://www.liv.ac.uk/evolpsyc/Evol_Anthrop_6.pdf.

Ebcioglu, K., *An expert system for Harmonizing Four-part Chorales,* Computer Music Journal, 12(3):43-51, 1988.

Eigenfeldt, Arne, *Managing Complex Patches in Max,* internet: *www.cyclying74.com.*

Elsea, Peter, *Fuzzy Logic and Musical Decisions,* 1995, internet: http://arts.ucsc.edu/ems/music/research/FuzzyLogicTutor/FuzzyTut.html.

Feyerabend Paul, *Against Method,* 1975.

Flanagin,G.K. & Holloway, A., *Patterns of creativity and Composition in Software Development, Music and Film,* OOPSLA, www, 2002.

Garton, Brad, *MaxLisp encapsulates CLISP within Max/MSP real-time music environment ,* internet: *h*ttp://music.columbia.edu/~brad/maxlisp/.

Garton, Brad, *Multi-language Max/MSP,* internet: *www.cyclying74.com*.

Garton, Brad, *Cooperating Computer Music Languages,* internet: *www.cyclying74.com* .

Gärdenfors, Peter, *How Homo bacame Sapiens – On the evolution of thinking,* Oxford: Oxford University Press, 2003.

Gärdenfors, Peter, *Conceptual Spaces – The Geometry of Thought,* Cambridge, MA: MIT Press, 2000.

Gärdenfors P. and Williams, *Reasoning about Categories in Conceptual Spaces,* 2001, internet: http://www.lucs.lu.se/people/Peter.Gardenfors/Articles/IJCAI2001ConceptualSpaces.pdf.

Goodman, Nelson, *Languages of Art,* Indianapolis: Hackett, 1975.

Goodman, Nelson and Elgin Cathrine Z., *Reconstructions in philosophy & Other Arts and Sciences.,*Indianapolis: Hackett, 1988.

Griffith, Niall, and Todd, Peter, *Musical Networks; Parallel distributed perception and performance ,* Cambridge, MA: MIT Press, 1999.

Griffiths, Paul, *The string quartet,*Thames and Hudson, 1983.

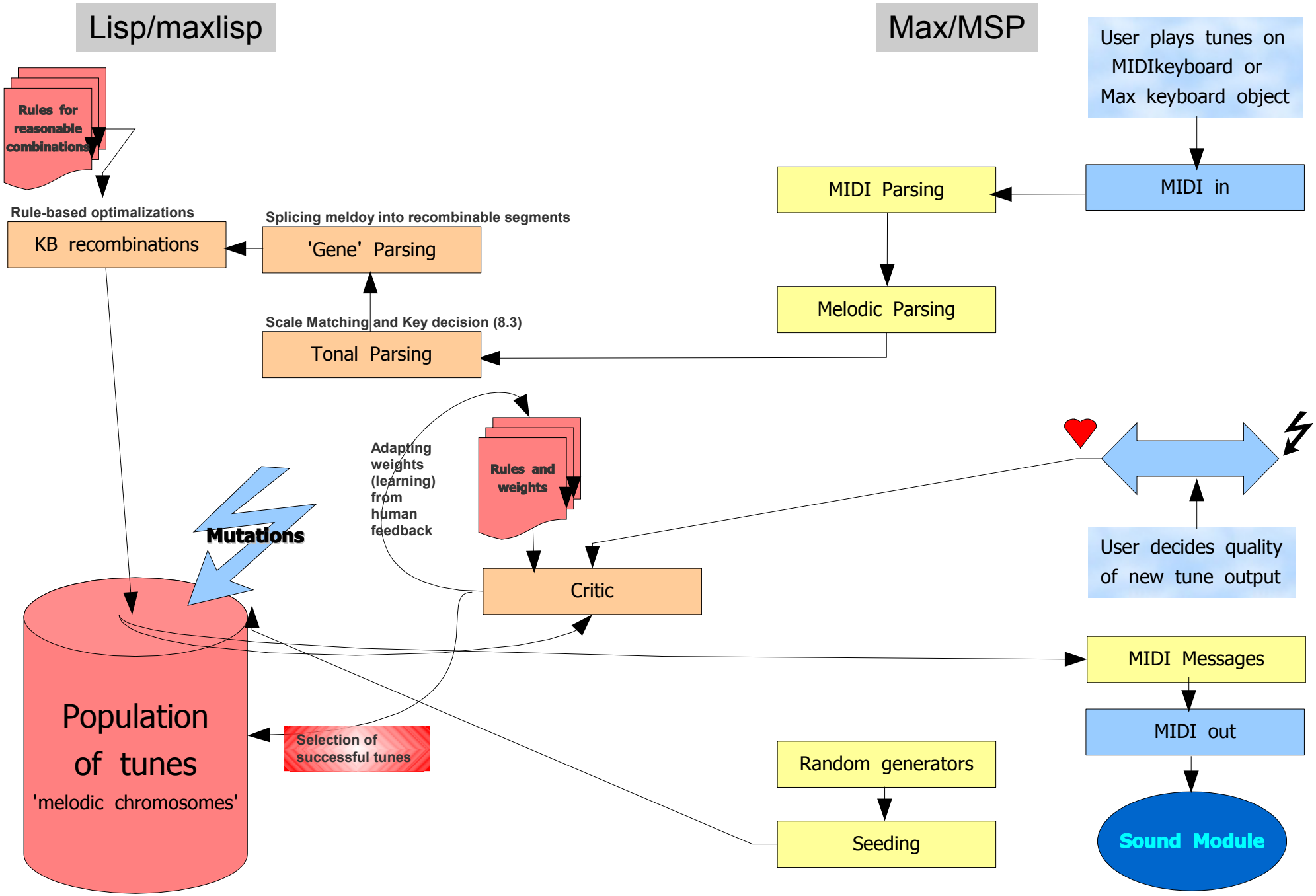
Grinde, Nils, and Nielsen Ludvig, *Lærebok i kontrapunkt etter Bach-stilen – tostemmig invensjon,* Oslo: Universitetsforlaget, 1966.

