David Cope and Experiments in Musical Intelligence

by

Patrício da Silva

www.spectrumpress.com

ABSTRACT

David Cope and Experiments in Musical Intelligence

by

Patrício da Silva

This paper traces the presence of algorithms in Western music and analyses the operating principles supporting David Cope's (1941-) software program *Experiments in Music Intelligence*.

Chapter 1 provides an overview of implicit and explicit algorithms as present in Western music literature, with relevance for formal rhythm, movement structure, phrase structure, phrase length, *fortspinnung*, tone systems, formalized processes to generate pitch material from text and rhythmic patterns from pitch, combinatoriality principles in *Musikalisches Würfelspiel* (musical dice game), and orchestration.

Chapter 2 covers the working details of *Experiments in Music Intelligence*, looking at data encoding, melodic, harmonic and SPEAC analysis, signatures and pattern

matching, and lexicons, which account for the creation of logical and stylistically coherent new works based on a user supplied database.

Chapter 3 provides a brief introduction to Lisp, the programming language used to code *Experiments in Music Intelligence*.

Chapter 4 presents an interview with David Cope, the author of *Experiments in Music Intelligence*.

Chapter 5 concludes with remarks regarding the impact of *Experiments in Music Intelligence* on the music community.

Appendix 1 documents the reaction of some music theorists to *Experiments in Music Intelligence* at the beginning of the XXI century.

Appendix 2 lists David Cope's publications.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter I

Algorithms in Western Music Before the Age of Computers

This chapter defines the word algorithm and looks at the presence of implicit and explicit algorithms in Western music through the definition and examples of formal rhythm, movement structure, phrase structure, phrase length, *fortspinnung*, tone systems, formalized processes to generate pitch material from text and rhythmic patterns from pitch, combinatoriality in *Musikalisches Würfelspiel* (musical dice game), and orchestration.

Algorithm

Most musicians have a difficult time admitting that music is composed of and can be understood with algorithms. Many believe that the act of composing is the consequence of inspiration and other imprecise concepts. Despite their romanticized perspective on the creation of music, we all agree on one point: music is an activity of the human mind.

Human life is a sequence of problems for which we must find solutions in order to survive. An algorithm can be described as a set of constraints that effectively accomplishes at least one task in a finite number of unambiguous steps.

Any musical style can be defined by constraints. To accept the idea that a composer can compose anything, is to endorse a chaotic perspective on the process of composition. Theoretically, a composer has unlimited freedom of choice, but music composition is not about "what you can do" but rather "what you can't do". To compose is to resolve musical problems ("what you can't do"), following sets of instructions and

observing rules. Any type of constraint requires algorithmic solutions. Every composition is a finite sequence of steps. It is logical, then, to assume that, regardless of who the composer is, the act of composing is as an algorithmic process. Algorithms emerge, then, as the most appropriate tool for the creation and study of music.

Formal Rhythm

A musical work consists of a segment of time, framed by its beginning and ending moments. In between these two boundaries, the perception of musical form is shaped by changes of content. A listener understands formal divisions based on differences and similarities between musical elements as present in time. The introduction of a new section, for example, can only be perceived as being in fact new, if it doesn't repeat what was heard immediately before.

Each section of a work occupies a portion of the overall duration. The sequence of ratios derived from the duration of each individual section to the total time length provides the formal rhythm of a work, the highest level of temporal hierarchy in music.

Formal rhythm, as an algorithm, can be either explicit, if it subordinates musical processes (*top-down* approach), or, implicit, when the algorithm is itself the consequence of musical processes (*bottom-up* approach).

In works where formal rhythm is composed from a *top-down* approach, musical processes are conditionally selected to fit predetermined lengths of individual sections. Phillip de Vitry (1291-1361), Johannes Ciconia (ca. 1370-1412), Nicolas Grenon (ca. 1380-1456), John Dunstaple (ca. 1390-1453) and Guillaume Du Fay (1397-1474), all composed using algorithms to control the time proportion between sections. Du Fay's

motet *Nuper rosarum flores,* written for the consecration of the cathedral of Florence in 1436, has its formal rhythm governed after the proportion 6:4:2:3, a symbolic reference to the biblical measurements of Solomon's temple, as described in the Old Testament (Wrigth 1994). Such deliberate concern with time proportions in early Western music, has no other historical parallel until the twentieth century. It is not until Béla Bartók (1881-1945) that form in music is again conceptualized with acute temporal intentionality. In the first movement of *Contrasts* (1938), for example, Bartók segments time according to a continuos proportion, the golden section (Lendvai, 1983), aligning the recapitulation precisely when the amount of time left to end the movement (b) compared to the time already used since the beginning of the movement (a), compared to the duration of the entire movement (c) exhibit a permanent ratio (a/b=b/c). Also John Cage's (1912-1992) "square root form", used in, among other works, *Imaginary Landscape No. 1* (1939), proposes an organizational principle in which "the whole having as many parts as each unit has small parts, and these, large and small, in the same proportion" (Harrison 1971). Karlheinz Stockhausen (1928-), in his opera *Licht* (1977-) (Stockhausen 2001), has pursued, for each of the seven week days that comprise the cycle, a compositional process with a systematic architecture of time, pre-composing, within the detail of seconds, the duration of each section.

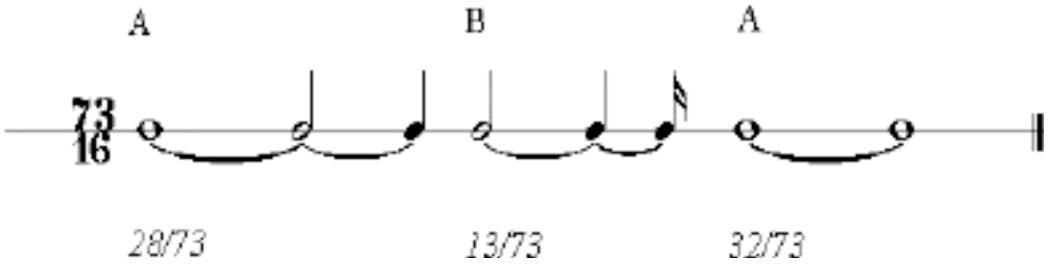
When the formal rhythm of a work derives from a compositional *bottom-up* approach, the act of composing parallels prose writing. When writing prose, one follows a trial and error strategy, never anticipating how many phrases and paragraphs will suffice to communicate the idea in mind. Similarly, a composer working from a *bottom-up* approach does not exercise any direct control over the time proportions in a work.

Instead, the number and length of individual sections follows subjective reactions to the shaping of materials leaving the formal rhythm of a work as an unaccounted consequence of phrase lengths, number of phrases, and thematic design.

Independently of which compositional approach is followed, top-down or bottom-up, the traditional representation used to describe formal designs, such as "ABA" for the sonata-allegro design, can only inform us about the flow of similarities. That is, in this particular case of an "ABA" design, we expect an initial section (A) to be followed by a second section different in content (B), followed by a third and last section that is different from the previous and similar to the first (A). Most frequently, one assumes that since the recapitulation is the reiteration of material previously presented, both the recapitulation and the exposition sections should have the same length, where the symmetrical properties of the formal design directly translate to non-retrogradable formal rhythms (i.e., palindromes). However, the description of formal designs is temporally completely uninformative. From the description of the formal design it is impossible to discern how the length of the B section compares to the A section leading to the assumption that the reiteration of A equals its first appearance. The first movement of Mozart's piano sonata in C Major KV. 545, for example, sums up to a total of 73 measures of 4/4 meter which equal to 292 beats displaying a straightforward design. The exposition (A), the development (B) and the recapitulation (A) last respectively 112, 52 and 128 beats. The ratios between each of these sections and the total amount of beats dictate a formal rhythm that can be simplified to 28/73, 13/73, 32/73. By notating these simplified ratios in traditional note values (Fig. 1), with 1/73 represented as a sixteenth-note, it becomes clear that, since the exposition and the recapitulation differ in length,

symmetrical properties in formal thematic designs are not expected to carry over to formal rhythm.

Fig. 1. Formal rhythm in Mozart's C major piano sonata (KV 545)



Aleatorism, *mobile*, and *open form*, all closely related aesthetic paths mostly explored between 1950-70, had as their implicit aesthetic goal to promote a concept of form where formal rhythm is purposely left undefined, as it varies with each performer and in each performance. In Stockhausen's *Klavierstücke XI* (1956), for example, while facing an oversize score with 19 different groups the performer may start with any them, as the composer instructs the pianist to play the first group at which the eye glances. To continue, the performer must again glance randomly at the score and select a different group, and so on. The piece concludes when a given group is played for the third time even if some groups were actually never played.

The absence of precise temporal and proportional information in the description of formal thematic designs remains a deficiency in most contemporary theoretical

thinking, underestimating the medium where music takes place: time.

Movement Structure

Many musical genres follow a multi-movement model. These models become often fashionable templates adopted by different composers and recognized by their audiences (i.e., 18th and 19th century movements based on dance tempos and meters). Such generalized models are explicit algorithms, determining tempo and formal design for each movement from which the individual composer may choose to deviate. For example, the sonata genre in the Classical period is expected to be a work in three movements, whose tempos follow a fast-slow-fast order, respectively composed in sonata-allegro form, episodic form, and rondo.

Phrase Structure

A musical style can be defined by certain permeating organizing principles. A common trait in the works of Franz Joseph Haydn (1732-1809), Wolfgang Amadeus Mozart (1756-1791) and Ludwig van Beethoven (1770-1827), for example, is the use of antecedent and consequent phrase structures as explicit algorithms, where a pair of rhythmically and thematically symmetrical, and also harmonically balanced statements complement each other. The first phrase (antecedent), ending with a half cadence, is followed by a second phrase (consequent), derived by means of algorithmic transformation of the antecedent, ending with a perfect cadence or a modulation to a different key.

Phrase Lengths

Music phrases can be thought as building blocks. Historical styles and certain music forms are characterized by a specific choice of phrase length. The classical style (Haydn, Mozart, Beethoven and their contemporaries) is undoubtedly marked by antecedent and consequent phrase structures where the predominant phrase length equals to a number of bars that is a power of 2 (i.e., 2, 4, 8, 16). Fryderyk Chopin's (1810-1849) *scherzi* for piano, for example, are constructed on phrases that are eight measures long.

Melodic Orientation

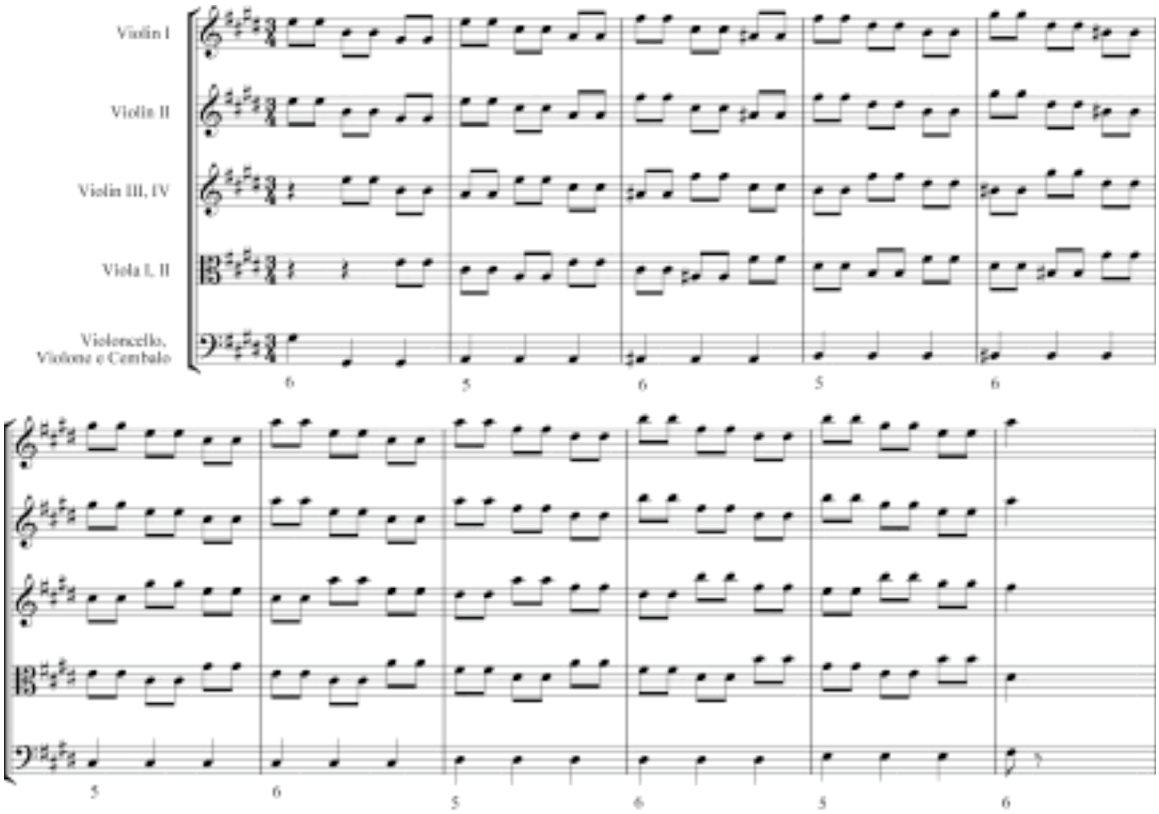
Baroque music literature is an endless resource for what twentieth century musicologists (Marissen 1999) have labeled as *Fortspinnung* (Ger.: 'spinning forth'), the continuation of musical material with reference to a template. Antonio Vivaldi's (1678-1741) concerto-style favors a *ritornello* model with three clearly defined segments from which the second is typically of sequential nature. In the third movement of his E major violin concerto Op. 3 no. 12 (RV 265) (Fig. 2), measures 7-17 provide a textbook like example of a spinning forth process. However, from an algorithmic perspective, this segment's melodic orientation and harmonic movement are generated independently. The harmonic content follows a root progression by fifths, where the fifth of one chord becomes the root of the following one. Once generated, the harmonic progression is filtered by an orientation unit cell, outputting a period melodic orientation.

