SCORE—A Musician's Approach to Computer Music*

LELAND SMITH

Department of Music, Stanford University, Stanford, Calif. 94305

Computer music is being created at a number of American universities. While it is far easier to maintain precise control over computer music than music produced by analog synthesizers, the necessity to give instructions concerning every detail of the sound can lead to rigid, unmusical results. A program called SCORE which greatly facilitates a musician's communication with the computer has been devised.

The primary musical elements of pitch and rhythm are dealt with in musical terms in the SCORE program. Even the most complex rhythmic situations are quite simple to enter into SCORE. Perhaps the most powerful aspect of SCORE is its capacity to take advantage of the many redundancies found in most music. A unique feature is that whereby any elements of the music which will be repeated may be designated as "motives." No matter how long or complex, these motives may be reused in either their original or some altered form with a minimum of effort.

SCORE also allows for the easy introduction of chance elements into music. The musician can exercise any degree of control he wishes over these aspects. In this way it is simple to introduce the minute discrepancies between the parts which are natural to human performance.

FORTRAN subroutines may be added to SCORE which allow for practical microtonic notation. As a musician becomes acquainted with the nature of programming, it is possible to write compositional subroutines which can perform a multitude of functions.

More than sixty students have used the SCORE program. It has been found that musical knowledge is much more important than mathematical or computer knowledge for the production of convincing music from the computer. The SCORE program allows any musician to create computer music after a few hours of instruction.

Computer music systems are currently being developed and refined at numerous locations throughout the United States. Progress has been halting at best because of meager financial support. But for those who have worked in this field the staggeringly vast potential of the medium has inspired continued effort. At Stanford University a sound-generating program of remarkable flexibility has been developed by computer scientist David Poole and composer John Chowning. This program is a completely rewritten and expanded version of MUSIC IV which was developed by Max Mathews at the Bell Telephone Laboratories. The Stanford music program runs on a time-sharing PDP10 (Digital Equipment Corporation) computer which is the central component of the Stanford Artificial Intelligence Project. From the user's point of view many of the details of this system are similar to MUSIC IV and so will not be dealt with here.[1]

When this author began working with computer music in 1967 it immediately became apparent that it would be of great value to have a generalized program for producing complex musical "scores" which could be understood by a computer. Conveying musical ideas to a computer sound-generating program or analog electronic music system has usually been a rather cumbersome

* Presented April 29, 1971 at the 40th Convention of the Audio Engineering Society, Los Angeles.

[1] See James Tenney, "Sound Generation by Means of a Digital Computer," *J. Music Theory*, vol. 7 (1963).

affair. Very often music assembled with analog equipment is created after the fashion of an improvisation wherein the composer is never quite sure of just what sounds will come out next. It is a great tribute to the musicality of some composers that a few pieces manufactured in this way are as successful as they are. Despite the frequent dependence on fortunate accident, this form of electronic music often suffers from the same rigid quality of sound that is associated with computer-generated music.

It is easy to see why computer music tends to be rigid when you consider that rigid correctness is the computer's prime attribute. No matter what, the computer will make only the precise sounds it is told to. For the accomplished pianist who still must struggle to play an Etude of Chopin, the computer's accuracy may seem like a great blessing. But for those who have struggled to get the computer to produce a mere scale in a musical way this precision is surely a mixed blessing.

Fortunately the computer can balance its puritanical strictness with the refreshing permissiveness of random selection. Through judicious use of this random capability, subtle discrepancies in pitch, rhythm, amplitude, etc., which are inevitable in human performance, can become part of computer music.

For the computer to produce truly musical performance a myriad of detail must be taken into account. Consider the amount of information that must be given for the computer to play a single *C*. When does it begin, how long does it last, what it the precise frequency, what overtones are wanted, how loud, how much vibrato, etc., etc.? This list of parameters is easily doubled for more complex sounds. A one-minute three-voice composition of a conventional nature might well contain over 500 notes. While there is no way to avoid the necessity of specifying the most basic parameters of pitch and rhythm for each note, most of the other parameters can be dealt with in more general terms.

The parameter list for a three-note phrase to be played on the Stanford system by an "instrument" identified as Buzz would appear as follows:

```
PLAY;
BUZZ   0   .2  G  2000  F1  F3;
BUZZ  .2   .1  A  1500  F1  F3;
BUZZ  .3   .2  B  1000  F1  F3;
FINISH;
```

In this case the number immediately following the instrument name (parameter *P*1) represents the starting time for each note. The second column of numbers (*P*2) gives the duration of each note. The third column (*P*3) indicates the pitch; either the letter names of the musical scale or frequency numbers may be used here. The notes of the scale may be tuned in any way desired.