Handel, Stephen, *Listening, An introduction to the perception of auditory events,* Cambridge, MA: MIT Press, 1989.

Haning R. Barbara, *Concise history of western music (based on Grout/Palisca),*Norton, 1998.

Harvard Dictionary, *Concise dictionary of music and musicians,* HUP,1999.
Hauser, M. and McDermot, J, *The evolution of the music faculty: a comparative perspective,* Nature Neuroscience 6, 663-668.
Hofstadter, Douglas, *Gödel, Escher, Bach,*Vintage books, 1980.
Holland John H., *Hidden Order,* New York: Basic books, 1995.
Holst, Finn, *Conceptual integration in the domain of music ,*Conference:The Way we think/Conceptual integration network theory. ed.Hougaard, Lund 2002.
Hromkovič, Juraj, *Algorithmics for Hard Problems,* Berlin: Springer, 2001.
Howat, Roy, *Debussy, Ravel, Bartók: Towards some new concepts of form,*Music&Letters, Vol.58, No.3 (pp.285-293)
Jacob, Bruce L., *Algorithmic composition as a model of creativity,* 1996, internet: www.ee.umd.edu/~blj/algorithmic_composition/algorithmicmodel.html.
Jarvelainen, Hanna, *Algorithmic musical composition,* 2000, internet: http://www.tml.tkk.fi/Studies/Tik-111.080/2000/papers/hanna/alco.pdf.
Jaxitron (Pseudonym), *Cybernetic Music ,*TAB 1985.
Jervell, Herman Ruge and Olsen, Kai A., *Hva Datamaskiner ikke kan,*Tromsø: Universitetsforlaget, 1984.
Langer, Susanne, *Form and Feeling,* Scribner Book Company, 1977.
Laske, Otto, *Navigating New Musical Horizons,* Greenwood P 1999.
Laske, Otto, *Algorithmic Composition in the New Century,* www
Lerdahl, Fred, and Jackendoff, Ray, *A Generative Theory of Tonal Music ,* Cambridge, MA: MIT Press, 1983.
Lerdahl, Fred, *Tonal Pitch Space,* Oxford: Oxford University Press, 2001.
Lloyd, Dan, *Simple Minds,* Cambridge, MA: MIT Press, 1989.
Loy, Gareth, *Musimathics; the mathematical foundations of music,* Cambridge, MA: MIT Press, 2006.
Kivy, Peter, *Introduction to philosophy of music ,* Oxford: Oxford University Press, 2002.
Maurer, John A. *„A brief History of Algorithmic Composition, 1999* internet: http://ccrma.stanford.edu/~blackrse/algorithm.html
Meeus, Nicolas, "Immanent tonality, manifested tonality. Some thoughts about Lerdahl's Tonal Pitch Space", *Musicæ Scientiæ*, spring 2003.
Meyer, Leonard, *Emotion and meaning in music,*2.ed (1988)., University of California Press, 1956.
Meyer, Leonard, *Style and Music,* UC 1989
Miell, Dorothy a.o, *Musical communication,* Oxford: Oxford University Press, 2005.
Minsky, Marvin, *Society of minds,*New York: Simon and Schuster, 1986.