The conceptual independence between melodic orientation and harmonic content can be observed in some of autographs of Johann Sebastian Bach's (1685-1750) music. In Figures 3 and 4 only the first five measures of both preludes were fully notated. Those

initial measures function, in the autograph, as a template with which the following measures' harmonic content is to be shaped.

The orientation unit cell can be described in a list of plus and minus signs, respectively describing a movement to higher and lower sounding pitches. The orientation unit cell for the C Major prelude is then represented as (+ + + + - + +).

Fig. 2. Vivaldi, Concerto in E major (RV 265), third movement (*Allegro)*, measures 7-17.



In the C minor prelude, the orientation unit cell (- - + - + - +) in the right hand is inverted for the left hand.

Fig. 3. Prelude in C Major (BWV 846a)

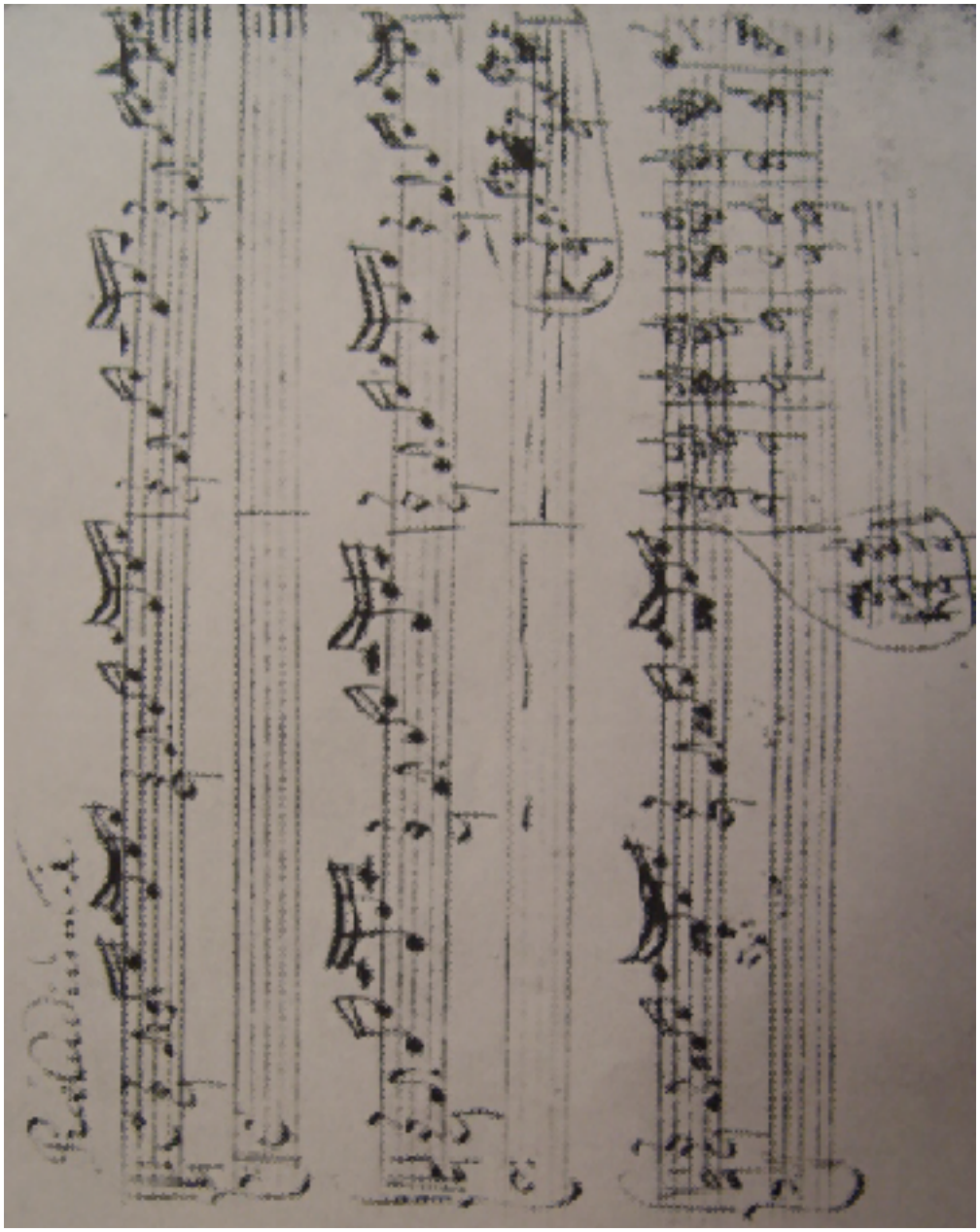
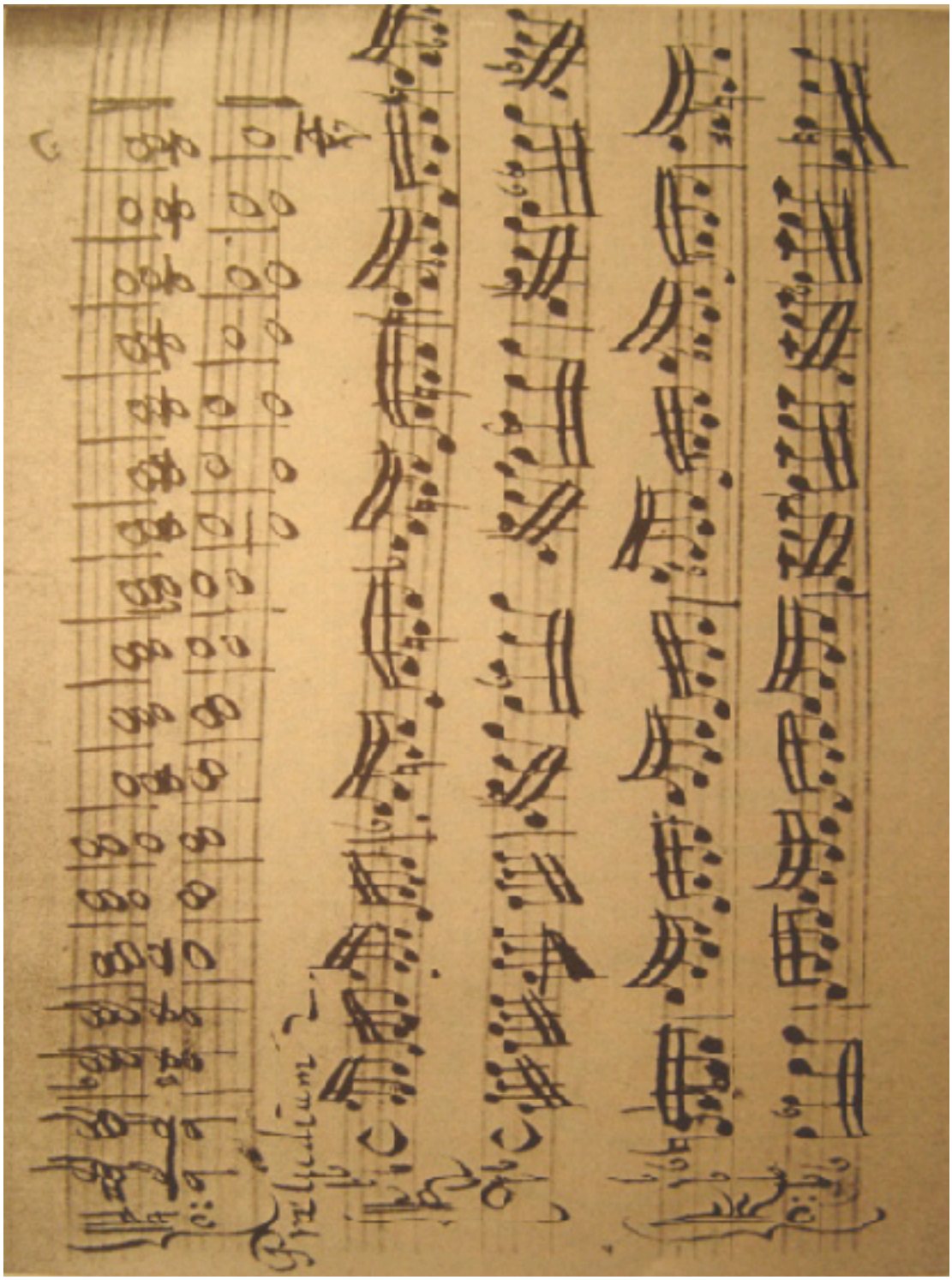
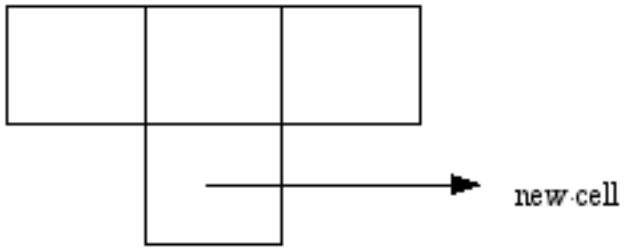


Fig. 4. Prelude in C minor (BWV 847/1)



Both unit cells presented above can be generated algorithmically with cellular automata. Originated in the theoretical work of American mathematician John von Neumann during the 1950s, cellular automata (CA) treats blocks of data as biological cells, where the behavior of each individual cell is dictated by the behavior of the surrounding cells (Fig. 5). In a three cell CA, each new cell is a consequence from the behavior in the three cells immediately above:

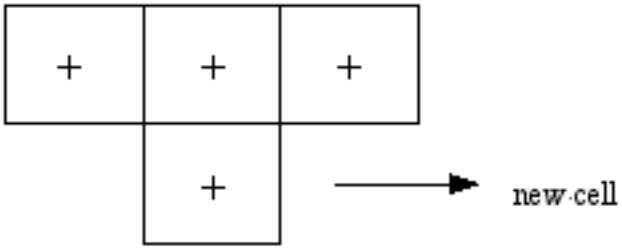
Fig. 5. Cell behavior in cellular automata



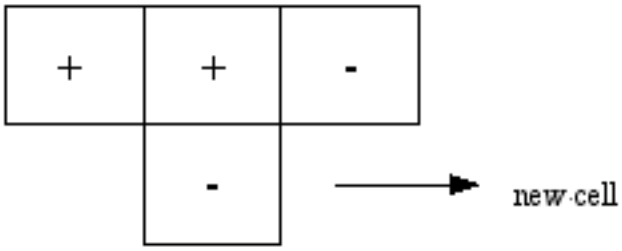
The orientation unit cell for the C Major prelude (+ + + + - + +), for example, has its presence traced in the following CA:

+	+	+	+	+	+	+	-
+	+	+	+	+	+	-	+
+	+	+	+	+	-	+	+
+	+	+	+	-	+	+	+
+	+	+	-	+	+	+	+
+	+	-	+	+	+	+	+
+	-	+	+	+	+	+	+
-	+	+	+	+	+	+	+

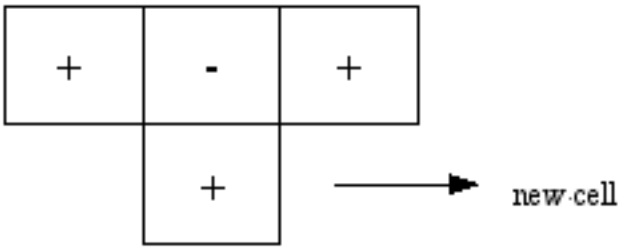
In the model above, each plus or minus sign is called a “state”. The status of each state depends of programmed rules. The cell behavior in the CA above follows the following rules:



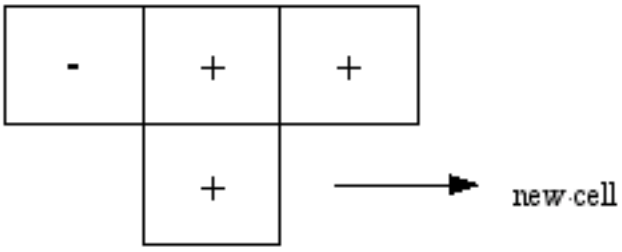
if (+ + +), then +



if (+ + -), then -



if (+ - +), then +



if (- + +), then +

Tone Systems

Certain musical styles are historically and geographically constrained to specific tone systems. These pitch organizations, when plotted in the circle of fifths (hereafter COF), can reveal their generative algorithm. Anhemitonic pentatonic scales, one of the oldest tone system known, typically associated with nonwestern traditional cultures and European plainchant's resonant backbone, can be generated from an algorithmic perspective by collecting any five consecutive pitches from the COF. Heptatonic tone systems follow similar procedures. For example, the interval content of the lydian mode

can be generated by collecting seven consecutive fifths in the COF, starting in C and moving clockwise to F#, the ionian mode starting from F to B, the mixolydian from Bb to E, the dorian from Eb to A, the aeolian from Ab to D, the phrygian from Db to G, and the locrian, moving counterclockwise from C to F#. The generative algorithm for the whole-tone scale, a trade mark for late 19th and early 20th century French music, uses one from each two consecutive fifths in the COF. Twelve-tone rows, a type of tone organization traditionally associated with serialism as initially developed by Arnold Schoenberg (1874-1951) during the first quarter of the 20th century, are generated with the rule that no tone of the equal-tempered chromatic scale can appear repeated. The original twelve-tone row is further systematized in a magic square of twelve rows, where each new row corresponds to an exact transposition of the original intervals, so that in each column all pitch classes appear only once.

Generating Pitch From Text

Western composers have, throughout history, devised algorithms to generate pitch material from written text. Guido d'Arezzo (ca. 991- ca. 1028) proposes, in chapter XVII of *Micrologus* (1025-26) (D'Arezzo 1978), an algorithm capable of producing a melodic line from any input text. The core idea behind Guido's algorithm can be reconstructed in the following steps:

Step 1- create a two-dimensional table, each dimension with sixteen empty slots.

Step 2- fill the empty slots in the top dimension with the rising steps of the gamut (G A B C D E F G a b c d e f g a), according to the standard vocal range.

Step 3- fill the empty slots in the bottom dimension with a cyclic sequence of all

vowels (a, e, i, o, u, a, e, i, o, u, a, e, i, o, u, a).

Step 4- given any input text, eliminate all consonants and retain all vowels in their original order of appearance.

Step 5- take the first vowel, find it in the bottom dimension and replace it by the corresponding pitch in the top-dimension.

Step 6- repeat Step 5 for each of the remaining vowels.

Step 7- if no more vowels, proceed to Step 8.

Step 8- Your melody is ready! You have the right to use or reject the output, partially or in its entirety. For every pitch in the melody that you reject, you may go back to Step 5 and look for a different slot that, while storing the same vowel, corresponds to a pitch different than the one first obtained.

Guido's table look-up method can be divided into different stages. The first stage is the transformation process of a vowel to one of the table's possible corresponding pitches and, the second stage, based on stylistic preferences, is the selection of the best choice (Loy 1989).

Musical acrostics, a related strategy to Guido's algorithm, derives the order of pitches in a theme from one or more words that, traditionally, hold a special interest to the composer. The *Bb-A-C-H* motive that J. S. Bach introduced in the last and unfinished fugue of the *Art of the Fugue* is a musical analogy of his own name, where the first and last letters of the name, *B* and *H* have been converted respectively to the pitches *Bb* and *B§*. Numerous composers such as Beethoven in his *String Quartet* Op. 59 No. 2, Robert Schumann (1810-1856) in his *Sechs Fugen über den Namen: Bach*, Franz Liszt (1811-

1886) in his *Fantasy and Fugue on B.A.C.H.*, Anton Webern (1883-1945) in his *String Quartet* Op. 28 and Pierre Boulez (1925-) in his *Second Piano Sonata* have included the same four tones of the B-A-C-H motive, as a symbolic gesture of admiration to J. S. Bach.

Musical acrostics have also been used with programmatic intentions, as in, for example, R. Schumann's *Abegg Variations,* variations on the name of a lady friend spelled as *A-Bb-E-G-G*, and in the motto theme 'ArnolD SCHönBErG', 'Anton wEBErn' and 'AlBAn BErG' with which Alban Berg (1885-1935), in the opening of the *Kammerkonzert*, represents the three members of the Second Viennese School.

Generating Rhythm From Pitch

Henry Cowell's (1897-1965) *New Musical Resources* (Cowell 1996), first published in 1930, proposes an analogy between rhythmic patterns and just intonation ratios. Since each harmonic in the overtone series is an integer multiple of a given fundamental (i.e., *f, 2f, 3f, 4f, 5f,* ...), Cowell imports this spectral information to rhythmically articulate harmony, thus redefining pitch and rhythm as different time scales of the same phenomenon (Roads 2002). A major triad in close position, would be rhythmically notated, so that the number of articulations and respective duration are in correspondence to the harmonics *2f*, *3f*, and *5f.* In this case, the root of the triad (second harmonic) would be rhythmically articulated as two eighth-notes, the fifth of the triad (third harmonic) as an eighth-note triplet and the third of the triad (fifth harmonic) as a sixteenth-note quintuplet, respectively *2f*, *3f*, and *5f* where *f* =quarter-note. Joseph Schillinger (1895-1943) idealized one of his composing machines, Rhythmicon (built by

Leon Theremin), to compose and perform patterns according to Cowell's principle of spectral rhythm (Schillinger 1948).

In "*.... How time passes....*" (Stockhausen 1961), Stockhausen describes the creation of a tempered chromatic scale of duration paralleling the equal-tempered pitch scale. Since, traditional note values cannot properly represent the values resulting from the division of the octave ratio in twelve equal parts, Stockhausen opted for a common note value set to a tempered chromatic scale of metronome settings. Such scale, within the octave 𝅗𝅥= 60 to 𝅗𝅥= 120, is represented as 𝅗𝅥w= 60, 𝅗𝅥= 63.6, 𝅗𝅥= 67.4, 𝅗𝅥= 71.4, 𝅗𝅥= 75.6, 𝅗𝅥=80.1, 𝅗𝅥= 84.9, 𝅗𝅥 = 89.9, 𝅗𝅥 = 95.2, 𝅗𝅥= 100.9, 𝅗𝅥= 106.9, 𝅗𝅥= 113.3, 𝅗𝅥= 120.

Musikalisches Würfelspiel

In the era of Pascal's probability calculus, illustrious composers, such as Johann Philipp Kirnberger (1721-1783), Mozart, Carl Philipp Emanuel Bach (1714-1788) and Haydn composed examples of *Musikalisches Würfelspiele* (musical dice games), one of the first examples of combinatoriality in formal types of music (Cope 1995). In such works, a two-dimensional table of musical figures offers, in the horizontal dimension, a sequence of musical phrases indexed measure by measure and organized by function. For each of the measures in any phrase, the table displays, in the vertical dimension, a set of syntactically equivalent musical figures. The realization of a such musical games is the assemblage process of musical figures by chance operations (i.e., the throwing of the dice). The entertainment purpose and the popularity earned back then by such musical games should not obscure their theoretical significance. The compositional process underlying a Musikalisches Würfelspiel suggests that, similarly to natural languages

(Chomsky 1976), can be defined in terms of syntactic relationships (essentially ordering of symbols).

Orchestration

Orchestration styles can be described as sets of principles operating on a given input. Orchestration treatises often suggest those principles as historical recipes, referring the reader to exemplar solutions in the literature: “When a very soft harmony is to be produced by string instruments, it is frequently better to give the bass part to the violoncellos alone and let the the double-basses rest. Weber did so in the accompaniment to the Adagio of Agathe’s wonderful aria in the second act of *Freischuetz* ” (Berlioz and Strauss 1991).

Chapter 2

Experiments in Musical Intelligence

This chapter describes the different steps and working details of David Cope's *Experiments in Musical Intelligence* (hereafter EMI) such as data encoding, melodic, harmonic and SPEAC analysis, signatures and pattern matching, and lexicons, which account for the creation of logical and stylistically coherent new works based on a user supplied database

Encoding Works in Databases

EMI's algorithms can't operate miracles, though sometimes a surprised user may believe so. One must keep in mind that any information outputted by EMI derives directly from previously existing data as stored in the database, the ground-level of the program. Only continued practice can educate the user to associate a given database with certain musical results. However, a few specific steps regarding the selection and editing of works must be observed to ensure an uncompromising start.

The works selected should share common stylistic traits if the user expects a stylistically coherent output. The more stylistic diversity in a database's content, the more unpredictable becomes the musical result of recombinancy.

Any errors present in the database, such as wrong notes that might've passed undetected during the sequencing process, will most likely reappear in the final output, corrupting the integrity of the musical style originally intended for replication. MIDI files downloaded off the internet offer no guarantee of being mistake free data. Only the proofreading of such sequences can assure its correctness.

Separate timbres must have their identity protected. Each timbre must be assigned to a different MIDI channel. Failure to observe this distinction will compromise the identification and matching of patterns since it forces the program to work with data in a format that does not correspond to the original score.

A database may include recurring objects rhythmically set to different note-values in multiple works or different movements from the same work . The classical Alberti Bass pattern, for example, can be found permeating most of Mozart's piano sonatas. Despite the apparent similarities between its multiple recurrences, the Alberti Bass may appear in some cases articulated using sixteen-notes and in others with eighth notes, depending on the movement's rhythmic context. Such discrepancies of note duration must be leveled out to a common denominator, an essential procedure for a successful recombinancy at the rhythmic level.

All works must be stored using the same tonal center. That is, the recombinancy of segments can only be successful if the segments are stored in the database according to a common key and mode.

The large number of non-harmonic tones usually involved in ornamentation figures can mislead the identification of *signatures* (recurring patterns that characterize a particular composer's *corpus* of works, typically two or more beats in length). Hence, trills, mordents, and other ornamentation figures should be removed from the sequence prior to its storage in the database. Once removed from the data its reinsertion in the output must be done by hand according to the user's judgment.

Music in EMI is represented and stored as lists of *events* in a database. An event is Cope's representation of the essential attributes of one note in a list of five parameters:

(0 60 500 1 127)

The positioning of parameters inside a list indicates their attributes. From left to right, the parameters in a list represent the attack time (on-time) in milliseconds, the MIDI key number (60 equals to middle C, 61 to C sharp, 59 to B, and so forth), the duration (where 1000 usually represents a quarter-note, 500 an eighth note, 2000 an half-note, etc.), the MIDI channel number (which allows for instrumental or voice differentiation), and the note velocity (where 0 is the softest, and 127 the loudest possible). On-times are relative to the metronomic indication in use. With the metronome set to quarter-note = 120, for example, an on-time of 1000 would actually be played 500 milliseconds after zero. Typically events in a list appear sorted by their on-times.

