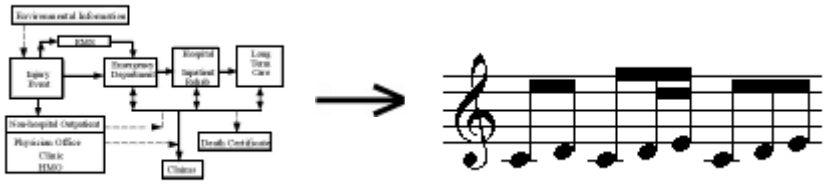
A Brief History of Algorithmic Composition



by
© john a. maurer iv
March, 1999

Table of Contents:

Introduction

"Since I have always preferred making plans to executing them, I have gravitated towards situations and systems that, once set into operation, could create music with little or no intervention on my part. That is to say, I tend towards the roles of planner and programmer, and then become an audience to the results" -Brian Eno (Alpern, 1995).

Algorithmic composition, sometimes also referred to as "**automated composition**," basically refers to "the process of using some formal process to make music with minimal human intervention" (Alpern, 1995). Such "formal processes," as we will see, have been familiar to music since ancient times. The title itself, however, is relatively new—the term "**algorithm**" having been adopted from the fields of computer science and information science around the halfway mark of the 20th century (Burns, 1997). Computers have given composers new opportunities to automate the compositional process. Furthermore, as we will explore, several different methods of doing so have developed in the last forty years or so.

To begin with the title itself, *Webster's* dictionary defines an "algorithm" simply as "a predetermined set of instructions for solving a specific problem in a limited number of steps." The "problem" composers are faced with, of course, is creating music; the "instructions" for creating this music according to the definition are "predetermined," suggesting that intervention on the part of the human composer is superceded once the compositional process itself is set into motion, as hinted at as well in the above Brian Eno quote. Thus, "automated composition" also suitably describes this kind of music, since "**automation**" refers to "anything that can move or act of itself."

I. Pre/Non-Computer Practices

- ancient Greeks, canon, Mozart, John Cage, serialism

The idea of utilizing formal instructions and processes to create music dates back in musical history as far back as the ancient Greeks. **Pythagoras** believed in a direct relation between the laws of nature and the harmony of sounds as expressed in music:

> "The word *music* had a much wider meaning to the Greeks than it has to us. In the teachings of Pythagoras and his followers, music was inseperable from *numbers*, which were thought to be the key to the whole spiritual and physical universe. So the system of musical sounds and rhythms, being ordered by numbers exemplified the harmony of the cosmos and corresponded to it" (Grout, 1996; italics added).

Thus, theoretical applications of numbers (i.e. "data," in a sense) and various mathematical properties derived from nature were the **formalisms**, or "algorithms," upon which the ancient Greek musicians had constructed their musical systems. **Ptolemy** and **Plato**, also, were two others who wrote about this practice. Ptolemy, the "most systematic of the ancient theorists of music," was also a leading astronomer of his time; he believed that mathematical laws "underlie the systems both of musical intervals and of the heavenly bodies," and that certain modes and even certain notes "correspond with particular planets, their distances from each other, and their movements" (Grout, 1996). This idea was also given poetic form by Plato in the myth of the "music of the spheres," the unheard music "produced by the revolutions of the planets" (Grout, 1996), and the notion was later invoked by writers on music throughout the Middle Ages, including Shakespeare and Milton (Grout, 1996).

These ancient Greek "formalisms," however, are rooted mostly in theory, and their strict application to musical performance itself is probably questionable since Greek music was almost entirely improvised (Grout, 1996). Thus, while Greek mathematical conjectures certainly created the musical system of intervals and modes with which the musician operated and probably also guided and influenced his/her performance practice in some ways, the musician was by no means entirely removed from the decision-making process. Ancient Greek music was not "algorithmic composition" in any pure sense, therefore, but it is undoubtably important historically in music for its tendency towards formal extra-human processes.

An extra layer of abstraction would later be acheived with the birth of "**canonic**" composition in the late 15th century:

> "The prevailing method was to write out a single voice part and to give instructions to the singers to derive the additional voices from it. The instruction or rule by which these further parts were derived was called a *canon*, which means 'rule' or 'law.' For example, the second voice might be instructed to sing the same melody starting a certain number of beats or measures after the original; the second voice might be an inversion of the first or it might be a retrograde [etc.]" (Grout, 1996).