*P*4 is the amplitude; abbreviations for musical terms such as "forte" and "piano" could be used. However it has been found that a range of numbers from 0 to 2048 is easier to manage. *P*5 and *P*6 are functions which define the amplitude envelope and the waveform of the note, respectively. While the first two parameters are always reserved for the begin time and note duration, the others (up to 30 or more) may refer to whatever elements of sound a particular instrument might control.

In MUSIC V,[2] which is currently operational on many university computers, the same three notes might appear as:

```
NOT  0  1  .2  .2555  20.0212  2000  1  3;
NOT .2  1  .1  .511   22.484   1500  1  3;
NOT .3  1  .2  .2555  25.2434  1000  1  3;
TER .5;
```

MUSIC V is divided into three sections with Pass I creating data for Pass II where it is further processed and then sent on to Pass III. Pass III deals only with the raw numbers which are needed for the computation of the final sound samples. In the example, the fourth and fifth columns of numbers include the conversions for envelope duration and pitch which are necessary for Pass III.

Computer music systems are basically just a set of programs which cause the many numbers representing a changing composite sound wave to be stored on some memory device such as magnetic tape or disk. The sound is produced by reading these numbers at a fixed rate through a digital-to-analog converter where each number is converted to a specific voltage level. Computers operate of course in a completely numerical fashion, hence at some point all communication with the computer must be converted to numbers.

Pass I of MUSIC V does a minimal translation of words into numbers (NOT = note = the number 1, etc.). By attaching FORTRAN subroutines to Pass I, various elements of the music can become automated or an entire composition may be computer generated. Pass II sorts the note cards (i.e., the output of Pass I) into the proper time order, and further subroutines are usually added which convert various numbers into the special form required by Pass III.

A program called SCORE has been devised by this author which prepares parameter lists for the Stanford sound-generating system. A slightly revised version called SCORV is used as a replacement for Pass I and Pass II in MUSIC V. An important advantage of SCORV is that any desired conversions of frequencies or durations to the proper MUSIC V numbers are done automatically. While it may take a musician some weeks' study to learn how to use a computer music system from scratch, that is, learning "computerese" for setting up instrument definitions, tone qualities, envelopes, etc., by using SCORE and a group of preprogrammed instruments, a neophyte is able to make music after a couple of hours' instruction.

Of course the musician must know in advance the characteristics of the instruments available so that he may prepare a manuscript of some sort before he creates his computer score. The manuscript can be completely conventional or it may be of some graphic form. The only requirement with graphic scores is that there must be specific points of reference in regard to real time (i.e., seconds) and pitch.

SCORE treats the data for each instrument parameter by parameter, that is, all the input for *P*2 in the first instrument are read, then all the data for *P*3 are read,

[2] A detailed description of MUSIC V, as well as an extremely valuable discussion of many technical aspects of computer sound, is found in Max V. Mathews, *The Technology of Computer Music* (M.I.T. Press, Cambridge, Mass., 1969).

and so on. After the processed data are all stored, the actual generation of the output (the "note" cards for Pass III in the case of MUSIC V) takes place. SCORV uses the parameter list in the same way as SCORE. Instruments are identified by names, and the numbering of parameters starts with the note begin times.

In the first processing, many of the time-saving features of SCORE take effect. The most important of these have to do with the various aspects of repetition which abound in most music and the translation of musical terminology into computer-digestable data. During the output phase, any desired interaction between parameters takes place and their final values are established. The data for *P*1, the note begin times, are generated automatically at this time and so requires little attention.

When dealing with conventional musical notation, SCORE first acts as an interpreter of musical rhythm into the real-time values which are necessary for the computer. If the appropriate codeword (RHYTHM or just RH) is typed after *P*2, the program understands all the note durations to be in terms of musical time. Each number entered will be the denominator of any fractional part of a whole note. Thus 4 equals a quarter note, 8 an eighth note, 12 a triplet (i.e., a third part of a quarter), 27 a 27th note, etc. A minus number is used for rests. As in ordinary notation, any number of dots may be added to any note value. For composite or tied notes, two or more numbers may be typed before the slash which will indicate the end of each note. Some examples follow.

```
4 16 /-8. /2  8. /16  / 1..  /
```

is equivalent to

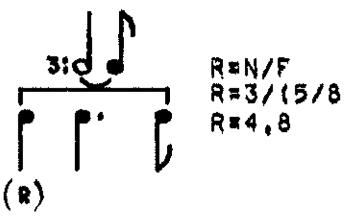
A slash with no number repeats the last seen value. If a single rhythmic value repeats several times an *X* is entered before the number of repeats. /4. X 5/ means that five dotted quarter notes in a row will be heard. Long sequences of varied rhythms may be repeated as often as desired by using the code word REP. Thus the pattern

repeated eight times would be typed as /4 16/32/ /8 *X* 4/REP 7 8/. That is, the last seven items should repeat eight times. In complex situations the true rhythmic value of nondupal units is found by simply multiplying the two elements of the ratio involved. Thus

is five 40th notes (5 × 8).