Melodic Analysis

Although Cope describes EMI's melodic analysis in Schenkerian terms, his results have proved to be far more relevant than the traditionally limited theoretical approach. EMI provides automated, stylistically nondiscriminatory, hierarchical and functional melodic analysis based on the terminal-point destinations, allowing the new compositions resulting from the recombination process to coherently recreate, moment by moment, middle and background structural levels (see Cope, 1996, pages 184-187).

A key point in the melodic analysis process is the concept of *signature*. Cope defines signatures (Cope, 1991) as patterns of 1 to 8 contiguous melodic intervals (potentially a larger quantity if including harmonic intervals) that occur in more than one

work from the same composer; that is, signatures are style-dependent but work-independent.

Harmonic Analysis

Harmonic functions are analyzed in EMI with the conviction that, similarly to natural languages, musical grammars depend on syntax rather then semantics. In language, nouns can function as subjects or objects depending on their context. In tonal music, for example, a dominant chord may be syntactically a statement when articulated in the beginning of a phrase, or a consequent when sounding at a cadential moment.

Cope developed in 1985 (Cope 1987) the SPEAC system, the first musical implementation of an augmented transition network (ATN), a finite-state automaton with recursive succession rules between sub-phrases allowing for logical syntax substitutions.

SPEAC is an acronym for a functional analytical system based on five identifiers, where S stands for *statement* (declaration of material or ideas, in most cases preceding or following any other SPEAC function), P for *preparation* (typically occurring prior to statements and antecedents), E for *extension* (primarily following statements but also any other SPEAC function), A for *antecedent* (normally preceding consequents), and C for *consequent* (must be preceded by antecedents either directly or indirectly when including intervening extensions). Succession rules for each of SPEAC abstractions have imposed limitations where a statement can be followed by either P, E or A, a preparation by S, A, or C, an extension by S, P, A, or C, an antecedent by E, C, and a conclusion by S, P, E, A.

Any of SPEAC abstractions are assigned to a grouping of notes depending on levels of tension between intervals, metrical placement, and agogic emphasis, measured

both in the preceding and following groups. At the intervallic level, tension measurements depart from basic acoustical concepts. Tension is evaluated based on lower-occurring and lower-rooted intervals, and upper-occurring and upper-rooted intervals, respectively indicating least and most tension. Given one harmonic series per pitch class (typically considering only the first sixteen partials), an interval's acoustic fundamental is determined by identifying the harmonic series where the given interval appears positioned closer to the fundamental. The next step is to define the interval's root by locating the member of the interval closest to a fundamental's multiple. In this fashion, minor seconds are ranked with the highest tension (occurring highest in the series (16/15) with upper root), and perfect fifths with the lowest tension (lower placement in the series (3/2) and lower root).

The data resulting from this preliminary analytical stage is converted to an intuitive scale of numerical values. The converting procedure reduces all intervals to the octave space, where numerical values are distributed symmetrically around the augmented fourth, with the exception of the perfect fifth which, just as unisons and octaves, produces the lowest tension levels.

The following stages in SPEAC analysis cover metric placement and agogic. The further away an interval is positioned to the right of the barline, the higher the level of tension produced independently measured at the beat and off-beat levels. Agogic analysis looks at note duration, where larger values translate to lower tension measurements.

The combined action between SPEAC analysis and pattern matching (explained ahead) allows the proper cataloguing of musical components and the construction of appropriate lexicons according to syntactical functions thus ensuring logical

recombinancy.

Pattern Matching

Cope (1996) reveals that the most successful works in EMI's output result from a small sample size recombination (typically beat-to-beat). A small sample size requires a large number of recombinancy operations, allowing the output of new works with no direct references to individual works stored in the database. However, signatures are typically larger than one single beat. The pattern matching (unsupervised learning) component in EMI ensures the protection of signatures once these are identified. The identification and protection of signatures is an essential step to preserve the stylistic identity of the database submitted for recombination.

Pattern matches in EMI can operate at several levels including pitch, rhythm, timbre (essential to detect orchestration signatures) and dynamic levels. The matching process is optimized through *controllers* (variables which control the amount of deviation allowed for a match to occur) (Cope 2003b). The use of controllers enables the identification of patterns within a user defined window of deviation (*allowance*), thus emulating the human ability to recognize the reoccurrence of musical objects even when these are not exact copies of each other. Slight variations in a recurrent pattern are very common in music, in fact, one should expect to find at least as many approximate as perfect matches. Without a deviation allowance, the pattern-match program would fail to recognize a fugue's tonal answer as essentially the same as the subject.

At the pitch level, for example, when looking for patterns, EMI accounts for deviations such as chromatic transposition, inversion, interpolation (intervening notes

among the original motive), fragmentation (suppressed notes from the original motive), re-ordering of notes, contour, number of differing intervals, amount of intervallic deviation in semitones, total number of notes to be matched, and a required minimum number of matches to accept a recurrent pattern in the database as a signature.

Chapter 3

The Tool

This chapter offers a brief guide to the programming language with which EMI has been coded. It is intended to provide those less familiarized with programming with an introduction to the simple mechanisms operating at the lowest levels of the program. When these simple mechanisms are connected and combined in larger programs as it happens, for example, in EMI, impressive results can be achieved.

Lisp

David Cope's preferred programming language, with which EMI continues to be programmed since the early 1980's, was first defined in 1958 under the direction of John McCarthy at the Massachusetts Institute of Technology (MIT). Lisp is a high-level interpreted computer programming language remaining today as the earliest functional programming language, the oldest programming language still in use, and as the *lingua franca* for the field of artificial intelligence (AI) (Tanimoto 1987).

A high-level computer language differentiates itself from an assembly language by providing built-in functionality such as declarations, control statements, automated memory management, etc. All these processes, contribute significatively to the performance of complex computations using a rather reduced number of instructions.

An interpreted language translates a program (source code) into machine (or object) code following the flow of written instructions statement by statement. The contrasting approach is the compiled language type, such as in Fortran, C or Pascal languages, in which all instructions are translated at once to machine level.

The word Lisp stands for its own structural principle, List Processing[1], where the elements of linked lists are connected in the machine's address space by pointers (rather then proximity), thus allowing far more flexible data structures.

Lisp was created for the processing and manipulation of structured symbolic information in an interactive environment (Harrison 1990). The user types expressions in the front-end (*listener window*) which evaluates them and replies by printing a value followed by a top-level prompt, letting the user know that the system is ready for another request. The top level prompt may appear signaled differently according to the Macintosh Common Lisp implementation in use. In conformity with Cope's own texts, I'll adopt here > as the symbol for the top level prompt.

An example of a dialogue between the user and Lisp would be to ask the system to print the value of the integer 7. The user types the given integer on the listener window after the prompt, presses the carriage return to evaluate the request, and the resulting value is immediately printed, followed by one more prompt signaling the system's availability to accept another user inquiry:

> **7**

7

>

Despite the infinite combinatorial possibilities of symbolic expressions in Lisp (hereafter s-expression), every legal s-expression in a Lisp program and data statements

[1] IPL (Information Processing Language), a low-level programming language is commonly referred as the first of the list-processing kind ever to be developed (Herbert Simon, Allen Newell and J. C. Shaw at Carnegie, ca. 1956) (Hofstadter 1985).

belongs to one of the two possible and mutually exclusive Lisp types of structure. With one exception, all legal Lisp's s-expressions are either atoms or lists. The exception is NIL, as will be explained ahead, can be both an atom and a list.

An atom is an s-expression that can be either numerical or literal. Numerical atoms have permanent values, meaning that, a number always represents itself as the value of 7 always remains 7. Literal atoms can be made into variables and symbolize any given value. Two exceptions to note are NIL and T that, similarly to numerical atoms, have permanent value, where T always equals to T (boolean value for true) and NIL to Nil (the boolean value for false):

> **t**

T

> **nil**

NIL

A list, an ordered collection of *n* elements, is an s-expression delimited by a matching number of left and right parenthesis. Every list carries content. A list's content is described by the elements inside. These elements can be atoms or other lists (i.e., nested lists) or a combination of both. In case of an empty list, such list appears represented as () or as NIL. As above mentioned, NIL has the exceptional quality of being simultaneously an atom and a list. That is, NIL, the literal atom that represents the empty list is a list itself. The content represented inside the bounding parenthesis of a list can enclose collections of data elements, a function's definition, a function call or it may be

left empty. Here follow some examples of Lisp's structures:

COPE	a literal atom
EMI	a literal atom
2003	a numerical atom
5/3	a numerical atom
(compose a (analyze b))	a list of S-expressions
()	an empty list
NIL	a literal atom and an empty list

Lisp protocols impose specific syntactic roles to the elements of a list according to their position inside the list. The first element following a left parenthesis, unless preceded by a single quote, should be a function. Functions are instructions for manipulation of data (atoms or lists). If the first element is indeed preceded by a single quote then that element is data and not a function.

(function-x 'data-1 'data-2)

The remaining elements following the function and prior to the right parenthesis are data:

> **(+ 1 3)**

4

Arguments that are themselves expressions are evaluated first. In the next example the asterisk means multiplication and the slash means divide, so that this expression equals to 25 - 1:

> **(- (* 5 5) (/ 2 2))**
24

Strings, just as numbers and the boolean constants T and NIL, evaluate to themselves:

> **"this is a string!"**
"this is a string!"

EMI and Lisp

EMI's music encoding process takes a MIDI file and returns a list of events. Each event (itself a list) describes one note in its five basic parameters, respectively, from left to right, attack-time in milliseconds, MIDI key number (60 equals middle C), duration in milliseconds, MIDI channel (instrument or voicing), and note velocity (see also chapter 2). The following example shows how the first beat of Bach's chorale no. 64 is encoded in EMI, with the soprano assigned to MIDI channel 1, alto to MIDI channel 2, tenor to MIDI channel 3, and bass to MIDI channel 4:

((0 60 1000 4 60)(0 64 1000 3 60)(0 67 1000 2 60)(0 72 1000 1 60))

Variables

Translating a complete composition from traditional notation to a list of events can typically result in a cumbersome list of data. Binding data to variables helps to efficiently enclose data in an abstraction, thus greatly facilitating its manipulation in the programming environment. Enclosing data in a variable can be done in Lisp by using **setq** (set quote). For example, the symbol **first-beat** can be assigned to represent the events that constitute the first chord of the chorale shown before:

(setq **first-beat** '((0 60 1000 4 60)(0 64 1000 3 60)(0 67 1000 2 60)(0 72 1000 1 60)))

Accessing Data in Lists of Events

Lisp provides some primitive functions (built-in functions) that, when combined with each other, can be used to retrieve any element at any depth in a list. The acronym **car**, used on the IBM 7090, stands for "content of the address register" and retrieves the first element in a list:

> **(car first-beat)**
(0 60 1000 4 60)

The acronym **cdr** ("content of the decrement register") retrieves all but the first element in a list:

> **(cdr first-beat)**

((0 64 1000 3 60) (0 67 1000 2 60) (0 72 1000 1 60))

Lisp also provides mechanisms to access data located at the end of lists. The function **last** will, for example, return the last element of a list in a list form:

> **(last first-beat)**

((0 72 1000 1 60))

Deeper levels in the list structure can be accessed by re-articulating these two same functions in different combinations. The following code shows a possible example of how to retrieve the first event's attack time:

> **(car (car first-beat))**

0

The second event's MIDI key number is the second element from the second list in the list of events. It can be accessed, for example, with:

> **(car (cdr (car (cdr first-beat))))**

64

Creating Lists of Events

Lisp provides the function **list**, that as the name indicates, creates a new list with the elements that it is provided. Our original list **first-beat** contains events sorted from lowest to highest voice. To construct a new list with only the outer voices from **first-beat** we could write:

(list (car (last first-beat)) (first first-beat))

((0 72 1000 1 60) (0 60 1000 4 60))

Some situations may require an insertion of an event into a previously existing list of events. The function **cons** ("constructor") takes two arguments, an atom and a list, and returns one list, as in the following example, where the attack time is re-inserted back into the event:

> **(cons 0 '(72 1000 1 60))**

(0 72 1000 1 60)

A related function to **cons** is **append** that accepts two lists as arguments and returns one::

> (append '((0 60 1000 4 60)) '((0 64 1000 3 60)))

((0 60 1000 4 60) (0 64 1000 3 60))

Defining Functions

Functions, as seen already with Lisp built-in **car**, **cdr**, etc, process data. To declare a new function other than those pre-existing ones, Lisp provides **defun**, itself a function, to be followed by a list of argument names representing data. The body of the function consists of other functions operation on the list of specified arguments. The following example demonstrates the definition of a new function that given any time-duration will return its golden-section:

> **(defun find-golden-section (time-duration)**
(* .618 time-duration))
FIND-GOLDEN-SECTION

Once defined, the new function can be called:

> **(find-golden-section 60)**
37.08

Recursive Functions

Much of Lisp's power and elegance lies in its recursive possibilities. Recursion can be found outside of Lisp, for example, as the mechanism generating the Fibonacci mathematical sequence. Each new number in this sequence is always the result from the addition of its two previous numbers. Given a seed of two numbers (1, 1), the recursive principle would generate the next five numbers as follows:

1+1=2

1+2=3

2+3=5

3+5=8

5+8=13

The same recursive process can be applied to collect the MIDI key numbers from a list of events.

```
(defun collect-MIDI-notes (list-of-events)
 (if (null list-of-events) ()
 (cons (second (first list-of-events))
 (collect-MIDI-notes (rest list-of-events)))))
```

The program collect-MIDI-notes can be analyzed line by line as:

(defun collect-MIDI-notes (list-of-events)

Defining a new function called collect-MIDI-notes that accepts one argument, in this case, a **list-of-events**.