These "rules" of imitation and manipulation are indeed the "algorithm" by which performers unfolded the music. In this case, then, as opposed to the previous one of the ancient Greeks, we can see a clear removal of the composer from a large portion of the compositional process: the composer himself only invents a kernel of music—a single melody or section—from which an entire composition is automatically constructed.

Mozart, too, used automated composition techniques in his *Musikalisches Wurfelspiel* ("**Dice Music**"), a musical game which "involved assembling a number of small musical fragments, and combining them by chance, piecing together a new piece from randomly chosen parts" (Alpern, 1995). This very simple form of "algorithmic" composition leaves creative decisions in the hands of chance, letting the role of a dice to decide what notes are to be used.

There are more modern examples, as well, of algorithmic composition without the use of the computer. **John Cage**, for example, like Mozart, utilized randomness in many of his compositions, such as in *Reunion*, performed by playing chess on a photo-receptor equipped chessboard: "The players' moves trigger sounds, and thus the piece is different each time it is perfomed" (Alpern, 1995). Cage also delegated the compositional process to natural phenomena, as in his *Atlas Eclipticalis* (1961), which was composed by laying score paper on top of astronomical charts and placing notes simply where the stars occurred, again delegating the compositional process to indeterminacy (Schwartz, 1993).

The **twelve-tone method** and **serialism**, furthermore, were movements of the post-World War II era that tried to completely control all parameters of music and to objectify and abstract the compositional process as much as possible. Decisions over everything from notes to rhythms to dynamic markings were often subject to pre-composed "series" and "matrices" of values, which, in effect, "automated" many of these parameters by determining the order in which each must occur in a piece. These series and matrices were, then, the "algorithms" that superceded the human creative process. Serialism can thus be labeled "algorithmic" or "automated" composition in a rather pure sense, especially when it strives to integrate as many musical parameters as possible. **Olivier Messiaen**'s 1949 piano etude, *Mode de valeurs et d'inensites*, for example, had a thirty-six pitch series, each pitch of which was given specific rhythmic, dynamic, registral, and attack characterstics with which to be used in the composition (Kostka, 1995).

* * *

II. Use Of The Computer

- 2 early pioneers: Lejaren Hiller, Iannis Xenakis
- 3 general approaches: stochastic, rule-based, artificial intelligence (AI)

Computers introduced incredible new capacities available for algorithmic composition purposes. **Ada Lovelace**, inventor of the "calculating engine," the precursor of computers, had this to say about the possibilities of automated composition (Alpern, 1995) in the 19th century:

> "Supposing, for instance, that the fundamental relations of pitched sound in the signs of harmony and of musical composition were susceptible of such expression and adaptations, the engine might compose elaborate and scientific pieces of music of any degree of complexity or extent" (Alpern, 1995).

And so it happened, as Lovelace had predicted, that the computer (or modern "calculating engine") brought scientists and composers together to construct such "elaborate" pieces of music out of new algorithmic programming methods.

The earliest instance of computer generated composition is that of **Lejaren Hiller** and **Leonard Isaacson** at the University of Illinois in 1955-56. Using the **Illiac** high-speed digital computer, they succeeded in programming basic material and stylistic parameters which resulted in the *Illiac Suite* (1957). The score of the piece was composed by the computer and then transposed into traditional musical notation for performance by a string quartet. What Hiller and Isaacson had done in the *Illiac Suite* was to (a.) generate certain "raw materials" with the computer, (b.) modify these musical materials according to various functions, and then (c.) select the best results from these modifcations according to various rules (Alpern, 1995). This "generator/modifier/selector" paradigm was also later applied to **MUSICOMP**, one of the first computer systems for automated composition, written in the late 1950s and early 1960s by Hiller and **Robert Baker**, which realized *Computer Cantata*: "Since [MUSICOMP] was written as a library of *subroutines*, it made the process of writing composition programs much easier, as the programmer/composer could use the routines within a larger program that suited his or her own style" (Alpern, 1995; italics added). This idea of building small, well-defined compositional functions—i.e. "subroutines"—and assembling them together would prove efficient and allow the system a degree of flexibility and generality (Alpern, 1995), which has made this approach a popular one, as we will see, in many algorithmic composition systems even into the present day.