In the following example the half note tied to an eighth will be called the "base rhythmic value." If the base rhythmic value is a composite, i.e., requires more than one type of rhythmic unit for its representation, the formula $R = N/F$ may be used. If N is the number of equal rhythms to be fitted into the base value and F is the fraction form of the base value, then R is the true rhythmic value of each of the equal units.

In the case of



These rhythms would be entered as /4.8 /4.8. /9.6 /. (SCORE distinguishes between decimal points and rhythmic dots.)

The real-time value of a quarter note will depend on how the tempo factor is set. The tempo may be set independently for each voice or for all parts at once. There is total flexibility in this regard so that retards and accelerations may be worked out in great detail either at the teletype or, by means of a supplementary program called RHYTHM, in "live" performance.[3] In RHYTHM the computer is told, in effect, to "listen" to a pair of telegraph keys. A sequence of up to 500 notes may be tapped at any one time. Since it is extremely easy to edit the results in every way, the composer can improvise freely and then fit these data accurately to other parts which have been precisely specified by SCORE.

Most often the rhythms of the various parts of a musical work are intended to be coordinated. However, a certain amount of flexibility is highly desirable. There is provision in SCORE for coordination, on any level of detail, of the rhythms of individually prepared parts. This is done by indicating to SCORE the points at which coordination is to take place and how many notes are to be fitted between each set of points. The tapped rhythms may be as free as desired and even in slow motion. If, for example, six notes of irregular duration had been tapped and the first five were to fit into the time space of nine eighth notes with the sixth note to be an eighth note arriving precisely on the tenth eighth note of the background time, the following pairs of data would be entered:

```
/ 1 8/5 / 8/1 / =
  (1.)     (2.)
```

[3] The RHYTHM program has not yet been activated to run in connection with MUSIC V. In order to run an interactive real-time program such as this the continuous attention of the computer is required. There are not a great many of the large computer centers that offer this possibility; however, it is probably feasible on most of them. The PDP10 at Stanford is a time-sharing system which can be programmed to give apparent continuous service while simultaneously serving several other users.

1) The relative durations of the first five notes will be maintained as they were tapped, but their real-time value will add up to the value given to one whole note plus an eighth by the tempo factor or the "conduct" feature of the RHYTHM program. 2) The time of an eighth note will be given to the last tapped duration.

The "conduct" feature may be used to direct the computer as if it were an orchestra. In this case the telegraph keys are not used for any details of the rhythm but rather only the main beats are tapped, after the manner of a conductor. The rhythmic details for all parts are entered in SCORE by means of the teletype or in RHYTHM with the telegraph keys. No matter how complex any of the parts may be, they will all follow whatever tempo fluctuations are desired by the "conductor."

Provision is made for editing the conducting input in several ways. The overall tempo factor may be changed, the durations of individual beats may be altered, beats may be added or deleted. Although the unit value for each beat is originally set as a quarter note, any beat may be reset to any combination of rhythmic units. Thus any irregular meters may be conducted.

Tempo changes are usually gradual but sometimes an abrupt change is desired. Of course the telegraph keys can indicate nothing more than the duration of each beat. From the duration of a beat, its predecessor, and its follower, the tempo at the beginning and at the end of each beat is computed. Naturally the average tempo of the beat cannot be changed unless either the duration or the rhythmic unit is changed. The RHYTHM program automatically sets the first and last tempo of each beat in such a way as to ensure the smoothest possible changes from one beat to another. With the editing provision it is possible to override these automatic features and adjust the tempo at each end of the beat, thereby causing abrupt tempo changes.

If the composer's manuscript is a graphic one, very similar procedures are followed since seconds and quarter notes are basically interchangeable. The codeword NUMBERS allows SCORE to understand the data as real-time values instead of the relative values of musical rhythm. However, the possibilities of repeating groups of values and altering the tempo of the final outcome still may be applied.