(if (null list-of-events) ()

This line of code acts as a test that the program must perform each cycle in order to proceed. In this case it functions as a question for which the answer can either be true or false. The program asks: is this **list-of-events** empty? If that's true, then stop, otherwise proceed with the following instructions.

(cons (second (first list-of-events))
(collect-MIDI-notes (rest list-of-events))))

Construct a list with the value corresponding to the second atom from the first event in the **list-of-events,** while re-calling the function **collect-MIDI-notes**, this time, with all but the first event in the **list-of-events. The** process is then repeat, and "is the **list-of-events** empty?" is asked again. Since the function **collect-MIDI-notes** gets re-called each time with one less event in the **list-of-events**, inevitably there will be a moment where the **list-of-events** will be empty. When the function **collect-MIDI-notes** is re-called with an empty list, it will stop and return a list with all the collected notes:

(60 64 67 72)

Chapter 4

Interview with David Cope.

This interview was conducted by e-mail from August of 2002 to June of 2003.

PdS: What is your concept of music?

DC: With the understanding that you've not asked "What is *good* music," I feel that music (with due respects to Varése) is "organized sound and silence."

What is *good* music made of?

A balance of unity and variety.

What do you want from music?

Order amidst chaos.

Does your musical thought accept and strive for beauty?

No. Beauty means different things to different people; in fact, it means different things to the same person depending on circumstances. If I depended on a sense of beauty in my work, I would never finish anything.

Can algorithms create expressive music?

I don't know of a single piece of expressive music that wasn't composed, one way or another, by an algorithm.

What is *expressive* music?

According to my dictionary, one of the meanings of "expression" is "a showing of feeling or character." There's nothing said about intent or about a shared response to expression. Therefore, to me, expression is what I receive from music when I *feel* something in response to it. What I feel need not be composer intended nor felt by anyone else. All the other meanings attributed to expression in my dictionary refer to words, which then don't apply directly to music.

Since what one *feels* in response to music has no obligatory direct correlation between what the composer intended to express nor to what others may have *felt* from whatever was meant to be expressed is there a true musical *expression* ?

I think we're lost in semantics over the word "expression." I want the word to mean "expressive" as in the musical term "espressivo" where one is to be expressive with the notes given. It's wonderfully vague. I suppose polemically, the only "true expression" according to my definition would be one that had no direct correlation with the intent (and therefore the only truly "false" expression would be one which accidentally matched the

received expression with the intended one). Sorry for the word games, but simply put (with expression aside), I don't believe that music communicates anything or that when I am moved by a piece of music it means anything other than that I am moved by it (possibly in similar ways that I am moved now by the fog as it drifts in from the ocean - it doesn't intend to move me nor do I imagine for a second that others even like the fog - which many don't - no less be moved by it).

What is musical inventiveness?

The ability to interlace melodies, harmonies, timbres, articulations, dynamics, rhythms, forms, and so on in ways which disguise their true origins and thus sound original.

What is musical coherence?

I think my answer to that is the same as an earlier answer: A balance of unity and variety.

Are you implying that *good* music can only be the output of coherent inventiveness?

Well, I certainly think that contributes to good music. I can't imagine a good piece of music lacking a demonstration of coherent inventiveness.

Is there a musical difference between coherent inventiveness to inventive

coherence?

Sure, and they're both wonderful!

How should the modern composer be educated?

The theory part would be:

(1) skills - singing, hearing, playing;

(2) music - the textbooks for the classes would be just music;

(3) algorithms - students and teacher would study and perform music to extract the algorithms that the composers used to create this music.

The composing part would be:

Use the skills, knowledge of music, and algorithms from the above to initially compose music in the styles and forms of known music and to slowly derive from this process their own style algorithms.

Will a composer educated in a standard American university get what you've just suggested?

Mostly, no. In general, American university music programs are too fragmented (i.e., not integrated) with one teacher in charge of sight-singing, another teaching ear-training, another teaching theory, another teaching keyboard. Often these segmented areas run at different paces and it's very hard for students to get any idea how they are related. Also,

theory is most often taught as a series of generalized rules prohibiting things which composers actually did. The rules are often expressed as definite rather than approximate rules and students create academic and often useless results. Rarely are students asked or encouraged to relate what they're doing to actual literature, thus it often seems to them that these are math classes rather than music classes. While certainly some skills are learned, because they seem divorced from reality they are quickly forgotten. Rarely do young composers get a chance to model music after music from the past. Rarely is style discussed. The term algorithm is often considered scientific and ignored.

You conclude your article *On Algorithmic Representations of Musical Style*, (Cope 1992), with the following statement: "Music may or may not be the universal language, but the evidence that it is a language seems substantial." Do you assume that, as modern linguistics proved with the concept of a generative grammar, there are similar biological constraints in our brain for the processing of music, of all music, independently of cultural contours, that is, if tonal, atonal or any other?

Note that I use the words "may or may not be" and "seems substantial" as cautious suggestions rather than statements of fact. Stating that "modern linguistics proved" on the other hand seems very bold. Given current psychological, genome, and brain revelations, I doubt very much that anyone has "proved" anything much at all regarding the brain. What I believe I assumed in my statement is that we "process" the pitches, loudness, rhythms, inflections, and articulations of music with many of the same "processors" that we do

language. The mix of biological and experiential influences on our thinking is so complex and individual that we may never understand it fully. However, I hope that while there are vague inherited reactions to, say, consonance and dissonance, that we are not preprogrammed toward, say, tonal music over atonal music. Unfortunately, even the studies at major universities of child reactions to consonance and dissonance are hopelessly biased (I was particularly privy to a recent such study at Harvard, for example).

At the core of EMI is the idea of recombinancy. Does a composer compose or recompose?

Music composition consists of a combination of what we hear and what formalisms we bring to bear. If I compose a work freely (i.e., without a prescription for voice-leading, allowable verticalities, etc.) then I will most likely integrate various ideas that I've previously heard. If I compose a piece strictly using a mathematical formula, then I won't be re-composing music that I've heard but following strict rules. Most music consists of a combination of these two factors. The notion that humans have some kind of mystical connection with their soul or God, and so on, allowing them to produce wholly original ideas (not the result of recombination or formalisms) seems ridiculous to me.

Has the term recombinancy an implicit genetic metaphor?

The term recombinancy certainly has biological relevancy. However, I use recombinancy

to mean two or more ideas which recombine to create a new idea.

Do you think of music recombinancy as a biological constraint of the human mind?

No.

If I recombine by hand the most recent works of David Cope, who gets the copyrights, you or me?

This would depend on the size and number of the recombinations. Reversing the order of two halves of one of my works would be plagiarizing. Composing a new work on the first four pitch classes of one of my compositions would not be plagiarism. There are, of course, an almost infinite number of gradations between these two extremes and somewhere in the middle, things get very gray. These should be decided on a case-by-case basis.

Part of the success of EMI is dependent on the selection and preparation of works for the database. If I prepare a database of Cope's music for EMI to recombine who's the author? You, EMI, me, or the three of us?

Same answer as above. If the program's sample size is small, then the credit goes to you, if the sample size is so large that large segments of my work are quoted verbatim, then it's

probably a bastardization of my work.

In the process of recombining music materials does EMI ever plagiarize?

I'll let the courts decide this. However, in general, the two factors we have been discussing - size and number of borrowed materials - represent the distinction. Note, however, that most composers plagiarize dozens if not hundreds of works in a single piece of music, and most do this subconsciously. This is creativity, or at least a part of creativity. The composer who borrows consciously from a single work is either plagiarizing or creating a set of variations (usually distinguished by the title of the work being composed).

How can we expect EMI to behave when working, for example, from a strict twelve-tone music input, let's say, a database of Webern works? Can the principle of recombinancy be articulated with that of strict twelve-tone writing? Does a row survive recombinancy?

No. As with fugues, certain formalisms do not work easily with recombinancy and require additional code for analysis and composition.

Beyond recognizing the style of EMI compositions, why do you feel many listeners are moved by these works?

Every work of music, unless it has been composed entirely by a formalism, contains within it many pointers to the musical culture which helped to create it. These pointers,

whether they be rules, allusions, signatures, earmarks, etc., help us to relate to that work, even as we're hearing it for the first time. These pointers also point to other styles and works which themselves have pointers providing us with a rich and deep history of the cultural evolution of the work being heard. The music of EMI, because of the manner in which this music is composed, also has pointers and belongs to the culture and traditions of the music of music in its database. This helps to explain, I believe, why many EMI works obtain an almost immediate sense of intimacy with those familiar with the inherited musical culture, even those who steadfastly resist feeling such intimacy.

Chapter 5

Conclusion

Experiments in Musical Intelligence began in the early 1980's as David Cope's attempt to create algorithmically new instances of his own music. The creative model proposed with recombinancy processes in EMI suggests that fresh thoughts and invention are culturally inherited. As Cope writes in (Cope, 1996):

"The genius of great composers, I believe, lies not in inventing previously unimagined music but in their ability to effectively reorder and refine what already exists."

Earlier examples of formalized recombinancy date back to the eighteenth-century with *Musikalisches Würfelspiele* (musical dice games). More generally, re-ordering and re-contextualization of previously existing music material can be found in innumerable examples of musical allusions in the Western music literature. In their works composers, consciously or unconsciously, "borrow" from previously existing works, establishing a network of musical pointers within the culture that created them (Cope, 2003a). EMI "composes" by attempting to make *Musikalisches Würfelspiel* out of a database of music that originally was not designed as such. The merit of EMI's success lies in SPEAC (EMI's analysis system), the pattern matching component and the implementation of an augmented transition network (ATN) responsible for the reconstruction process.

SPEAC analyses the function of note groupings in a work based on levels of tension between intervals, metrical placement, and agogic emphasis. Note groupings have their function labeled with an identifier such as statement, preparation, extension,

antecedent, and consequent. The identification of a note grouping's function allows its logical re-constructions according to an ATN, which EMI is its first music implementation.

The pattern-matching component ensures the protection of musical signatures. Musical signatures are patterns occurring in different works from the same composer constituting essential elements that define and help and to recognize a composer's style. Since typically recombinancy is most successful when operated on a beat-to-beat level, pattern-matching in EMI protects musical signatures (patterns larger than one beat) from disintegration during recombinancy.

EMI is a software program that, although not intelligent, has produced aesthetically convincing new music. Intelligence seeks survival by the exercise of power over a surrounding environment. In composition, intelligence equals decision making. Every composition results from the selection of a finite set of constraints to operate on selected materials; even the most intuitive decision remains itself a decision, and consequently, a product of constraints.