Another pioneering use of the computer in algorithmic compostion is that of **Iannis Xenakis**, who created a program that would produce data for his "**stochastic**" compositions, which he had written about in great detail in his book *Formalized Music* (1963). Xenakis used the computer's high-speed computations to calculate various probability theories to aid in compositions like *Atrées* (1962) and *Morsima-Amorsima* (1962). The program would "deduce" a score from a "list of note densities and probabilistic weights supplied by the programmer, leaving specific decisions to a random number generator" (Alpern, 1995). "Stochastic" is a term from mathematics which designates such a process, "in which a sequence of values is drawn from a corresponding sequence of jointly distributed random variables" (*Webster's* dictionary). As in the previous example of the *Illiac Suite*, these scores were performed by live performers on traditional instruments.

With Xenakis, it should be noted, however, "the computer has not actually produced the resultant sound; it has only aided the composer by virtue of its high-speed computations" (Cope, 1984): in essence, what the computer was outputing was not the composition itself but material with which Xenakis could compose. In contrast, the work of Hiller and Isaacson attempted to simulate the compositional process itself entirely, completely delegating creative decisions to the computer.

Already in these first two examples—Xenakis and Hiller—we find two different methodologies that exist in computer-generated algorithmic composition: (1.) "**stochastic**" vs. (2.) "**rule-based**" systems. As we will see, there is also a third category, (3.) which we can label **AI**, or **artificial intelligence** systems.

Stochastic approaches, already somewhat touched upon, are the simplest. These involve **randomness** and can be as simple as generating a random series of notes, as seen already in the case of Mozart's *Dice Music* and in the works of John Cage, though a great amount of conceptual complexity can also be introduced to the computations through the computer with **statistical theory** and **Markov chains**. Basically, many of the creative decisions in the stochastic method are merely left to chance, essentially the same as drawing notes out of a hat. Another example of non-computer-oriented "stochastic" composition can be found in **Karlheinz Stockhausen**'s *Klaveirstucke XI* in that the sequence of various fragments of

music are to be performed by a pianist in random sequence. A different slant to usages of unexpectedness is that of applying **chaos theory** to algorithmic composition (Burns, 1997). These applications employ various nonlinear dynamics equations that have been deduced from nature and other chaotic structures such as fractals to relay different musical information:

> "In recent years [the '70s and '80s], the behaviour of systems of nonlinear dynamical equations when iterated has generated interest into their uses as note generation algorithms. The systems are described as systems of mathematical equations, and, as noted by Bidlack and Leach, display behaviours found in a large number of systems in nature, such as the weather, the mixing of fluids, the phenomenon of turbulence, population cycles, the beating of the human heart, and the lengths of time between water droplets dripping from a leaky faucet" (Alpern, 1995).

This is a large and mathematically complex field of algorithmic composition, and the author refers the interested reader to my website on the topic as well as to the article by Jeremy Leach ("Nature, Music, and Algorithmic Composition." *Computer Music Journal*, 1995) as good starting points for more in-depth investigation.

A second approach to algorithmic composition using the computer is that of "**rule-based**" systems and **formal grammars**: "An elementary example of a rule-based process would center around a series of tests, or rules, through which the program progresses. These steps are usually constructed in such a way that the product of the steps leads to the next new step" (Burns, 1997). Non-computer parallels to rule-based algorithmic composition that have been previously mentioned include the 15th-century canon of the Renaissance period as well as the post-WWII twelve-tone method and integral serialism. Rather than delegating decisions to chance as in the stochastic methods just described, rule-based systems pre-compose a "constitution," so to say, or a "grammer," by which the compositional process must behave once set into motion—"grammar" being a term borrowed from linguistic theory which designates the formal system of principles or rules by which the possible sentences of a language are generated (Burns, 1997). Like Hiller's MUSICOMP, these efforts usually take the form of a computer program or a unified system of subroutines, and often also involve databases of various rules either collected from compositional techniques of the past or newly invented. One example of using a "rule-based" method of algorithmic composition is that of **William Shottstaedt**'s automatic species counterpoint program that writes music based on rules from Johann Joseph Fux' *Gradus ad Parnassum*, a counterpoint instruction book from the early 18th-century aimed at guiding young composers to recreate the strictly controlled polyphonic style of Palestrina (1525-1594) (Grout, 1996):

> "The program is built around almost 75 rules, such as 'Parallel fifths are not allowed' and 'Avoid tritones near the cadence in lydian mode.' Schottstaedt assigned a series of 'penalties' for breaking the rules. These penalties are weighted based on the fact that Fux indicated that there were some rules that could never be broken, but others did not

> have to be adhered to as vehemently. As penalties accumulate, the program abandons its current branch of rules and backtracks to find a new solution" (Burns, 1997).