Minsky, Marvin, *Music, Mind and Meaning,* Cambridge, MA: MIT Press, 1989 (in 'The music machine')
Miranda, Eduardo Reck (ed.), *Readings in Music and Artificial Intelligence,*Harwood A P, 2000.
Miranda, Eduardo Reck, "On the music of emergent behavior: What can evolutionary computation bring to the musician?", in *Leonardo*, 36(1), 2003, 55–59.
Mithen, Stephen, *The singing Neanderthal – the origins of music, language, mind and the body,* Weidenfeld&Nicolson, 2005.
Tomoya, Miura, and Kazuto, *An Approach to Algorithmic Music Composition with an Artificial Chemistry,* internet: http://www.tomilab.net/alife/paper/MiuraMusic2CR.pdf
Moynihan, M., *Communication and Noncommunication by Cephalopods*. Bloomington: Indiana University Press, 1985.
Osvath, Mathias and Gärdenfors, Peter, "Oldowan culture and the evolution of anticipatory cognition", *Lund University Cognitive Studies*, 126, Lund: LUCS. 2005, internet:http://www.lucs.lu.se/Abstracts/LUCS_Studies/LUCS122.html
Papadopoulos, George;Wiggins, Geraint, *AI Methods for Algorithmic Composition: A Survey, a Critical View and Future Prospects,*??.
Peterson, Ivars, *Mozart's Melody Machine,*2001, internet: http://www.maa.org/mathland/mathtrek_8_21_01.html
Pope Stephen Travis, *The Well-Tempered Object – Musical applications of Object-Oriented Software Technology,* Cambridge, MA: MIT Press, 1991.
Prigogine, I., and Stengers, I., *Order out of chaos,* Bantam books, 1984.
Puckette, Miller, *Max at Seventeen,* 2002, internet: http://crca.ucsd.edu/~msp/Publications/dartmouth-reprint.dir/.
Reiners, Paul, *Cellular automata and music,* 2004, internet: www-106.ibm.com/developerworks/jave/library/j-camusic/
Resnick, Mitchel, *Turtles, Termites and Traffic Jams,* Cambridge, MA: MIT Press, 1994.
Riemann, Hugo, *Wie hören wir Musik? Grundlinien der Musik-Asthetik,*Hesse, Berlin, 1919.
Roads, Curtis, *The Computer Music Tutorial,* Cambridge, MA: MIT Press, 1996.
Roads, Curtis, *Microsound,* Cambridge, MA: MIT Press, 2002.
Rowe, Robert, *Musicianship,* Cambridge, MA: MIT Press, 2004.
Rowe, Robert, *Interactive Music systems Machine listening and composing,* Cambridge, MA: MIT Press, 1993.
Rowe, Robert, *Machine Musicianship,* Cambridge, MA: MIT Press, 2001.
Russel, Stuart; Norvig Peter, *Artificial Intelligence – A Modern Approach,* Prentice Hall 1995. [**AIMA**]
Santos, Antonio e.a., *Evolutionary Computation Systems for Musical Composition,* internet: www.galileo.dc.fi.udc.es/cm
Scharfstein, Ben-Ami, *Of Birds, Beasts, and other artists – an essay on the universality of Art,* New York: New York University, 1988.