Much of the fear and uneasiness surrounding EMI's acceptance and appreciation, lies on the unwillingness to understand it's purpose and design. EMI is a composer's tool. It can be said to "compose", not because it has the intelligence, the motive or the free will to do so, but because it performs the functions David Cope, the composer, programmed.

References

Berlioz, Hector, and Richard Strauss. 1991. *Treatise on Instrumentation*. Mineola, New York: Dover Publications, Inc.

Chomsky, Naom. 1976. *Syntactic Structures*. The Hague: Mouton.

Cope, David. 1991. *Computers and Musical Style*. Madison, Wisconsin: A-R Editions, Inc.

Cope, David. 1992. "On Algorithmic Representation of Musical Style". In *Understanding Music with AI*. M. Balaban, K. Ebcioglu, and O. Laske, editors. Cambridge, Massachusetts: MIT Press.

Cope, David. 1996. *Experiments in Musical Intelligence*. Madison, Wisconsin: A-R Editions, Inc.

Cope, David. 2001. *Virtual Music*. Cambridge, Massachusetts: The MIT Press.

Cope, David. 2003a. "Computer Analysis of Musical Allusions". *Computer Music Journal* 27(1):11-28.

Cope, David. 2003b. *Computer Models of Musical Creativity*. Forthcoming.

Cowell, Henry. 1996. *New Musical Resources*. Cambridge, Great Britain: University Press.

Harrison, Lou. 1971. *The Music Primer*. New York: C. F. Peters Corp.

Harrison, Patrick. 1990. *Common Lisp & Artificial Intelligence*. New Jersey: Prentice Hall.

Hofstadter, Douglas. 1985. *Metamagical Themas*. New York: BasicBooks.

Loy, Gareth. 1989. "Composing with Computers". In M. Mathews and J. R. Pierce, eds. *Current Directions in Computer Music Research*. Cambridge, Massachusetts: The MIT Press. pp. 292-396.

Marissen, Michael. 1999. *The Social and Religious Designs of J. S. Bach's Brandeburg Concertos*. Princeton, New Jersey: Princeton University Press.

D'Arezzo, Guido. 1978. "Micrologus". In Claude Palisca, ed. *Hucbald, Guido, and John on Music*. New Haven: Yale University Press.

Roads, Curtis. 2002. *Microsound*. Cambridge, Massachusetts: The MIT Press.

Schillinger, Joseph. 1948. *The Mathematical Basis of the Arts*. New York: Philosophical

Library.

Stockhausen, Karlheinz. 1961. ".....How Time Passes.....". *Die Reihe* (3): 10-40.

Stockhausen, Karlheinz. 2001. *Komposition-Kurs über Lichter-Wasser*. Kürten: Stockhausen-Verlag.

Tanimoto, Steven. 1987. *The Elements of Artificial Intelligence*. Maryland: Computer Science Press.

Wright, Craig. 1994. "Dufay's Nuper Rosarum Flores, King Solomon's Temple, and the Veneration of the Virgin". *Journal of the American Musicological Society* vol. XLVII, (3): 395-441.

Appendix 1

This appendix documents an e-mail thread originated in an electronic mailing list devoted to music theory, where unexpectedly David Cope's work became the subject of discussion**.** The e-mails shown here constitute a genuine sample of the understanding, acceptance and appreciation of his work in the beginning of the 21th century.

With the exception of David Cope's, all the other interveners true names, as well as their e-mail addresses and other information that potentially could lead to their identification were omitted. All the remaining text, including typographical errors, was left unedited. For the sake of clarity, each e-mail message is reprinted from the top of a new page, leaving in some cases large blank spaces in between.

The flavor of people's comments and the ideas they defend, will surely benefit from aging. It was included here for those who, later on, may get curious about the way of thinking about algorithms in music around 2002.

>Date: Thu, 4 Jul 2002 20:51:39 -0700 (PDT)
>
>There have been several attempts of computer-assisted Schenkerian
>analysis: by Michael Kassler (1964, 1975a, 1975b, 1977), by Robert
>E. Frankel, Stanley J. Rosenschein, and Stephen W. Smoliar (1976, 1978),
>and by Smoliar alone (1977, 1980). As far as I remember, David Cope's
>LISP-based system "Experiments in Musical Intelligence" (EMI) is
>partially based on Schenkerian analysis (Cope published several
>books on that since 1991).
>
>However, as far as I know, none of the approaches uses MIDI-
>files. One of the main reasons is the limitation of the MIDI code
>for music theoretical research (at least in certain aspects). This
>is also the reason for the development of an extension to the MIDI
>code by Peer Sitter (see Sitter 2000).
>
>Literature:
>
>Deliege, Celestin. 1984. "Some Unsolved Problems in Schenkerian Theory,"
>Musical Grammars and Computer Analysis, ed. by Mario Baroni and Laura
>Callegari. Firenze: Leo S. Olschki. 71-82.
>
>Frankel, Robert E., Stanley J. Rosenschein, and Stephen W. Smoliar. 1976. "A
>LISP-Based System for the Study of Schenkerian Analysis," Computers in the
>Humanities X (1976): 21-32.
>
>Frankel, Robert E., Stanley J. Rosenschein, and Stephen W. Smoliar. 1978.
>"Schenker's Theory of Tonal Music - its Explication through Computational
>Processes," International Journal of Man-Machine Studies X (1978): 121-138.
>

>Hughes, Matt. 1977. "A Quantitative Analysis," Readings in Schenkerian
>Analysis, ed. by Maury Yeston. New Haven: Yale University Press. 144-164.
>
>Kassler, Michael. 1964a. A Report of Work, Directed Toward Explication of
>Schenker's Theory of Tonality, Done in Summer 1962 as the First Phase of a
>Project Concerned with the Applications of High-Speed Automatic Digital
>Computers to Music ana Musicology. Princeton, N.J.: Princeton University Music
>Department. (Mimeographed.)
>
>_____. 1975a. "Explication of Theories of Tonality," Computational
>Musicology Newsletter II/1 (1975): 17.
>
>_____. 1975b. Proving Musical Theorems I: The Middleground of Heinrich
>Schenker's Theory of Tonality. Technical Report No. 103. Sydney: The
>University
>of Sydney, Basser department of Computer Science.
>
>_____. 1977. "Explication of the Middleground of Schenker's Theory of
>Tonality," Miscellanea Musicologica 9 (1977): 72-81.
>
>Schuler, Nico. Methods of Computer-Assisted Music Analysis: History,
>Classification, and Evaluation. Ph.D. dissertation. East Lansing:
>Michigan State University, 2000.
>
>Sitter, Peer. 2000. Computergestutzte Arbeitsmethoden in der
>Musikwissenschaft:
>ein Beitrag zu ihrer Entwicklung. Osnabruck: Universitätsverlag Rasch.
>
>Smoliar, Stephen William. 1977. "SCHENKER: A Computer Aid for Analyzing Tonal
>Music," SIGLASH Newsletter X/1-2 (1977): 30-61.

>

>_____. 1980. "A Computer Aid for Schenkerian Analysis," Computer Music

>Journal IV/2 (1980): 41-59.

>

>Best regards,

>Mr. X

>Date: Fri, 5 Jul 2002 04:40:48 -0700 (PDT)

>Thanks very much for these references.

>

>Mr. X wrote:

>>

>As far as I remember, David Cope's

>> LISP-based system "Experiments in Musical Intelligence" (EMI) is

>> partially based on Schenkerian analysis (Cope published several

>> books on that since 1991).

>>

>I wonder if there is anyone here who understands EMI suffciently well

>to explain its schenkerian (or otherwise analytic) aspects?

>

>> However, as far as I know, none of the approaches uses MIDI-

>> files. One of the main reasons is the limitation of the MIDI code

>> for music theoretical research (at least in certain aspects).

>

>I personally don't find that. It's true that MIDI isn't as rich as

>notation but insofar as it can completely represent a performance it

>invites analysis at a de facto listening level. Could you summarize

>Sitter's objections?

>best wishes,

>Mr. Y

>Date: Sat, 6 Jul 2002 19:56:03 -0700 (PDT)

>On 7/5/02 11:41 AM, "Mr. Y" wrote:
>> Mr. X wrote:
>>> As far as I remember, David Cope's
>>> LISP-based system "Experiments in Musical Intelligence" (EMI) is
>>> partially based on Schenkerian analysis (Cope published several
>>> books on that since 1991).
>>>
>> I wonder if there is anyone here who understands EMI suffciently well
>> to explain its schenkerian (or otherwise analytic) aspects?
>
>David Cope's most recent book is _Virtual Music: Computer Synthesis of
>Musical Style_ (MIT Press), but you may also find good answers to your
>question in _Computers and Musical Style_ (A-R Editions).
>
>EMI takes (as input) a number of pieces in a given style and returns (as
>output) pieces purported to be in that style. Cope has been working on this
>program for many years. The results are impressive.
>
>In order to convert the input pieces into a form that can be used by the
>computer, they have to be "pre-digested". This means that Cope analyzes
>them and that the computer is actually fed analyzed pieces. Cope notes that
>the method of analysis he uses (his SPEAC technique) tags the functions of
>musical material on multiple levels. And he cites Schenker's theories as an
>important source of his SPEAC technique.
>
>Furthermore, EMI uses Augmented Transition Networks (ATNs) to produce its
>pieces. An ATN may be regarded as a generative grammar that involves
>hierarchical relationships and transformation operations. The parallels

>with Schenker's ideas are striking.

>>Mr. Z

>Date: Sun, 7 Jul 2002 05:31:18 -0700 (PDT)

>07.07.2002 4:53 Uhr, Mr. Z:

>

>> EMI takes (as input) a number of pieces in a given style and returns (as

>> output) pieces purported to be in that style. Cope has been working on this

>> program for many years. The results are impressive.

>

>can you give a more detailed example?

>which kind of pieces does the program creates?

>how complex they are, and how long?

>(normally, attempts to reconstruct a musical language by a computer-software

>work fine with simple models (chorales or something else) but if you try to

>create a more complex structure, great problems emerge.

>

>Mr. W

>Date: Sun, 7 Jul 2002 15:40:38 -0700 (PDT)

>Mr. Y schrieb:
>
>>
>>> I personally don't find that. It's true that
>>> MIDI isn't as rich as notation but insofar
>>> as it can completely represent a performance
>>> it invites analysis at a de facto listening
>>> level. Could you summarize Sitter's
>>> objections?
>
>To which Mr. X antwortete:
>
>> The Standard MIDI-File Format (SMF-Format)
>> contains performance data, not audio data.
>>
><snip>
>
>This is an excellent point. The sound that
>results from the playing of a particular MIDI
>note is produced by a synthesizer program or
>the playing of a sample. The quality, in the
>senses both both of (1) characteristics of the
>sounds and (2) the "good" or "poor" fidelity
>to the instrument it is to represent, can vary
>markedly from one MIDI module, sound card, or
>instrument to another. Important information
>such as the mixture of overtones is virtually
>impossible to indicate with standard MIDI, let

>alone subtle characteristics of a particular
>approach to intonation that demonstrate its
>inadequacy with even some kinds of performance
>data.
>
>It calls to mind player piano "performances"
>where, despite the gratifying nuances sometimes
>captured, much depends on the selection and
>preparation of the reproducing instrument.
>
>Mr. T

>Date: Sun, 7 Jul 2002 15:50:27 -0700 (PDT)

>

>Mr. W wrote:

>>

>> 07.07.2002 4:53 Uhr, Mr. Z:

>>

>> > EMI takes (as input) a number of pieces in a given style and returns (as

>> > output) pieces purported to be in that style. Cope has been

>>working on this

>> > program for many years. The results are impressive.

>>

>> can you give a more detailed example?

>> which kind of pieces does the program creates?

>> how complex they are, and how long?

>

>You can listen to them here:

>

>http://arts.ucsc.edu/faculty/cope/Worklist.html

>

>> (normally, attempts to reconstruct a musical language by a computer-software

>> work fine with simple models (chorales or something else) but if you try to

>> create a more complex structure, great problems emerge.

>

>

>Well, you be the judge.