Another example is that of Kemal Ebcioglu's automated system called CHORAL which generates four-part chorales in the style of J. S. Bach according to over 350 rules (Burns, 1997).

One last unique approach that I found to algorithmic composition using the computer is that of **aritifical intelligence (AI)** systems. These systems are like rule-based systems in that they are programs, or systems of programs, based on some pre-defined grammar; however, AI systems have the further capacity of *defining their own grammar*—or, in essence, a capacity to "learn." An example of this is **David Cope**'s system called **Experiments in Musical Intelligence (EMI)**. Like the previous example of Shottstaedt and of Ebcioglu's CHORAL, EMI is based on a large database of style descriptions, or rules, of different compositional strategies. However, EMI also has the capactiy to *create its own grammar and database of rules*, which the computer itself deduces based on several scores from a specific composer's work that are input to it. EMI has been used to automatically compose music that evokes already somewhat successfully the styles of Bach, Mozart, Bartók, Brahms, Joplin, and many others.

Another interesting branch of AI techniques is that of "**genetic programming**," a very recent technique in the field of computer science for "**automatic programming**" of computers (Alpern, 1995). Rather than basing its grammar on scores input to the computer as in EMI, genetic programming *generates its own musical materials* as well as form its own grammar. The composer must also program a "critic" function, therefore, which then *listens* to the numerous automatically produced outputs at various stages of the processing to decide which are "fit" or suitable for final output (the composer having final say, then, as to which of these to discard and which to save). Below is a more in-depth description of the different processes involved in genetic programming methods:

> "[Genetic programming] is a method which actually uses a process of artificially-created natural selection to evolve simple computer programs. In order to perfrom this process, one uses a small set of functions and terminals, or constants, to describe the domain one wishes an evolved program to operate in. For example, if the human programmer wishes to evolve a program which can generate or modify music, one would give it functions which manipulate music, doing things such as transposition, note generations, stretching or shrinking of time values, etc. Once the functions have been decided on, the genetic programming system will create a population of programs which have been randomly generated from the provided function set. Then, a fitness measure is determined for each program. This is a number describing how well the program performs in the given problem domain. Since the initial programs are randomly generated,

> their performance will be very poor—however, a few programs are likely to do slightly better than the rest. These will be selected in pairs, proportionate to their fitness measure, and then a new population of programs will be created from these individuals, and the whole process will be repeated, until a solution is reached (in the form of a program which satisfies the critic), or a set of number of iterations has passed. Operations which may be performed in generating this new population include reproduction (passing an individual program on into the next generation unchanged), crossover (swapping pieces of code between two 'parent' programs in order to create two unique 'children'), mutation, permutation, and others" (Alpern, 1995).

The composer, thus, provides the system with a library of functions, or subroutines, as we have already seen in the case of Hiller's MUSICOMP and other systems, which can do various things to the generated musical materials: however, in this case, the composer does not define the way in which these functions will be used—the composer merely defines for the computer what is desirable in an output (i.e. designs a "critic") and the computer in turn tries to automatically achieve these results using the provided subroutines. This form of "algorithmic composition," thus, (using AI or genetic programming) can be seen as an extreme case, abstracting itself *even from its own "algorithm"* since the output it produces *as well as the formal process by which it performs* is automatically constructed.

Besides the three various methods of algorithmic composition using the computer that I have described—stochastic, rule-based, and AI—further distinction also occurs in the type of musical output different algorithmic composition systems produce. Some systems specify **score information** only (i.e. pitch, duration, and dynamic material) to be realized by whatever acoustic or electronic instruments, as seen already in the early cases of Hiller and Xenakis and which is also true in the case of Cope's EMI compositions (the MIDI scores of which are fed into a Disclavier or other MIDI sound device for output) and most others mentioned in this paper. Other systems, however, do not create scores and focus instead on electronic **sound synthesis** or manipulation of recorded sounds (i.e. ***musique concréte***), or on a combination of these activities. Sound synthesis algorithms, furthermore, "have been used in a variety of ways, from the calculation of complex waveforms (building sounds), to the evolution of timbre development over time" (Burns, 1997). A last approach is to combine both score and electronic sound synthesis in the system's output, controlling both structural content and its own timbral realization.