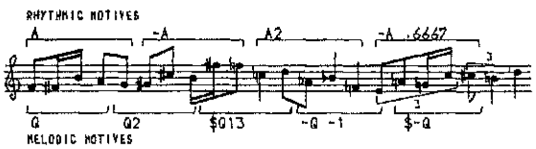
SCORE allows one to treat pitch either in terms of note letters or frequency numbers. While the tuning of the tempered scale is given to the note letters, it is rather simple to retune the scale in any way desired. The octave ranges are numbered in the standard manner from the lowest *C* on the piano keyboard (*C*1), and these numbers need only appear when there are changes in octave ranges. Sharps and flats are indicated by the letters *S* and *F*. (No particular need has been found for double sharps or flats.) The same repeat features are available here as were found in the rhythmic realm. The letter *R* is available for rests since, in most cases, the total number of pitch entries must match the number of rhythmic entries.

The preceding pitches would be encoded as follows:

*/CS4/R/D/AF/G/C5/*REP 3/REP 9,3/.

The first REP 3 simply repeats the previous three notes. REP 9,3 causes the last nine items to be played a total of three times. These features apply equally well if frequency numbers are used instead of the note letters.

If any motivic portion of a string of rhythms, notes, or numbers is to be used several times, it is possible to associate the group of items involved with some identifying symbol. The limits of the motive are established with parentheses. Thereafter, when the motive is to be used, the identifier is preceded by the sign @. Motives may be played in retrograde by including the minus sign just before the identifier, e.g., @ − *Q*. When dealing with the names of notes, a transposition number may be added. @*Q*12 would play motive *Q* 12 half steps (an octave) higher. The dollar sign is used to create exact intervalic inversion. If the motivic material is rhythmic or just a string of numbers, any number following the identifier acts as a multiplier. Thus all augmentation or diminution of rhythmic patterns is made quite simple.




```
P2 RHY/ A (8/16//8//) @-A / @A2 / @-A .6667 / 8 / 4 *;
P3 NOTES/Q (F4/FS/B/A/) @Q2 / @$Q13 / @-Q -1/ @$-Q / B4/ D5*;
```

Once the motives are defined, they may be freely used by any other instrument. It is readily apparent that this feature greatly facilitates the coding of many musical situations.

A brief passage from page 44 of Stockhausen's *Zeitmasse* (Universal Edition, c/o Theo. Presser, Bryn Mawr, Pa.) will be used as an illustration of SCORE's ability to handle complex rhythmic notation (Figs. 1 and 2).

In this example the details of articulation and dynamic changes within notes are ignored. These parameters present no special problems and would require only two more lines of data for each instrument. An arbitrary scale of dynamics is established as follows:

ppp = 100	*mf* = 750
pp = 180	*f* = 1100
p = 300	*ff* = 1500
mp = 500	*fff* = 2000.

It should be noted that in Music V the amplitude limit for all the instruments combined at any moment is 2048. If this is exceeded, SCORE has provision for the scaling of all voices to meet this requirement before the final computation of the sound samples.

Ordinarily the tempo would be set on a single line for all parts at once, but this piece requires individual treatment for each voice. The real-time values of the rhythms will be established by the particular tempo in force in each instrument at any given time. The tempo indications come between the word TEMPO and the asterisk, which is used as a general closing indication. For the oboe the tempos are 2.25 quarters at 56, one quarter at