>

>-- Mr. Y

>Date: Mon, 8 Jul 2002 09:17:34 -0700 (PDT)

>Dear List Members,
>experimental studies on the interplay of analysis and performance may have
>two directions, namely (1) to measure performances in order to interpret
>performance data as traces of analytical structures and (2) to create
>artificial - but analytically controlled - performances in order to
>evaluate them. For two reasons I would like to recall a software project
>RUBATO developed by Guerino Mazzola and Oliver Zahorka (University of
>Zurich), supporting the second type of experiments:
>
>(1) the conceptual framework behind that program includes fruitful ideas
>concerning the modelling of analytical and performance structures (i.e. on
>a meta level).
>(2) to make those who are invloved in the implementation of specific
>analytical approaches aware about the possibility to test their theory
>beyond music-theoretical discourse also through artificial performance
>experiments.
>
>In reply to Mr. Y I would suggest to distinguish between a
>representation of a performance result (e.g. through a MIDI-file) and a
>performance transformation, namely a mapping of symbolic data into a
>ramified shaping process that finally results in a physical, gestural or
>technological description of a performance. The symbolic data includes
>notes and all kinds of "primavista" predicates of a score as well as
>analytical predicates. Central to the RUBATO-software is a quantization of
>symbolic predicates into so called "weights" which can be used in various
>ways to shape a performance. The artificial rehearsel processes are encoded
>into so called "stemmata", i.e. a ramified trees of refined shaping
>operations. A survey of the original RUBATO programm as well as the program

>(for NEXTSTEP) is avilable at (...).

>Mr. Q

>Date: Thu, 18 Jul 2002 13:02:30 -0700 (PDT)

>

> > At the risk of being overly cryptic, and paraphrasing Wittgenstein: if a

>> computer could talk, we wouldn't be able to understand it.

>

>

>That's precisely why programs like EMI pass turing tests.

>

>-- Mr. Y

>Date: Thu, 18 Jul 2002 19:34:12 -0700 (PDT)

>>On 7/18/02 1:03 PM, "Mr. Y" wrote:
>>> At the risk of being overly cryptic, and paraphrasing Wittgenstein: if a
>>> computer could talk, we wouldn't be able to understand it.
>> That's precisely why programs like EMI pass turing tests.
>
>I am not aware of a computer program that artificial-intelligence
>researchers would describe as having passed the Turing Test. The Turing
>Test is a game in which a human judge sits at two computer screens -- one
>connected to a keyboard operated by another (unseen) human, and the other
>connected to a computer program -- and then by typing questions to both, and
>seeing their responses on the two screens, tries to determine which screen
>is controlled by a computer and which is controlled by a human. If the
>human judge can't tell which screen is operated by the computer program,
>then the program passes the Turing Test.
>
>I think that the question "What should a 'musical Turing test' be like?" is
>an unanswered but intriguing question. And I would be interested to hear
>what contributors to this list think would be a good answer to that
>question.
>Mr. Z

>Date: Thu, 18 Jul 2002 20:26:41 -0700 (PDT)
>
>At 10:33 PM 7/18/2002, Mr. Z wrote:
>>I am not aware of a computer program that artificial-intelligence
>>researchers would describe as having passed the Turing Test. The Turing
>>Test is a game in which a human judge sits at two computer screens -- one
>>connected to a keyboard operated by another (unseen) human, and the other
>>connected to a computer program -- and then by typing questions to both, and
>>seeing their responses on the two screens, tries to determine which screen
>>is controlled by a computer and which is controlled by a human. If the
>>human judge can't tell which screen is operated by the computer program,
>>then the program passes the Turing Test.
>>
>>I think that the question "What should a 'musical Turing test' be like?" is
>>an unanswered but intriguing question. And I would be interested to hear
>>what contributors to this list think would be a good answer to that
>>question.

>Douglas Hofstadter (of _Goedel, Escher, Bach_ fame) gave a lecture at UMass
>a couple years ago in which he discussed Cope's work with EMI. The key
>point of the lecture was two pairs of recordings: a Bach invention, an EMI
>invention in the style of Bach, a Chopin prelude, and an EMI prelude in the
>style of Chopin. Hofstadter played the pieces in pairs, and polled the
>audience (made up of musicians and computer scientists) as to which one was
>the real Bach or Chopin and which one was composed by EMI. The results
>were abysmal; the better part of the audience could not tell the
>difference, even though it was fairly obvious. (The computer-composed
>works didn't have the same economy of material or subtlety of variation,
>though the passagework was spot on.)
>Mr. L

>Date: Fri, 19 Jul 2002 09:48:01 -0700 (PDT)

>

>Mr. Y wrote:

> >...programs like EMI pass turing tests.

>

>To which Mr. Z replied:

>>I am not aware of a computer program that artificial-intelligence

>>researchers would describe as having passed the Turing Test. The Turing

>>Test is a game in which a human judge sits at two computer screens -- one

>>connected to a keyboard operated by another (unseen) human, and the other

>>connected to a computer program -- and then by typing questions to both, and

>>seeing their responses on the two screens, tries to determine which screen

>>is controlled by a computer and which is controlled by a human. If the

>>human judge can't tell which screen is operated by the computer program,

>>then the program passes the Turing Test.

>>

>>I think that the question "What should a 'musical Turing test' be like?" is

>>an unanswered but intriguing question. And I would be interested to hear

>>what contributors to this list think would be a good answer to that

>>question.

>

>While this is quite literally the test described by Turing, the underlying

>criterion is more general: that you cannot tell a human from a computer if

>they communicate to you through the same medium.

>

>In the case of Turing's original test, the medium was text on a screen --

>one of the only media available to a computer in Turing's day. In EMI's

>case, the medium is sheet music provided to a performer who plays it to an

>audience.

>

>So I think the spirit of the Turing test is preserved in Mr. Y's

>analogy.

>

>Best wishes,

>Mr. J

>Date: Fri, 19 Jul 2002 12:46:47 -0700 (PDT)

>Mr. A said:
>
>> >> At the risk of being overly cryptic, and paraphrasing Wittgenstein: if a
>> >> computer could talk, we wouldn't be able to understand it.
>
>To which I said:
>
>> > That's precisely why programs like EMI pass turing tests.
>
>
>Mr. L and Mr. J clarified the Turing test part of my
>remark, so maybe I should clarify the rest of it.
>
>As everyone now knows, David Cope wrote a program called EMI that
>produces
>pieces "in the style of other composers," eg Rachmaninoff and Mozart.
>Some people
>are unable to tell whether a given work is machine-composed or by one of
>the
>greats whose "style" it emulates.
>
>Mr. A was making a quip based on (as I assume everyone knows, but you
>never know)
>the Wittgenstein adage, "if a lion could speak, we wouldn't be able to
>understand
>it," because the lion's experience is completely unlike ours, whereas,
>according
>to Wittgenstein, understanding language means understanding the world in

>which the
>speaker lives. We could never understand what it is like to be a lion.
>
>The computer has no experience unless you believe that "strong AI" is
>possible,
>ie, that one day, by dint of extremely complicated programs, a computer
>can become literally conscious.
>
>I took Mr. A's remark in a different way though. When we listen to a
>piece created
>by EMI, we're not usually listening relative to how it was actually
>constructed,
>unless we happen to understand exactly how the program works. We're
>listening
>according to our idea of what things happen when someone composes a
>piece
>of music -- that they may be expressing themselves, and so forth. We
>can't
>understand this computer music *as* computer music, so we impute to it
>a human intentionality, which of course it completely lacks.
>
>Of course this misunderstanding is no doubt a big part of what listening
>to music
>is all about, anyway.
>
>-- Mr. Y

>Date: Sat, 20 Jul 2002 18:55:30 -0700 (PDT)

>Mr. L wrote:
>"The key point ... was two pairs of recordings: a Bach invention, an EMI
>invention in the style of Bach, a Chopin prelude, and an EMI prelude in the
>style of Chopin. ... [T]he audience [was] made up of musicians and computer
>scientists ... [T]he better part of the audience could not tell the
>difference, even though it was fairly obvious. (The computer-composed works
>didn't have the same economy of material or subtlety of variation ...)"
>
>Maybe now programmers will program the mentioned ("fairly obvious")
>features. Recall the development of chess computers. If something has been
>understood (has become part of theory) then it can be programmed, no?
>
>Ms. R

>Date: Sat, 20 Jul 2002 20:28:12 -0700 (PDT)

>

>At 06:54 PM 7/20/02 -0700, Ms. R wrote:

>>Maybe now programmers will program the mentioned ("fairly obvious")

>>features. Recall the development of chess computers. If something has been

>>understood (has become part of theory) then it can be programmed, no?

>

>Excellent question. My answer: no. At least not in the sense intended

>by the "Turing Test."

>

>(I haven't been following this thread closely so if someone else has

>already come up with a similar argument, I apologize.)

>

>The Turing test, as I understand it, is a test of artificial intelligence,

>NOT programming skill. Artificial intelligence implies the existence of

>some entity, a computer or computing machine of some sort which is claimed

>to possess intelligence. In my view such claims are absurd. Patently so.

>

>Kasparov was NOT defeated by "a computer." It would be closer to the

>truth to say he was defeated by a "computer program." Closest to the

>truth would be the assertion that he was defeated by a huge team of chess

>experts and computer programmers working for many years with huge budgets

>to figure out a way to beat him, using the fantastic speed and

>computational power of high performance computers as a *tool* to assist

>them in *their* effort. In other words he was not so much defeated as

>gang raped. How is this a test of "artificial intelligence"?

>

>I have written several algorithmic musical (and visual) works using a

>computer as a tool. These works were NOT written "by a computer." They

>were written by me. I programmed the computer specifically to

>automatically produce the results I wanted. All the intelligence, and
>creativity, were mine -- ALL of it. NONE of it was produced by "the
>computer."
>
>The hobby of imitating the work of renowned composers has a long history in
>Western society. Many many people, performers as well as composers, have
>succeeded quite well over the years in convincing various people that there
>was "no difference" between such imitations and "the real thing."
>
>I listened with great interest to the MIDI examples presented on this
>list. The Bach were more convincing than the others (possibly because Bach
>lends himself more readily to MIDI than, say, Chopin). And some of these
>imitations were quite impressive. But as far as I can see, what we have
>here is, again, as in the Chess example, just a matter of various very
>skillful musicians with a flair for this sort of thing, teaming up with
>programmers to use computers *as a tool* to produce works for which they,
>NOT "the computer" ought to receive credit.
>
>The Turing test is no test at all because it can never distinguish between
>results produced by experts using the computer *as a tool* and results
>produced by an "intelligent machine."
>
>To really speak of true artificial intelligence one would have to develop a
>program, or programming system developed strictly as a general purpose
>device, with NO expert input whatsoever -- and then hope the thing would
>eventually be "smart" enough to, say, re-invent calculus (or even
>arithmetic), figure out ON ITS OWN how to write a passable two part
>invention, play a meaningful game of chess, etc. Don't hold your breath.
>Mr. O

>Date: Sun, 21 Jul 2002 08:09:58 -0700 (PDT)

>Mr. O wrote:
>
>> But as far as I can see, what we have
>> here is, again, as in the Chess example, just a matter of various very
>> skillful musicians with a flair for this sort of thing, teaming up with
>> programmers to use computers *as a tool* to produce works for which they,
>> NOT "the computer" ought to receive credit.
>
>One small correction: the program was written, as far as I know,
>entirely
>by David Cope, who is a composer.
>
>If a machine makes art, of course the real artist is the programmer,
>not the machine. If, however, at some point a program is developed that
>is sophisticated enough to simulate what it is like to hear music,
>and which possesses an adaptive network of theoretical models, surfs the
>web looking
>for music which it studies and classifies, is able to construct higher
>order metamodels of composition which can be
>instantiated as compositions, then such a program might be able to
>compose
>something completely unexpected and new of which it might rightly be
>said, "this
>was created by an AI."
>
>I don't think that can be said of EMI's music, though -- Smoliar raised
>a decisive point in his (...) review of an EMI concert
>(found here:

>(...)

>asking about the role of the ATN:

>

>"...in designing specifications with

>ATN representations, Cope is the one doing the composing,

>even if EMI is actually "generating the output.""