Closing

As for new developments in the field today, **automatic *listening*** programs seem to be a new trend and focus: not only does the computer automatically compose, it is also being designed to listen and *respond* to music being performed around it, a field of music that is labelled "**live electronics**":

> "Another tendency is to use the computer as an accompanist who listens to what is being

> played and responds appropriately in real-time. Here, the human input is used to generate rules on which the machine will base its output. This is seen in such programs as Cypher (Rowe, 1993) and IBL-Smart (Widmer, 1994)" (Jacob, 1996).

Another slant on "automatic listening" is that of Jonathan Berger and Dan Gang (Berger, 2004) who have created computational models of perception and cognition of music using AI approaches that have given new insights into the creative properties inherent in listening and, furthermore, to the process of creativity itself . These new techniques could also potentially improve algorithmic composition, it would seem, since the "critic" functions that we have seen in examples of genetic programming could gain much improvement from their insights into how humans listen to music: the computer could, then, better judge itself as to the quality of its output.

Aesthetically speaking, the more recent and complicated brands of algorithmic composition that utilize the computer are still in their infancy and much improvement is, perhaps, left to be desired. As Cope himself remarks, for example, in regard to Hiller's early experiments with the Illiac, many "directions in 'computer control' have not proven to be great artistic successes" (Cope, 1984). These various new directions with the computer (i.e. stochastic, rule-based, and AI) have been, nevertheless, extraordinarily important for they have "opened the door to new vistas in the expansion of the computer's development as a unique instrument with significant potential" (Cope, 1984). They have also broadened our conception of music and how it can be realized, as well as given us rare opportunities to test different compositional theories, listen to them in action, and also then try to improve upon them. Thus, not only has the composer been able to do new things with the computer through algorithmic means, s/he has also been able to investigate him/herself more closely and to gain new insights not only into his/her own compositional processes but into the techniques and strategies of composers throughout history. These experiments are thus intellectually stimulating and important in their own right for these reasons, and time will tell whether they cannot also produce many "great artistic successes."

~end~

References

- Alpern, Adam (1995), Techniques for algorithmic composition of music. http://alum.hampshire.edu/~adaF92/algocomp/algocomp95.html. Hampshire College.
- Berger, Jonathan (2004), Who cares if it listens? An essay on creativity, expectations, and computational modeling of listening to music. In *Virtual Music: Computer Synthesis of Musical Style*. David Cope. MIT Press. 584 pp.
- Burns, Kristine, H. (1997), Algorithmic composition, a definition. http://music.dartmouth.edu/~wowem/hardware/algorithmdefinition.html. Florida International University.
- Cope, David (1984), *New Directions in Music*. 4th ed. W. C. Brown: Dubuqe, Iowa. 259 pp.
- Grout, Donald Jay and Claude V. Palisca (1996), *A History of Western Music*. 5th ed. W. W. Norton & Company: New York. 843 pp.
- Jacob, Bruce L. (1996), Algorithmic composition as a model of creativity. http://www.ee.umd.edu/~blj/algorithmic_composition/algorithmicmodel.html. University of Michigan.
- Kostka, Stefan and Dorothy Payne (1995), *Tonal Harmony: With an Introduction to*

Twentieth-Century Music. 3rd ed. McGraw-Hill, Inc.: San Francisco. 688 pp.
- Leach, Jeremy and John Fitch (1995), Nature, music, and algorithmic composition. *Computer Music Journal*. 19(2): 23-33.
- Schwartz, Elliott and Daniel Godfrey (1993), *Music Since 1945: Issues, Materials, and Literature*. Schirmer Books: New York. 560 pp.
- Xenakis, Iannis. *Formalized Music: Thought and Mathematics in Composition*. Indiana University Press. 387 pp.

Note: An extensive algorithmic composition bibliography has been compiled by Tobias Kunze at Stanford University (1998) which can be found online at http://ccrma.stanford.edu/~tkunze/res/algobib.html.

top of page
