Larry Polansky

Works for Performers and Live Interactive Computer

New Langton Arts
San Francisco
Friday April 22nd, 1988

with guests
John Bischoff, George Brooks, Phil Burk, Jody Diamond, Ann LaBerge, Jeanne Parson, and Melody Sumner

---

I

בְּרֵאשִׁית **(B'rey'sheet) (In the beginning...) (Cantillation Study #1)** for voice and interactive computer; Jody Diamond, voice (1984)

II

וְלִמְשֹׁל **(V'leem'shol: And to rule...) (Cantillation Study #2) (for five flutes)** for Ann LaBerge and David Rosenboom (1983); revision to extended just intonation (with Ann LaBerge) (1987-88)

III

Cocks crow, dogs bark, this all men know. But even the wisest cannot tell, why cocks crow, dogs bark, when they do for computers, computer controlled signal processor, performers, and text (1986); revision (with Melody Sumner, text, and John Bischoff) (1988)

IV

17 Simple Melodies of the Same Length (for Daniel Goode) for musician and computer. George Brooks, tenor saxophone; Jeanne Parson and Phil Burk, computers and synthesizers (1987)

V

Simple Actions (2) (Distance Music VI) for computer and performer (1987)

Thanks to Tony Gnazzo for sound and recording; Jody Diamond for program design; Phil Burk and Robert Marsanyi for technical assistance.

which execute a user-defined function at a certain time interval (the time interval can of course be changed in real-time by the job itself as part of its function!).

As in *Cocks crow...*, the notion of a MIDI "note-on" event is avoided entirely. Instead, the FBO1 is controlled entirely through system exclusive code. This allows me to not only specify actual frequencies for the instrument but to also change all of the voice parameters rapidly in real time. Once again, the notion of "preset" becomes meaningless; all parameters of any preset can be (and are) modulated by the computer as fast as one wishes, and as such, the timbral possibilities of this inexpensive device become much more expansive. In *Simple Actions,* many of the little "critters" are designated to control real-time timbral parameters of the FBO1 (like LFO rate, shape and depth, algorithm, etc.). The DEP-5 and the Amiga local sound can also be controlled from the screen in similar ways.

Simple Actions (2) is, for me, a deliberate attempt at musical and cognitive co-evolution of myself and the machine. Like *17 Simple Melodies...*, it is, admittedly, a small beginning of a new form of musical and performance intelligence. But it is, I think, a beginning, and not an end. The nature of the piece obviates all attempts on my part at traditional notions of creativity, cleverness, drama, and "musicality," and forces me to rethink, in its performance, the very nature of the composer and of compositional, creative and perceptual intelligence.

Program notes by Larry Polansky.

About the pieces

The pieces on this concert are written and, except for *V'leem'shol*, performed live in the computer music language *HMSL (Hierarchical Music Specification Language)*, which was written and developed by myself, Phil Burk, and David Rosenboom at the Center for Contemporary Music at Mills College. HMSL is a language and programming environment for experimentation in musical artificial intelligence, composition, and real-time performance. It offers the user/programmer a large set of high-level musical and experimental tools, and can be used in a real-time performance context or in "non"-real-time compositional contexts.

The language itself is the product of about 6 years work on the part of the authors. It is still evolving, improving, and expanding, and many other composers and experimenters have contributed to its development. HMSL is currently available for the Apple Macintosh (Plus, SE, and II) and the Commodore Amiga computers, and is equivalent on the two machines.

Most of the pieces on tonight's concert were written after June 1987. *B'rey'sheet* existed earlier to that on the HMSL prototype machine at the CCM; the first, non-"tuned" version of *V'leem'shol* is also a few years old. Each piece explores and attempts to extend the possibilities of the HMSL environment, particularly the real-time environment, in different ways. Technically, each piece constitutes a kind of experiment in a different aspect of real-time intelligent software design. All the pieces emphasize my interest in the evolution of composition, perception and performance through intelligent software written by composers.

I have constructed this concert to involve many of the musicians and thinkers who have been important to my work in the Bay Area, my home for the last eight years, as a kind of temporary farewell, since my wife Jody Diamond and I are spending the next year in Asia. In addition, I think the works on this concert represent to an extent the conclusion of one phase of work with HMSL, and the beginning of a new, fertile and exciting phase of exploration, not only of the more powerful intelligent possibilities of the language, but of musical artificial intelligence itself.