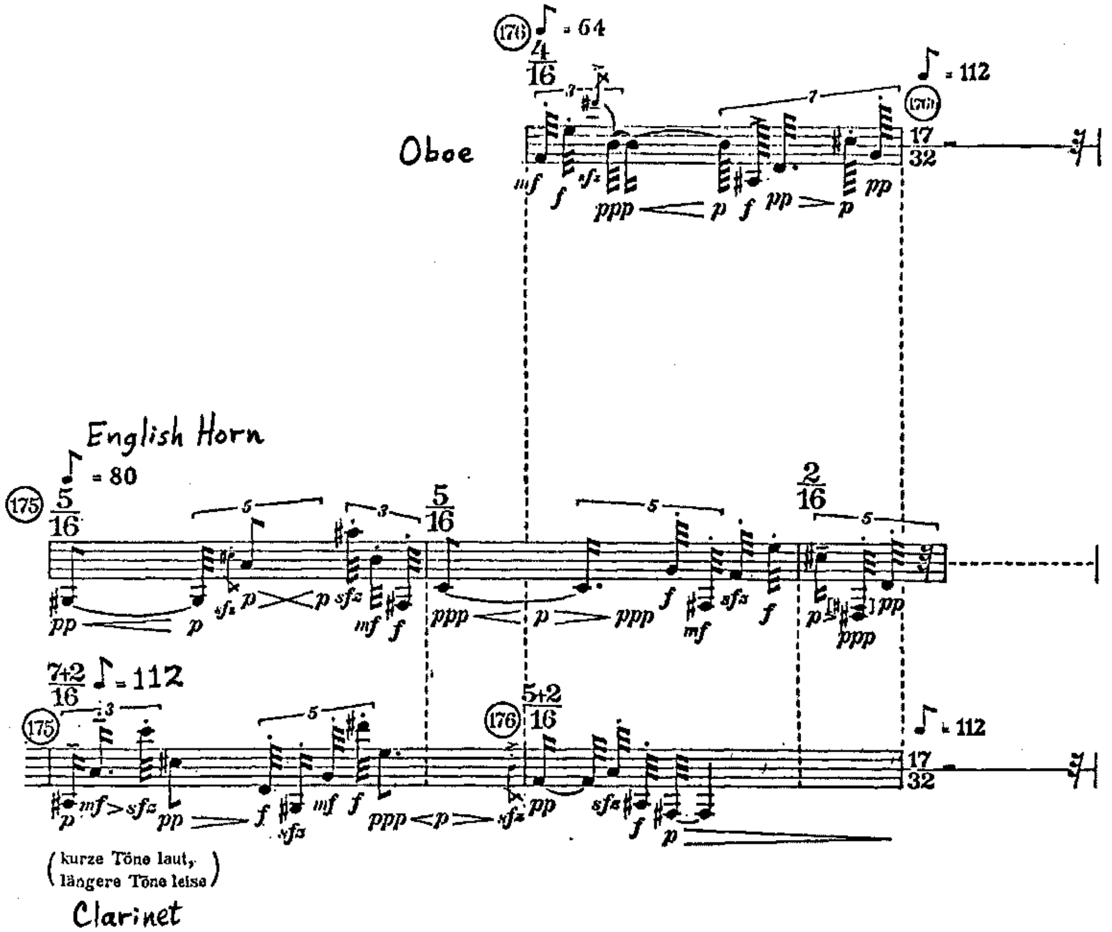


Fig. 1. "Zeitmasse" (page 44), Karlheinz Stockhausen.

32, and the remainder of the excerpt at 56. The English horn (EGHN) and clarinet have constant tempos at quarter 40 and quarter 56, respectively.

The grace notes are taken to have one third the value of the shortest nearby note.

The FINE as the final item in each $P3$ line is an indication that the instrument will stop playing at that point. The word END appears as the last item for each instrument. This lets SCORE know that all data for that instrument has been seen.

```
ZEITMASSE -- PAGE 44 (number 175)
OBOE /P2 RHY/TEMPO/2.25  56/1  32/56*)
*2 16 /48 /72 /144 /48 16 56 /56 /28, /56 //-2 32*;
P3 NOTES;
R  /E (F4 /E5 /DS6 /) J.(B4 /AS3 /D4 /) CS5 /G4 /R /FINE*;
P4 NUMBERS;
0  /750 /1100 /1400 /100 /1100 /180 /300 /180 /0*;
END;

EGHN /P2 RHY/TEMPO/40*;
8 60 /120 /10 /48 / / /8 20, /40 / /32 / /20 /40 / /-40*;
P3 NOTES;
GS3 / 0-J 11 /B4 /FS3 /C4 /G / 0Z -11 / CS5/ DS3 / D4 / R / FINE*;
P4 NUM;
180/1100/300/1100/750/1100/100/1100/750/1400/1100/300/100/100/0*;
END;

CLAR /P2 RHY/TEMPO/56*;
24 /24, /40 /8 /40 /20 /40 //8 32, /64 /16, /32 //32 4 /-2 32*;
P3 NOTES;
AS3 /A4 / 0-E -3 / GS3 / G4 / DS6 / E5 /B4 /F /A /AS3 /FS /R /FINE*;
P4 NUM;
300/750/1100/180/1100/1400/750/1100/100/1100/180/1400/1100/300/0*;
END;
RUN;
```

Fig. 2. Computer input for "Zeitmasse" excerpt.

Fig. 3 is a print of the output SCORE would produce for the Stanford sound system. Each line represents one note. All the rhythmic elements have been converted to real time, the first column of numbers being the begin times and the second being the note durations in seconds. The amplitudes, found after the note letters, have been scaled by a factor of 0.72 so as not to exceed a total of 2048. The notes that appear are internally converted to the frequencies of the twelve tones from middle C up to B. $C/2$ would be one octave lower, $C*4$ would be two octaves higher, etc.