>

>-- Mr. Y

>Date: Mon, 22 Jul 2002 18:29:42 -0700 (PDT)

>With all due respect to David Cope (who was one of my theory teachers), the
>EMI project appears to have been promoted with considerable modesty
>regarding his own contribution while the role of the program appears to be
>overstated. If I understand the program correctly, it's an expert system,
>an elaborate form of the classical-era dice game for generating music from a
>collection of randomly recombined but ordered fragments. Such a system
>depends entirely upon a analysis and generation of suitable fragments. In
>this case, it is Cope's own analysis of existing works of music, which is
>used to generate a library of source material, which is then recombined
>through a process involving a random element, folowing a formal procedure
>also defined by analysis. Similary, the final, public, output of the program
>is a selection made by Cope from a number of runs through the program.
>
>The critical issue here would seem to be the quality of the analysis, but I
>am afraid that Cope does not make his techniques as explicit as one would
>like. In fact, by foregrounding the "intelligent" aspects of his program,
>he is distracting our attention away from this critical point, perhaps even
>using the "computer generated" aspect to pre-empt criticism of work which is
>very much his own.
>
>I note that Cope's most recent volume is entitled "Virtual Music". Again, I
>suspect that he is using the "computer generated" aspect of the work to
>lesson expectations, in this case, even presenting the output to the public
>as "virtual", something _less_ than music. But the output is inevitably
>music, with nothing virtual about it. Isn't this just a way of avoiding the
>issue of whether said ouput is good or bad music?
>Mr. F

>Date: Mon, 22 Jul 2002 19:22:23 -0700 (PDT)

>>On 18 Jul 2002, Mr. L wrote:
>>
>> Douglas Hofstadter (of _Goedel, Escher, Bach_ fame) gave a lecture atUMass
>> a couple years ago in which he discussed Cope's work with EMI. The key
>> point of the lecture was two pairs of recordings: a Bach invention, anEMI
>> invention in the style of Bach, a Chopin prelude, and an EMI prelude inthe
>> style of Chopin.
>
>Hofstadter also gave this lecture at Eastman/U of Rochester, with the
>pieces performed by an Eastman faculty member (Norm Carey). The musicians
>in the audience correctly identified the real from the EMI, the computer
>scientists did not (as a generality)[2] . What was very interesting to me was
>that Hofstadter claimed that the EMI program was resurrecting the souls of
>these composers, by imitating their compositional styles so well. He was
>seriously concerned about this conclusion, and thought the program was
>therefore possibly unethical. I disagree with this conclusion,
>particularly that a
>person's soul or persona consists only of the works that person has
>achieved, especially only one type of works (i.e. composition).
>Chopin did not consist only of his piano music, or even of his entire
>oeuvre.
>
>The other point is that EMI only imitates the compositional style
>that was clearly evident from the existing works. This does not
>allow for the evolution of style that a truly new Chopin or Bach
>piece might show.
>

[2] It may be of interest for the reader to compare Mr. S' account of Hofstadter's lecture with Alla Persons statement as published in (Cope 2001), pp. 66-67.

>I agree with others who already commented that Cope is the real

>composer of these pieces, not EMI. Cope even had to specially

>program in the general partwriting rules and counterpoint rules when

>getting EMI to imitate Bach, as the music EMI was learning bent too

>many of these rules.

>

>Mr. S

>Date: Mon, 22 Jul 2002 19:53:31 -0700 (PDT)

>

>At 06:29 PM 7/22/02 -0700, Mr. F wrote:

>>With all due respect to David Cope (who was one of my theory teachers), the

>>EMI project appears to have been promoted with considerable modesty

>>regarding his own contribution while the role of the program appears to be

>>overstated.

>

>Despite the critical tenor of my last post regarding claims which, to me,

>are patently absurd, I feel nevertheless that this sort of research can be

>extremely important and should definitely be continued. It represents, in

>fact, a line of thought initiated by one of my own favorite profs, Lejarin

>Hiller, one of the fathers of computer music, who developed an automated

>system for producing species counterpoint, among other things.

>

>I am NOT a believer in "artificial intelligence", but I AM a believer in

>the computer as a tool, of research and also composition. I believe in

>algorithmic composition and I believe in algorithmic compositional

>imitation (if that's what it can be called). I think it makes an

>excellent test of ones theoretical/analytic approach to take it to the next

>step and have it generate imitations of the real thing. My only objection

>regards the sort of claims being made. Clearly the computer is and always

>will be a tool like any other. Computers do NOT have minds, they cannot

>"write" music or anything remotely like that. But they ARE just absolutely

>fantastic tools, which should certainly be pushed to their limits.

>

>Mr. O

>Date: Tue, 23 Jul 2002 13:16:56 -0700 (PDT)

>

>I appreciate the recent posts to this list regarding my work with

>Experiments in Musical Intelligence. I would, however, like to clear

>up a couple of misconceptions:

>

>(1) I do not pre-analyze the data which the program uses to compose -

>I have written four books which explain Emmy's algorithmic analytical

>processes and refer those interested to read these sources (I should

>add that the program is also far more than an elaborated

>Musikalisches W¸rfelspiel);

>

>(2) I place "By David Cope (with Experiments in Musical

>Intelligence)" on all of the printed music (see my website for PDF

>examples) and I have repeatedly stated (often in print) that

>Experiments in Musical Intelligence is *not* intelligent. I am at

>work now on a book called Computer Models of Musical Creativity which

>deals with whether or not the association network at the heart of the

>program is creative.

>

>I apologize for "lurking" in the background during this discussion. I

>enjoy reading this forum's posts but tend to shy away from

>contributing due to a lack of time. I sincerely hope that my sudden

>loss of anonymity does not discourage those of you still interested

>in discussing this work freely. I don't mind negativity toward my

>work in the least - in fact I expect it. I do, however, hope that any

>expressed opeinion, pro or con, is based on knowledge and not merely

>on speculation.

>All the very best,

>Dave Cope

>Date: Wed, 24 Jul 2002 11:08:47 -0700 (PDT)

>

>Dave Cope wrote:

>

>>

>> (1) I do not pre-analyze the data which the program uses to compose -

>> I have written four books which explain Emmy's algorithmic analytical

>> processes and refer those interested to read these sources

>

>I wonder whether Prof. Cope would care to throw in a few words

>concerning one of the points at issue in this discussion, namely the

>relationship between automated analysis and the construction of the

>ATN.

>

>I'll express this point simply since not everyone on this list is

>conversant with the techniques and problems of computerized

>composing.

>

>In your Leonardo article you discuss Deep Blue and you defend the idea

>of using a musical database for composition. I have no trouble with

>this. But there's, for me at least, an intuitive gap between the

>notion of a chess database and a musical database. Partly this has to

>do with the very lower order specifiability of musical "state," as

>opposed to the state of a chessboard. Two consecutive game states

>never differ by more than one move. Two consecutive moments in music

>could differ by a universe.

>

>So the problem would seem to be this. If I wanted to try to replicate

>EMI's success, I imagine I'd have to write a program that would

>analyze Mozart, eg, and deduce which passages are plausibly

>interchangeable, plausible in the sense that continuity would be
>roughly ensured, and the overall purposes of design wouldn't be lost.
>I can see how to do this for two or three carefully chosen pieces,
>with, let's say, rough structural equivalence.
>
>I would go about it as follows. I would look through the music I
>planned to submit to further analysis, designing a parse tree by hand,
>based on criteria I found in the music itself. I would attempt to
>generalize and I would construct branches that dealt with obvious
>exceptions to my generalisms. The program could then go about making
>its deductions, such as, "the theme 1 xyz branch consists of the
>following possible harmonic progressions, etc."
>
>Obviously in this situation I HAVE pre-analyzed the music in that its
>analysis is reflected in my parse tree. And it seems to me that if I
>wish to obviate the step of manually designing the parse tree for a
>given piece, the only alternative I can think of would be to attempt
>construct a universal grammar for musical structure, which seems like
>a very tall order indeed.
>
>So the question I have is this. Is the ATN in EMI tailored for
>piece-specificity via manual coding, or has it been sufficiently
>developed to generalize musical transitions between, say, something by
>Schumann and something else by Chopin? In other words, how "universal"
>is its grammar? Since you're explicit that EMI is not "intelligent" do
>I thereby understand it does not attempt to expand its parse-tree
>through "learning"?
>
>> I do, however, hope that any
>> expressed opeinion, pro or con, is based on knowledge and not merely

>> on speculation.

>

>I hope your appearance on this list will help favor that hope.

>

>best,

>

>-- Mr. Y

Appendix 2

List of Publications by David Cope.

Books:

Cope, David. 2001. *Virtual Music*. Cambridge, Massachusetts: The MIT Press.

———. 2000. *The Algorithmic Composer*. Madison, WI: A-R Editions.

———. 1997. *Techniques of the Contemporary Composer*. New York: Schirmer Books.

———. 1996. *Experiments in Musical Intelligence*. Madison, WI: A-R Editions.

———. 1991. *Computers and Musical Style*. Madison, WI: A-R Editions (national) and Oxford University Press (international).

———. 1970-1993. *New Directions in Music*. 7 editions. Prospect Heights, IL: Waveland Press.

———. 1977. *New Music Notation*. New York: Schirmer Books.

———. 1977. *New Music Composition*. New York: Schirmer Books.

Articles (since 1987):

Cope, David. 2003. "Computer Analysis of Musical Allusions." *Computer Music Journal* 27/1: 11-28.

———. 2002. "Computer Analysis and Composition Using Atonal Voice-Leading Techniques." *Perspectives of New Music* 40/1: 120-46.

———. 1999. "One Approach to Music Intelligence." *Intelligent Systems* 14/3: 21-5.

———. 1999. "Facing the Music: Perspectives on Machine-Composed Music." *Leonardo Music Journal* 9: 79-87.

———. 1998. "Signatures and Earmarks: Computer Recognition of Patterns in Music." *Computing in Musicology* 11: 129-138

———. 1997. "CUE." *Computer Music Journal*. 21/3: 20–37.

———. 1997. "Composer's Underscoring Environment." In *Proceedings of the International Computer Music Conference*. San Francisco: Computer Music Association.

———. 1993. "Virtual Music." *Electronic Musician* 9/5 : 80–85.

———. 1992. "A Computer Model of Music Composition." In *Machine Models of Music*, Stephan Schwanauer and David Levitt, eds. Cambridge, Massachusetts: MIT Press.

———. 1992. "On Algorithmic Representation of Musical Style." In *Understanding Music with AI*. M. Balaban, K. Ebcioglu, and O. Laske, editors. Cambridge, Massachusetts: MIT Press.

———. 1992. "Algorithmic Composition [re]Defined." In *Proceedings of the International Computer Music Conference*. San Francisco: Computer Music Association.

———. 1992. "Computer Modeling of Musical Intelligence in EMI." *Computer Music Journal* 16/2: 69–83.

———. 1991. "Recombinant Music." *Computer* 24/7: 22–28.

———. 1991. "Computer Simulations of Musical Style." In *Computers in Music Research*. Belfast, Ireland: Queen's University: 15–17.

———. 1990. "Pattern Matching as an Engine for the Simulation of Musical Style." In *Proceedings of the International Computer Music Conference*. San Francisco: Computer Music Association.

———. 1989. "Experiments in Musical Intelligence (EMI): Non-Linear Linguistic-based

Composition." *Interface* 18: 117–139.

———. 1988. "Music and LISP." *AI Expert* 3/3: 26–34.

———. 1988. "Music: The Universal Language." In *Proceedings of the First Workshop on Artificial Intelligence and Music*. St. Paul, MN: AAAI: 87–98.

———. 1987. "An Expert System for Computer-Assisted Music Composition." *Computer Music Journal* 11/4: 30–46.

CDs:

Virtual Bach. 2002. Centaur Records (complete CD of music by Experiments in Musical Intelligence).

Virtual Mozart. 2000. Centaur Records (complete CD of music by Experiments in Musical Intelligence).

Classical Music Composed by Computer. 1997. Centaur Records (complete CD of music by Experiments in Musical Intelligence).

Towers. 1997. Ensemble MW2 on Vienna Modern Masters (VMM 2024).

Bach by Design. 1993. Centaur Records (Complete CD of music by Experiments in Musical Intelligence).

Recombinant Music. 1991. Recording (CD) examples published with article in *Computer*.

Tapes:

Threshold and Visions and *Glassworks*. Smithsonian Institution (33452).

Navajo Dedications (Vortex, Rituals, Teec Nos Pos and *Parallax)*. Smithsonian Institution (33869).