CHAPTER 3

LITERAL SELECTION

A literal strategy for making decisions is a strategy which provides exactly one acceptable option for each decision. Such a strategy can be thought of as a compositional 'recipe' which takes as its 'ingredients' some collection of musical material. This material is designated as the prime material or original material. It may include tangible themes or more abstract patterns such as motivic cells, rhythmic figures, chordal progressions, and so on. The strategy subjects this material to one or more wholly determinate procedures in order to expand it into a complete work.

When literal derivations are clearly heard, they impart a sense of unity or coherence which many composers consider aesthetic. Musical compositions which are completely determined using a literal strategy are are often called totally organized. The strategy used to produce such works completely describes the music which results; description of specific notes reduces to elementary calculation.

3.1 TERMINOLOGY

Literal procedures for expanding prime material into a composition divide into two categories:

1. Transformations (also called operations) serve to create variants of already-defined material. In the most general case, transformations may be capable of action not only upon the prime material, but also on their own results.

2. Productions combine original and varied material into increasingly large aggregates and ultimately into a finished composition.

3.1.1 Transformations

Figure 3-1 illustrates six procedures for varying melodic

motives along with six analogous procedures for varying rhythmic patterns. These procedures fall conveniently into three generic groups:

1. Translations maintain the original shape, but shift the frame of reference;

2. Reflections restate a pattern upside-down or backwards; while

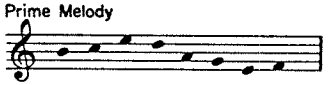
3. Permutations shuffle the elements in a pattern without changing its original overall content.

Transformations may be applied singly or in combination. Notice that the three groups are by no means mutually exclusive. Melodic inversion may be loosely regarded as a translation, since the relationship between shapes is easily discernable by ear; similarly, melodic retrograde qualifies as a permutation because the overall collection of pitches is not affected.

Figure 3-1: Musical transformations - Each melody and rhythm illustrates the effect of a transformation upon the "prime" material given at the top of the Figure. Melodic and rhythmic transformations are paired

Prime Material

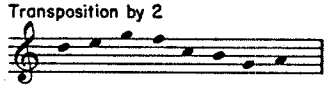
Prime Melody



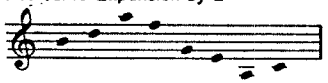
Prime Rhythm

Translations

Transposition by 2



Intervallic Expansion by 2

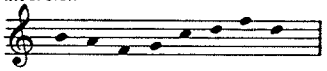


Metric Displacement by 2

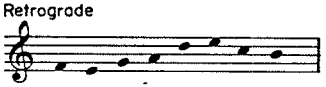
Augmentation by 3:2

Reflections

Inversion



Retrograde

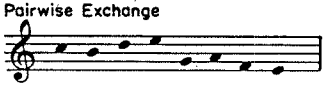


Inversion (with diminution by 4)

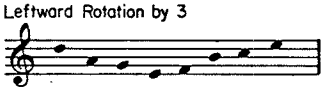
Retrograde

Permutations

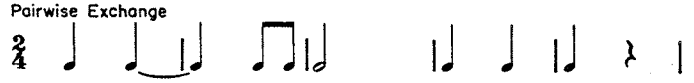
Pairwise Exchange



Leftward Rotation by 3



Pairwise Exchange



Leftward Rotation by 2

according to procedural similarities.

The simplest translation -- transposition -- along with the simplest reflections -- melodic inversion, simultaneous melodic and rhythmic retrograde -- are classically associated with the historic canon. The word "canon" is Latin for "law;" strict canons are notated as a single line of music combined with a riddle (i.e., law) hinting at how to derive the counterpoints. Celebrated canons include those of the Musical Offering (1747) by Johann Sebastian Bach (1685-1750). It must be stressed that while canonic recipes as expressed in the riddle are literal, these historic canons also obeyed the stylistic norms of their times -- the act of composing a canon is much more complex than the act of recreating it from the riddle.