The listing from SCORV would be quite similar. However, the numbers which would be sent to Pass III of MUSIC V for the parameters of the first note of the English horn would be as follows:

$$7\ 1\ 0\ 2\ .85\ 10.611\ 129.6\ .06$$

where 7 = wordcount, 1 = an instrument, 0 = action time, 2 = Inst. #2, .85 = note duration, 10.611 = increments for the note $G\#3$, 129.6 = amplitude, .06 = increments for envelope.

When changes are desired within the duration of a single note, the instrument definition must include this capability. Thus swells, glissandos, stereo motion, etc., may be achieved by a single set of note parameters.

```
 <  ZEITM,DAT
PLAY;
<OBOE      Ø,ØØØ     2,411          REST  <  1
EGHN       Ø,ØØØ     Ø,85Ø GS/2     129,6ØØ;< EGHN     1 >
CLAR       Ø,ØØØ     Ø,179 AS/2     216,ØØØ;< CLAR     1 >

CLAR       Ø,179     Ø,268 A        54Ø,ØØØ;< CLAR     2 >

CLAR       Ø,447     Ø,Ø89 C*4      792,ØØØ;< CLAR     3 >

CLAR       Ø,536     Ø,536 CS*2     129,6ØØ;< CLAR     4 >

EGHN       Ø,85Ø     Ø,Ø5Ø CS*2     792,ØØØ;< EGHN     2 >
EGHN       Ø,9ØØ     Ø,6ØØ A        216,ØØØ;< EGHN     3 >

CLAR       1,Ø72     Ø,1Ø7 D        792,ØØØ;< CLAR     5 >

CLAR       1,179     Ø,214 GS/2    1ØØ8,ØØØ;< CLAR     6 >

CLAR       1,393     Ø,1Ø7 G        54Ø,ØØØ;< CLAR     7 >

EGHN       1,5ØØ     Ø,125 AS*2     792,ØØØ;< EGHN     4 >
CLAR       1,5ØØ     Ø,1Ø7 DS*4     792,ØØØ;< CLAR     8 >

CLAR       1,6Ø7     Ø,737 E*2       72,ØØØ;< CLAR     9 >
EGHN       1,625     Ø,125 B        54Ø,ØØØ;< EGHN     5 >

EGHN       1,75Ø     Ø,125 FS/2     792,ØØØ;< EGHN     6 >

EGHN       1,875     1,2ØØ C         72,ØØØ;< EGHN     7 >

CLAR       2,344     Ø,Ø67 B        792,ØØØ;< CLAR    1Ø >

OBOE       2,411     Ø,156 F        54Ø,ØØØ;< OBOE     2 >
CLAR       2,411     Ø,4Ø2 F        129,6ØØ;< CLAR    11 >

OBOE       2,567     Ø,1Ø4 E*2      792,ØØØ;< OBOE     3 >

OBOE       2,671     Ø,Ø52 DS*4    1ØØ8,ØØØ;< OBOE     4 >

OBOE       2,723     Ø,759 B         72,ØØØ;< OBOE     5 >

CLAR       2,813     Ø,134 A       1ØØ8,ØØØ;< CLAR    12 >

CLAR       2,947     Ø,134 AS/2     792,ØØØ;< CLAR    13 >

EGHN       3,Ø75     Ø,15Ø G        792,ØØØ;< EGHN     8 >
CLAR       3,Ø81     1,2Ø5 FS/2     216,ØØØ;< CLAR    14 >

EGHN       3,225     Ø,15Ø FS/2     54Ø,ØØØ;< EGHN     9 >

EGHN       3,375     Ø,188 F       1ØØ8,ØØØ;< EGHN    1Ø >

OBOE       3,482     Ø,134 AS/2     792,ØØØ;< OBOE     6 >

EGHN       3,563     Ø,188 E*2      792,ØØØ;< EGHN    11 >

OBOE       3,616     Ø,4Ø2 D        129,6ØØ;< OBOE     7 >

EGHN       3,751     Ø,3ØØ CS*2     216,ØØØ;< EGHN    12 >

OBOE       4,Ø18     Ø,134 CS*2     216,ØØØ;< OBOE     8 >
EGHN       4,Ø51     Ø,15Ø DS/2      72,ØØØ;< EGHN    13 >

OBOE       4,152     Ø,134 G        129,6ØØ;< OBOE     9 >
EGHN       4,2Ø1     Ø,15Ø D        129,6ØØ;< EGHN    14 >

<OBOE       4,286    2,277          REST  < 1Ø
<CLAR       4,286    2,277          REST  < 15

<EGHN       4,351    Ø,15Ø          REST  < 15

FINISH; < ZEITM.DAT

AMPL, FACTOR=Ø,72, MAX.AMP.=2Ø16, AT TIME   3,482
TOTAL DURS:  OBOE =  6,563  EGHN =  4,5Ø1  CLAR =  6,563
```

Fig. 3. SCORE output for "Zeitmasse" excerpt.