I

Fl'e'13 **(B'rey'sheet: In the beginning...) (Cantillation Study #1) (for voice and interactive live computer) (for Jody Diamond)** *B'rey'sheet* is the first of a set of pieces called the *Cantillation Studies*, which are based on computer-aided morphological transformations of the 11th and 12th century

Masoretic cantillation melodies for the singing of Torah (Shabat morning tropes). The other two works in this set so far are:... אלה תולדות *(E'leh Tol'd'ot: These are the generations...) (Cantillation Study #3)* (for William Winant) for four marimbas and interactive computer; and לרשת אל *(V'leem'shol: And to rule...) (Cantillation Study #2)* (for Ann LaBerge and David Rosenboom) for five flutes. Each of these pieces is based on successive 17-verse sections of the Torah. The tropes are used as a basis for melodic transformation by the computer.

In *B'rey'sheet,* the tropes are sung, unadorned. The computer "listens" to the melodies and generates it's own events based on what it hears, and on where it is in the piece. There is a predefined trajectory of "computer attention," which specifies that at the beginning of the work the computer will be mostly ignoring the voice, but will gradually and continuously increase its attention until the end, when it tries to follow the voice closely. The computer deals with several musical parameters in its transformations: pitch, duration, intonation and many types of timbral and spectral deformations of four sine waves. Many of the spectral deformations used here are HMSL routines which are usually used for melodic transformation (retrograde, inversion, localized and ranged randomization, etc.), but here they are applied directly to the waveform itself in real-time, at audio rates. The degree and frequency of deformation of the sine waves that the computer will produce depends on which of the piece's 17 verses (or sections) it is currently in.

One interesting aspect of the computer algorithms used here is that the computers tune "on the fly," making use of an simple intonational trajectory which guides the tuning decisions. This trajectory begins in a complex 17-limit tuning space, and ends in a kind of simple 3-limit one (or Pythagorean tuning). The tuning algorithms used here fall into the category of what I have called "paratactical tuning" — all of the tuning is done in real-time, in response to the input, and no concept of scale, or gamut, is ever invoked. (For more on this, see my article in the *Computer Music Journal*). The pitches used are not chosen, but generated by the machine in real-time, similar to the way a chorus or string quartet would dynamically tune to itself over the course of a piece.

B'rey'sheet primarily uses Amiga local sound, with the exception of a single sine wave produced by the Yamaha FBO1 that follows the voice throughout. The sine wave and the voice are tuned to a simple septimal just scale (8/7's for major seconds, 12/7's for major sixths, etc.). This version of the piece was premiered at the International Computer Music Conference, Urbana, Illinois, August, 1987.

The piece was first written on the Amiga, using a custom built analog to digital converter and the Gentle Electric Pitch Rider to capture the melodies. This new portable version was ported to the Macintosh (and now, back to the Amiga!) by Robert Marsanyi and myself, and the idea was stimulated by Daniel Goode. This portable version allows much more performer liberty in the selection of sounds and the piece's overall character, and I am grateful to Robert Marsanyi for his several fine suggestions in this regard.

V

Simple Actions (2) (Distance Music VI) (for Daniel Kelley) for computer. *Simple Actions (2)* is a revision of *Simple Actions,* an earlier work (March 1987) done entirely for Amiga local sound. This new version integrates a new system exclusive library for the FBO1, and the ability to build one's own custom graphics screens in HMSL, and interact with them live. The piece itself is one of a set of works for performer/programmers called *Distance Musics,* recently published in *Perspectives of New Music.* This is a set of pieces for live interactive computer systems, in which each of the performers must write the code to perform the specific pieces.

Simple Actions is based to some extent on Minsky's "society of mind" concept — that complex intelligences are often created by the difficult-to-predict interaction of many simple intelligences ("agents") that share a certain informational ecology. Other ways to describe this notion might be as groups of cellular automata, or in a biological sense, as a large Petri dish of highly interdependent small organisms. The score for the piece defines a "simple action" as some musical process which can be described in less than one sentence or so. These actions have the further description that they must, as much as possible, share data with each other, so that when they alter that data through their own behavior, the data is altered for all other actions which might use it. In other words, all of the actions tend to alter the ecology of the system in some way, and the behavior of each is affected by the behavior of all.