Change ringing is a historic example of permutations as contrasted to the translations and reflections appearing in canons. Change ringing is based upon pairwise exchanges. It arose in English churches prior to the fifteenth century (Snowdon, 1940).

Musical theorists -- especially Milton Babbitt -- have drawn upon the mathematical theory of groups to isolate those transformations which they consider most useful. For one example, it can be shown mathematically that unless the size of an intervallic expansion is relatively prime to the number of

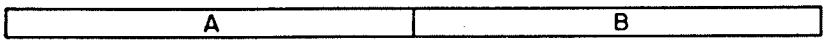
scale degrees, one risks "mapping" distinct degrees in a prime melody into identical degrees in a varied melody. Specifically, expanding semitones into whole tones will map chromatic material (e.g., a twelve-tone row) into a whole-tone scale because the interval of expansion, 2, divides evenly into the number of degrees, 12. By contrast, expanding semitones into perfect fourths will produce a new chromatic melody because here the interval of expansion, 5 (there are five semitones in a perfect fourth) does not divide evenly into 12. For another instance, if a pattern has N elements then the transformation "rotate by M" will produce N distinct variants of this pattern if and only if M is relatively prime to N.

3.1.2 Productions

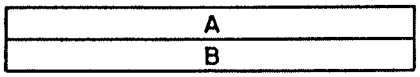
Figure 3-2: Some elementary productions.

Figure 3-2 illustrates some elementary productions. The two most fundamental productions for combining original or varied material into a larger aggregates are splicing and layering. Splicing (also appending, concatenating, or sequencing) begins one pattern after another has run to completion, that is,

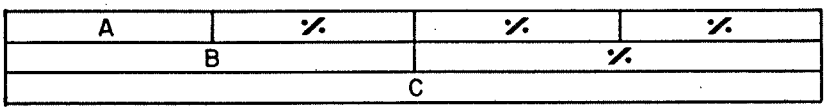
Splicing



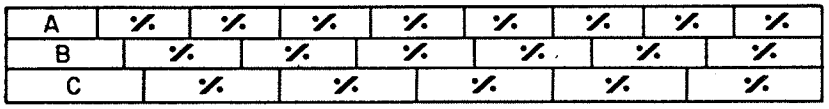
Layering



Cycling



Phasing



sequentially. Layering (also superimposing, mixing, or merging) begins two patterns at the same time, that is, simultaneously. Examples of these two productions include the two basic techniques of thematic development: variation and fugal imitation. The two fundamental productions exist as extremes of a continuum, with staggered layerings as an intermediate production. In staggered layerings, each new pattern enters only after another has played for a certain time. Fugal stretti are familiar examples of staggered layerings.

Cyclic productions take two or more patterns of unequal length, start them off together, and repeat them over and over until they coincide at least once more. The aggregate pattern is called a resultant, the instants of coincidence are called nodes, while the music between two consecutive nodes constitutes one cycle (or period). In the simplest cyclic productions, shorter patterns divide equally into each longer pattern. One statement of the longest pattern gives one cycle, and nodes are usually marked off by concerted attacks.

In the Western musical tradition, the earliest documented use of cycling to produce compositional material is isorhythm. Isorhythm first appeared around the time of Philippe de Vitry (c1350). It cycles a melodic pattern called the color against a rhythmic pattern called the talea. The talea typically divides evenly into the color so that each new color coincides with a new

talea. The full cycle is a melody called the tenor (from the Latin, "to hold") around which medieval composers wove free counterpoint in shorter note values. Cyclic designs are also employed in Indonesian music (cf. McPhee, 1966).

Phasings are cyclic productions in which none of the patterns divide evenly into the others. Accents which are concerted in one alignment become staggered in the next. Each pattern has its own intrinsic tendencies, but as they come together in different alignments, they interfere (cf. Schillinger, 1941) "constructively" and "destructively" to create a resultant with new tendencies of its own. Phasings of melody against rhythm produce a syncopated effect favored in Indian music and in jazz.