However, it is frequently desirable to have gradual shifts in parameters from note to note. With the MOVE feature of SCORE, straight-line changes for any parameter become automated. The beginning and end points of these changes are set over a duration which is conceived in terms of seconds or quarter notes of time.

If one wanted a change in amplitude (*P*4) from 200 to 1200 during five seconds, it would be entered as

P4 MOVE/5 200 1200/.

If each note were one second long, *P*4 would contain successively 200, 400, 600, 800, 1000 and, at the beginning of the sixth second, 1200.

By having several segments in the slope of change it is easy to simulate curves (Fig. 4).

By using pairs of numbers between the slashes, moving ranges of random selection are created. Fig. 5 shows how changing random selection of stereo location might be notated. The first note will be entirely in channel *A* while any note sounding at the 20th second will be precisely in the middle. At all other times the notes will be randomly positioned according to the limits shown in the graph. This approach may be used with any aspect of the sound.

The random selection feature may also be used apart from the MOVE feature. Several fixed ranges of randomness may be set for any parameter. In the following example for *P*5, 50% of the time a number between 20 and 100 will be chosen, 10% of the time between 0 and .7, 10% of the time the number 200, 25% of the time between 107 and 115, and 5% of the time the number 1000.

P5 .5 20,100 .1 0, .7 .1 200,200 .25 107,115 .05 1000, 1000;

If random selection of tempered scale notes is desired, note names may be used instead of numbers.

```
P3 .5 C2,C3 .2 FS4,FS4 .3 D6,F6;
```


```
P4 MOVE/3 1ØØ 2ØØ/2 2ØØ 3ØØ/1 3ØØ 4ØØ/1 4ØØ 6ØØ/1 6ØØ 9ØØ/
```

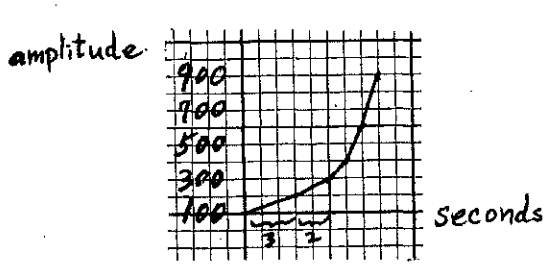
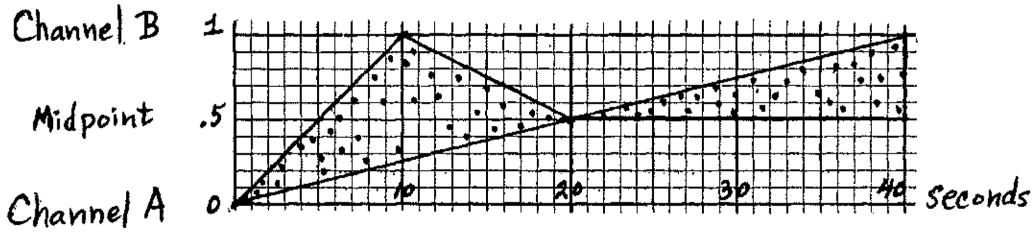


Fig. 4. Simulated curve using MOVE Feature.



P1Ø MOVE/1Ø Ø,Ø 1,,25/1Ø 1,,25 ,5,,5/2Ø ,5,,5 1,,5/

Fig. 5. Random selection of stereo position using MOVE.

A random tempo factor may be set to simulate "human error" and, in the most aleatoric works, even the rests can be created at random.

One of the most useful aspects of SCORE is that all the various ways of dealing with the parameters of sound can be rapidly interchanged. The changes may have begin times in terms of seconds or note numbers. In a chance composition one could begin to use specific amplitudes, or timbres, etc., starting at the 31st second, or perhaps one might want such changes to occur on the 59th randomly chosen note duration.

The parameter numbers in computer music are usually shared by all instruments. In SCORE the values of parameters will remain in force until specifically changed. Because of this, it is possible to set up dummy parameters for the purpose of passing information from one note to another, even though several instruments might use many of the same parameter numbers.