The performance of this piece is rather unusual, and difficult. The performer can bring to life, or put to sleep, any of the actions, and in some cases, alter the range of behavior of the actions. More than that is difficult to control, since the system behavior often takes on a kind of life of its own. The actions themselves are extremely simple: beeps, glissandi, timbral changes, simple melodies, chords, simple modulations, etc. However, the number of processes possible at any time is very large — in this version seven FBO1 voices and four Amiga voices can do any or all of these things at any given time. In HMSL, these processes are simply set up as *jobs* — processes

lists, each containing all the melodies. In the third section, the machine plays back those three lists.

My main compositional interest in this work is in the second section, or the sorting procedure. For several years I have been working on a theory of what I call *morphological metrics* — ways in which distances between arbitrary morphologies (shapes, melodies) can be computed in ways that are perceptually meaningful (and cognitively interesting and evolutionary). This theory has been described elsewhere in some detail, but this piece is a very direct application, or experiment, in using these formulas and techniques in real-time. The metric function used here is something I call the Ordered Combinatorial Direction metric. It is fairly complex, but it might be simply described as a distance function between two melodies which is solely interested in the "up/down/sameness" of them, paying careful attention to the order in which this directionality occurs. I believe that this metric gives a good indication of what is sometimes called the "profile" of a morphology. It ignores things like "magnitude" of intervals in the melody, register (or tempo in the duration dimension), etc.

In the second section of the work this metric (OCD) is applied to the seventeen melodies to sort them in three ways. All three of the resultant lists represent "distances from some source melody," in this case the first melody that the performer plays. In other words, the first entry in each list is the melody "furthest" from the source (by some definable measure of "far" and "near"), and the last entry in each list is the source, or first melody itself (whose distance from itself is zero). For the first list, only the pitches of the melodies are used for the metric, in the second, only the durations, and in the third, equal weight is given to the pitches and the durations.

The first section captures the melodies by use of a pitch to MIDI converter. This information is interpreted by a class of intelligent objects in HMSL called *recorders*, which are more usually used in the system for conventional MIDI multi-track recording. The second section, which calculates all of the distances between shapes and sorts them according to these distances in the three lists, takes about 3 seconds. It makes use of a high-speed Batcher Sort that is part of HMSL, written by Phil Burk. The third section, where the computer actually makes sound, is extremely simple. The three sorted lists are played back polyphonically by the machine —in tonight's case on George Brooks synthesizer, for which he has created various sounds. Many other aspects of the piece are also customizable by the performer, including relative length, degrees and types of timbral change, and of course, the character of the melodic material itself. Deliberately, the piece does not allow me to make any compositional choices regarding the melodies used, or the sounds heard.

II

SIENSI **(V'leem'shol) (And to rule...) (Cantillation Study #2) (for Ann LaBerge and David Rosenboom) (for five flutes).** The version of *V'leem'shol* performed tonight is a revision of the older work, written using some simple algorithms that were being developed in the prototype of HMSL. *V'leem'shol* is a setting of the second set of 17 verses of the book *B'rey'sheet*. In this piece, Flute 5 (on tape) plays the cantillation melody throughout.

The form of the piece is similar to *B'rey'sheet* in that it goes from maximal transformation of the tropes to no transformation, but in this case the transformations are primarily *morphological*, or motivic, rather than statistical as in the case of the voice work. The computer algorithms produce a kind of motivic transformation canon, with each flute entering successively later into the piece , and going through a series of transformations in shorter and shorter times. Flute 1, the live flute, starts the work. Towards the end of the work, the individual flute lines converge on the trope itself, first in chorales, and then in unison, as the degree of transformation is reduced to nothing. This is technically a "mensuration canon," as are my other computer works the *Four Voice Canons*. The only anomaly in this formal procedure occurs when the phrase, "And it was evening, and it was morning, the (nth) day," appears in the text. The trope is always the same for this phrase; in these cases the flutes play a simple chorale.

This new version of *V'leem'shol* differs from the first version in that it uses an extended just intonation, similar to *B'rey'sheet*. A septimal just scale is used for the trope part (Flute 5), but the other flute parts range from complex tuning relationships to the trope at the beginning to simple ones at the end. This tuning was worked out by Ann LaBerge herself, and made use of some very useful scale and intonation programs, written for the Macintosh by tuning theorist John Chalmers, which allowed Ann to tune, learn, and experiment with complex just intonation in the work.