Schillinger, Joseph, *The Schillinger System of Musical Composition,* Carl Fischer Inc, NY, 1946.
Schillinger, Joseph, *The mathematical basis of the Arts,* Mathematical library NY, 1948.
Sedgewick, Robert, and Flajolet Philippe, *An introduction to the analysis of algorithms,*Addison-Wesley, 1996.
Schwanauer, Stephan M., and Levitt, David A. (ed.), *Machine Models of Music,* Cambridge, MA: MIT Press, 1993.
Sedgewick, Robert, *Analysis of Algorithms,* Addison-Wesley 19996.
Selfridge-Field, Eleanor, *Beyond MIDI: The handbook of Musical Codes,* Cambridge, MA: MIT Press, 1997.
Sipper, Moshe, *Machine Nature,* New York: McGraw-Hill, 2002.
Laurie Spiegel, "Brief History of Intelligent Instruments"*, Computer Music Journal*, Vol. 11, #3, 1987.
Spitzer, Michael, *The metaphor of musical space,* MUSICÆ SCIENTIÆ, spring 2003.
Sturm, Bob L., "Synthesis and Algorithmic Composition Techniques Derived from Particle Physics"*,* in *Proc. of the Eighth Biennial Arts and Tech. Symposium,* New London, CT: internet:http://www.composerscientist.com/content/research/2001_CAT.pdf
Taube, Rick, *Notes from a Metalevel,*London: Routledge, 2004.
Temperly, David, *The cognition of basic musical structures,*Cambridge, MA: MIT Press, 2001.
Temperly, David, *Music and Probability,* Cambridge, MA: MIT Press, 2007.
Todd, Peter, *A connectionist approach to algorithmic composition* ,Computer Music Journal, 13:4 ; 1989
Todd, Peter M, and Werner, Gregory, M., *Frankensteinian Methods for Evolutionary Composition, i*n NG1999.
Tsougras, Costas, *Modal Pitch Space – A theoretical and analytical study,* MUSICÆ SCIENTIÆ spring 2003.
Achille C. Varzi, and Roberto Casati, *Parts and Places: The Structures of Spatial Representation,* Cambridge, MA: MIT Press, 1999.
Visell, Yon, *Spontaneous organisation, pattern models and music,*Organised Sound, 2004.
Wallin, Nils L., *The Origins of Music ,* Cambridge, MA: MIT Press, 1999.
Walser, Robert, *Running with the Devil- Power,Gender, and Madness in Heavy Metal Music,* Hanover: University Press of New England, 1993.
Walter, Bruno, "Bruckner and Mahler", *Chord and Dischord*, internet: http://www.uv.es/~calaforr/walter.html
Weisberg, Robert W., *Creativity – Genius and other Myths,* Freeman 1986.
Wilkins, Margaret L., *Creative Music Composition,* London:Routledge, 2006.
Winkler, Todd, *Composing Interactive Music,* 2.ed.,Cambridge, MA: MIT Press, 2001.
Østerberg, Dag, *Brahms – om ekspressivitet,* Oslo: Gyldendal, 2003.
Xenakis, Iannis, 'Determinacy and indeterminacy'. *Organized Sound*, 1996.
Zlatev J., Persson T., and Gärdenfors P., "Bodily mimesis as "the missing link"", in *Human cognitive evolution,* Lund: Lund University Cognitive Studies, internet: http://www.lucs.lu.se/ftp/pub/LUCS_Studies/LUCS121.pdf.


Lisp/maxlisp
Max/MSP
User plays tunes on MIDIkeyboard or Max keyboard object
MIDI in
MIDI Parsing
Melodic Parsing
Tonal Parsing
Scale Matching and Key decision (8.3)
'Gene' Parsing
Splicing meldoy into recombinable segments
KB recombinations
Rule-based optimalizations
Rules for reasonable combinations
Rules and weights
Adapting weights (learning) from human feedback
Critic
Mutations
Population of tunes
'melodic chromosomes'
Selection of successful tunes
User decides quality of new tune output
MIDI Messages
MIDI out
Sound Module
Random generators
Seeding