Suppose one wished to have instruments play glissandos with randomly chosen notes as beginning and end points, but without there being any skips between notes. The beginning pitch might be in *P*3 and the ending in *P*4, and *P*20, if it is not used in any instrument, could be used to store the value of *P*4 at any given time. After the first note, *P*3 (start of gliss) will always pick up the value found in *P*20. 100% of the time *P*4 will choose a note (end of gliss) from the range *C*2–*C*6 and *P*20 will store *P*4's choice so that it will not be wiped out as other instruments use *P*4. This would be encoded as

*P*3 *P*20/*P*4 1 *C*2,*C*6/*P*20 *P*4 /

Perhaps SCORE cannot do everything the musician desires, but by adding a user-written FORTRAN subroutine it can approach this ideal. To use this feature it is necessary that the musician become acquainted with the less complex elements of FORTRAN programming. Since all the input–output is handled by the main program, it is input–output is handled by the main program, it is necessary only to deal with encoding the logic of the special compositional needs. The subroutine may be used by any or all instruments. Parameter information may be left behind in the subroutine for future use, and any parameters for the current note may be revised. For example, it would be quite simple to have middle *C*, among randomly chosen notes of several instruments, always produce a complete *C* major chord. Each instrument, as it entered the subroutine, would check to see if another had chosen *C*. If so, its own random choice would be replaced by some specified element of the *C* chord.

Another use for a subroutine might be to create an easily managed quarter tone system. In the following example *P*3 will have the standard tunings and *P*4 will contain the quarter tone data: 0 = no change, −1 = quarter tone lower, 1 = quarter tone higher. The input to produce a quarter tone scale "chromatically" up from *C* to *D* and back again could be

*P*3 NOTES/C4//CS//D//DF//C/

*P*4 NUMBERS/0/1/0/1/0/-1/0/-1/0/

The essential part of the subroutine would appear as follows:

1) $X = 30.868*2**(P(3)/12.)$
2) $P(3) = X*2**(P(4)/24.)$.
RETURN

1) At this stage *P*3 will contain a note number. (1 would be the lowest *C* on the piano keyboard, etc.) This formula uses *P*3 as the number of half steps above the lowest *B* (30.868 Hz).

2) Here the final pitch is computed and put into *P*3. The formula raises or lowers by one quarter tone or leaves unchanged the value of *X* depending on the number in *P*4.

It is rare that even this much mathematics is required when working with computer music. In the case of compositional subroutines it is usually a case of fundamental logic. If *A* is true then go to *B*, etc.

In effect, the main program of SCORE is simply a series of subroutines which deal with most musical requirements. The user-added subroutine can extend SCORE's capability as far as the musician's programming abilities allow. SCORE is written in conventional FORTRAN IV. The "heart" of the program is a scanning routine which reads most lines a single character at a time. By this means most of the objectionable restrictions concerning FORTRAN input are avoided. The scanner decides whether a character is a number, letter, or punctuation mark and translates this information according to the few key words it looks for. The main program has been conceived in what might be called a modular sense. Thus as new features are desired (and this seems to happen with every useful computer program), their implementation requires relatively little programming effort.

The use of SCORE will bring computer music easily within the grasp of any musician who can gain access to a computer capable of running an extensive sound-generating program. The output of SCORE may be adapted to any system which treats the parameters of sound individually and, incidentally, may be used as input for a wide variety of computer programs for musical research and education.

It is incorrect to think of computer music as being for the avant-garde composer alone. At this time the computer has become merely(!) another musical instrument, but an instrument with capabilities and potential far beyond those of any other instrument. It is not a "performer's" instrument, but rather a "musician's" instrument. Theoretically, any performance, clearly conceived in the mind, can be realized on this instrument. Doubtlessly some imaginative musicians will dream up ideas which will exceed the limits of present systems, but the continuing advance of computer technology will quickly minimize such gaps, if they come to exist.

THE AUTHOR

Leland Smith was born in Oakland, Calif., in 1925. He studied musical composition with Darius Milhaud and Roger Sessions, as well as in the class of Olivier Messiaen at the Conservatory of Music, Paris, France.

Prof. Smith taught at Mills College, University of Chicago, for six years. Since 1958 he has been with Stanford University, Stanford, Calif., where he is now Professor of Music. He is teaching composition and computer music. His compositions have been played by the Orchestra of America at Carnegie Hall, the San Francisco Symphony, and at concerts in Paris, London, Athens, Taipei, and many U.S. locations.