3.2 IMPLEMENTATION

Literal procedures are by far the easiest compositional processes to implement as computer programs. The basic approach in most instances is first to acquire an untransformed "sample" from the prime material and then to manipulate this sample using arithmetic formulae and/or array references.

3.2.1 Sampling a Row

The library subroutine SEQUEN is the first of many self-contained routines included within this book for the purpose of selecting one option from a repertory supplied by the user. It corresponds to the SEQUENCE feature of Gottfried Michael Koenig's PROJECT2 program, and implements the most basic of literal procedures: sampling elements in order from a row. Calls to SEQUEN require five arguments:

1. RESULT - SEQUEN selects a new element from a row and returns an associated value in this location (line 11). RESULT may be either integer or real in the calling program.

2. VALUE - Repertory of values (options), arranged in order of occurance. Duplicate values are permitted. VALUE must be an array whose dimension in the calling program is NUM and whose type is identical to that of RESULT.

3. IDX - Index to most recent element. IDX must be in

integer in the calling program. SEQUEN automatically increments IDX by INC with each call (line 5); whenever IDX threatens to exit the range from 1 to NUM, SEQUEN applies "wrap-around" arithmetic to restore it into range (lines 6-7; note 6). Initializing IDX to 1-INC causes the first call to SEQUEN to return the first element in VALUE.

4. INC - Sampling increment. INC must be an integer. It can be either positive or negative (setting INC to -1 produces the retrograde) but it should always be relatively prime to NUM.

5. NUM - Number of element in row (dimension of array VALUE in calling program).

SEQUEN is encoded as a subroutine, rather than a function, so that it may handle both integer-valued and real-valued repertories. Internally, the FORTRAN compiler regards both RESULT and VALUE as real symbols; SEQUEN's ambivalence toward type is possible only because the transfer from VALUE(IDX) to RESULT (line 9) includes no arithmetic.

-- Programming example 3-1: subroutine SEQUEN --

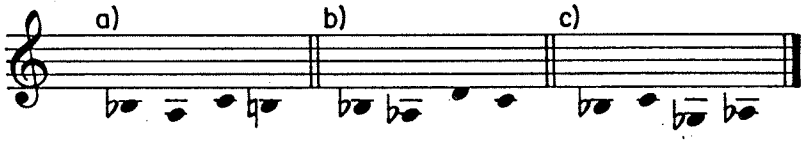
```
      subroutine SEQUEN(RESULT,VALUE,IDX,INC,NUM)
      dimension VALUE(1)
C     Increment index to current location in VALUE
C     "wrapping around" indices outside 1 to NUM
      IDX = IDX + MOD(INC,NUM)
      if (IDX.lt.1) then
        IDX = IDX + NUM
      else if (IDX.gt.NUM) then
        IDX = IDX - NUM
      end if
C     Transfer element from current location in VALUE to RESULT
      RESULT = VALUE(IDX)
      return
      end
```

3.2.2 Rote Transformations

The most straightforward translations can be implemented by generating a element of a row and then applying a simple formula. Depending on the context, adding a constant to successive elements will produce melodic transposition or metric displacements. Suppose, for example, that the integer array PCHROW contains MROW pitches indexed by the varible IDXROW. Then the following excerpt of FORTRAN '77 code will generate consecutive pitches of this tune, transposed by rote upward a perfect fifth:

-- Programming example 3-2 --

Similarly, multiplying successive elements by an integer constant will produce rote intervallic expansions or rhythmic augmentations while dividing by an integer will produce rote intervallic contractions or diminutions. It is necessary to indicate pitches relative to a reference pitch which does not transpose; for example, suppose that the integer array PCHROW contains 4 values:


a)
b)
c)

```
IDXROW = MROW
do (MROW times)
  call SEQUEN(IPCH,PCHROW,IDXROW,1,MROW)
  IPCH = IPCH + 7
  ...
repeat
```

```
IDXROW = 4
do (4 times)
  call SEQUEN(IPCH,PCHROW,IDXROW,1,4)
  IPCH = IEXPND*IPCH + ITRAN
  ...
repeat
```

```
IDXROW = 8
do (8 times)
  call SEQUEN(ISCL,SCLROW,IDXROW,1,8)
  ISCL = 2*(ISCL-7) + 7
  IPCH = PCHSCL(ISCL)
  ...
repeat
```

0, -1, 2, 1.

Consider the following excerpt of code:

-- Programming example 3-3 --

For ITRAN=46 and IEXPND=1 (first pitch on Bb3 and no interval expansion), this code will generate the sequence of pitches depicted in Figure 3-4a; resetting IEXPND to 2 (doubling each interval) will generate the pitches depicted in Figure 3-4b. Negative values of IEXPND produce melodically inverted expansions; Figure 3-4c depicts the sequence of pitches obtained by setting IEXPND to -2.

Figure 3-4: Intervallic expansions.

3.2.3 Modal Transformations

Modal tranformations such as the ones illustrated in Figure 3-1 are effected by considering each number as a position in a scale rather than as a chromatic pitch. Suppose, for example, that the integer array PCHSCL provides conversion from diatonic

to chromatic pitches for a C major scale running upwards for 2 octaves above middle C (PCHSCL(1)=48). Suppose also that the integer array SCLROW contains the following 8 values:

7, 8, 10, 9, 6, 5, 3, 4.

These values correspond to the "prime melody" depicted in Figure 3-1, while the following excerpt of code produces the melodic transformation depicted in the same Figure under "intervallic expansion by 2":

-- Programming example 3-4 --

Among the most important of modal transformations is modal transposition, which retains a diatonic motive's original contour while changing the motive's intervallic structure. Emmanuel Ghent has coined the term "translocation" in order to distinguish this type of transposition from the rote variety illustrated under the preceding heading. Modal transpositions, or "translocations", may be effected simply by adding a constant prior to converting from diatonic to chromatic pitches.

3.2.4 Permutations

Permutations, including retrogrades, affect the order in which elements are extracted from an array. There are two approaches to implementing permutations. One way is to manipulate the array index directly; in subroutine SEQUEN, for example, the user simply sets INC to -1 in order to obtain the retrograde.

An alternate approach involves reordering the array itself prior to each individual statement of the row. Rotation is an instance of a permutation which can only be obtained via SEQUEN by physical reordering. The library subroutine ROTATE is a generalized utility which accepts an array of arbitrary dimension and shifts the elements of this array an arbitrary number of positions. Calls to ROTATE require three arguments:

1. VALUE - Array of values, arranged in order of occurance. VALUE must be an array whose dimension in the calling program is NUM. VALUE may be either a real or integer array.

```
 1            subroutine ROTATE(VALUE,NSHIFT,NUM)
 2            dimension VALUE(1)
 3      C     Determine number of shifts N in range from 0 to NUM-1.
 4            N = MOD(NSHIFT,NUM)
 5            if (N.lt.0) N = NUM + N
 6      C     Shift each element of array VALUE clockwise by N positions
 7            do (N times)
 8      C       Shift values clockwise by one position
 9              V2 = VALUE(NUM)
10              do (I=1,NUM)
11                V1 = VALUE(I)
12                VALUE(I) = V2
13                V2 = V1
14              repeat
15            repeat
16            return
17            end
```

2. NSHIFT - Number of positions each element is to shift. If NSHIFT is positive, then the shift will be rightward/clockwise; if NSHIFT is negative, then the shift will be leftward/counter-clockwise.