III

Cocks crow, dogs bark, this all men know. But even the wisest cannot tell, why cocks crow, dogs bark, when they do (for computers, computer controlled signal processor, performers, and text). *Cocks crow...* is a live improvisation for three performers and computer. One of the computers (mine, running HMSL) generates, in real-time, simple instructions for the performers to follow. These instructions consist of computer data sent to John's machine for him to process, and text instructions sent to Melody (on the screen) telling her what

to read, and (to some extent) how to read. Melody's instructions were written by her, she created the text especially for this piece. In addition, pitch information from Melody's voice is passed to me and John, greatly processed, and used in the total sonic environment in various ways. My computer also passes "presets" to John's machine, so that he may alter the more global parameters of his sound for each section of the piece. The length of each of the 17 sections is determined to some extent by the machine, and to some extent by one of the performer's input to the machine while the piece is being played.

The piece makes significant use of the ability to define custom *instruments* in HMSL. That is, raw compositional data in the language is never interpreted until the user specifies its "destination," and the way that data is interpreted is entirely up to the user. In this piece, four such instruments are defined for my computer. The first, called a *Chorder*, is an instrument which reads "melodies" (called *shapes* in HMSL) of five dimensions: duration, fundamental pitch of the chord, average loudness of the chord, harmonic complexity of the chord, and on/off time ratio for the duration of the chord. Harmonic complexity is here defined as the height of a pitch in the harmonic series of a fundamental.

The second instrument, and most important for this piece (again, for my computer only), is defined for the MIDI controlled signal processor, the Roland DEP-5. A system exclusive library has been written for this device in HMSL which allows the user to control all of the DEP-5's parameters in real-time, without using such MIDI notions as "preset" (or "program"), "note-on", etc. One can modulate, for example, the reverb time by the computer at very rapid rates, without changing the other device parameters. The DEP-5 instrument in this piece responds to "melodic" information for reverb time, pre-delay time, high frequency damping, filter Q and center frequency, and chorusing rate, depth, and feedback. As is the case with the Chorder instrument, one of the performers can graphically edit the shape of change for the complex "melodies" of this instrument in real-time during the piece. The third and fourth instruments used here, though used very sparsely, are similar system exclusive instruments for the FBO1. These will be described in more detail below, in *Simple Actions.* As in that piece, no MIDI "note-on" is ever sent to the device, and it is used rather like a computer controlled analog synthesizer.

The entire piece is defined as an HMSL *structure*, a class of musical objects in the language which "knows" how to play its component parts, or more accurately, has an internal intelligence for deciding when and how to play its component parts. This intelligence is user defined, and can be of great complexity, as can the components of the structure itself (for

example, it could contain other structures with equally complex intelligences). In this piece, the structure decides what instructions to give to the performers, how long to make the sections, what to do itself in each section, and other things pertaining to the instruments and sounds themselves.

One of the main ideas of the work was to implement a collaborative, highly complicated but essentially unpredictable piece, in a kind of stylistic homage to much of the early electronic music of composers like Gordon Mumma, John Cage and David Tudor. I was also interested in making use of the inherent intelligence possibilities of HMSL to create real-time notation, make its own sonic and temporal decisions, and invent its own form as it played itself.

This piece was premiered, in a much simpler form, at the Center for Contemporary Music at Mills College in June of 1987. The performers were myself, Amy Neuburg, and Jarrad Powell. This new version is a collaboration between Melody, John and myself.

IV

17 Simple Melodies of the Same Length (for Daniel Goode) for performer and computer (with technical assistance from Robert Marsanyi and Phil Burk) (December, 1987-January, 1988). One of my motivations in writing *17 Simple Melodies...* was to have a truly "portable" computer music piece. I was interested in writing something which would offer a live, interactive, and flexible computer piece for wind players, and one in which only their own equipment would be used. In tonight's concert, for example, we are using three sound producing machines: George's and Jeanne's synthesizers and Phil's Amiga (which acts as a sampler). These three performers are responsible for the choice of all sounds, as well as many additional large scale parameters. This piece can be, and has been, performed without the composer present, since all the equipment required is affordable, commercially available, and simple to use. All I have to do is send a disk to the performer!

The form of *17 Simple Melodies* ... is more or less "classic" artificial intelligence. Data is gathered by a kind of perceptron (seventeen melodies of seventeen notes each), the data is sorted (the intelligence), and finally, the data is re-played as a kind of simulation. The form of the piece very clearly follows these three sections. First, the performer may tune the computer, specify various timbral, length, and "probabilistic" aspects of the piece's "surface." The musician then improvises seventeen melodies, and the computer captures them. In the second section, the computer "sorts" these melodies into three different