3. NUM - Number of elements in array (dimension of VALUE in calling program).

The first step taken by ROTATE is to eliminate unnecessary shifts resulting from the fact any element of VALUE shifted by NUM positions returns to its original location (line 4). The next step (line 5) converts counter-clockwise rotations into clockwise ones. ROTATE accomplishes its main task through a structure of nested loops: the inner loop (lines 12-18) shifts each element in VALUE by one position while the outer loop (lines 8-19) repeats this operation for the requisite number of shifts.

-- Programming example 3-5: subroutine ROTATE --

We now consider an example implementing rotation as a musical transformation. Given arrays SCLROW and PCHSCL as defined immediately above, the following excerpt of code produces the melodic transformation depicted in Figure 3-1 under "Leftward rotation by 3":

```
call ROTATE(SCLROW,-3,8)
IDXROW = 8
do (8 times)
  call SEQUEN(ISCL,SCLROW,IDXROW,1,8)
  IPCH = PCHSCL(ISCL)
  ...
repeat
```

-- Programming example 3-6 --

3.3 CONTEMPORARY ANTECEDENTS TO AUTOMATED LITERAL SELECTION

Though the procedures discussed in this chapter operate by straightforward mechanisms, it would be wrong to assume that they were inspired in any way by the computer age. In fact, the bulk of these procedures are drawn either from traditional canonic practices or from formalized compositional approaches developed during the first half of this century. In order to appreciate these methods fully, it is necessary to look at where they came from and why they were developed.

3.3.1 Arnold Schoenberg

Arnold Schoenberg (1874-1951) devised his twelve-tone system with two major objectives: The first objective grew out of late nineteenth century chromaticism and the feeling among Schoenberg and his pupils that a greater proliferation of chromatic resources produced a greater "boldness" of expression (note 2).

Schoenberg extrapolated upon this premise to conclude that "boldness" would be greatest if all twelve degrees were always present. The second objective responded to a structural dilemma brought through Schoenberg's abandonment of traditional keys; he felt that rigorous thematicism derived from canonic practices would compensate for lack of tonal structure.

To use Schoenberg's twelve-tone system, a composer first selects a row (also called a series or basic set) containing all twelve degrees of the chromatic scale without repetitions. Every note in the composition is derived from this row, which may appear in any of four forms: prime (also called original), inversion, retrograde, and retrograde inversion. Each form may appear in any of its twelve transpositions.

Mere numeric balance between the twelve chromatic degrees is insufficient to neutralize a sense of key. Schoenberg himself was quick to point out this fact, which Alban Berg (1885-1935) demonstrated empirically in the latter's Violin Concerto (1935). However, Schoenberg's strictures concerning the use of the row go a long way toward defining the nature of a composition. His requirement that no degree in a row be repeated until the whole row has been stated imposes a pan-chromatic idiom, and in works such as his Variations for Orchestra (1928), Schoenberg took elaborate steps to insure chromatic

saturation even between superimposed halves of different row-forms.

Like the canons of earlier times, Schoenberg's twelve-tone compositions adhere to stylistic norms, in this case the principles of "dissonant counterpoint:" parallel movement is usually avoided, as are consonant sonorities, especially octaves. One way in which Schoenberg's method 'transcends' canonic imitation is through his proviso that adjacent degrees may be used either melodically or chordally. There is nothing revolutionary here; melodies have outlined broken chords for centuries. What is significant is that the row can provide not only the thematic vocabulary of a compostion, but the chordal vocabulary as well.

3.3.2 Joseph Schillinger

An antipode to Schoenberg was Joseph Schillinger (1895-1943). Schillinger was a modern-day Pythagorean, believing that all that is truly aesthetic in art has an underlying mathematical symmetry (note 3). This mathematical orientation often led Schillinger to derive his musical patterns rigorously from the simplest musical elements (e.g. single intervals or

single durations), much as a true mathematician will derive a theorem rigorously from "first principles". In other instances, Schillinger ~~he~~ derives material from numeric series such as the Fibonacci series and from straight lines or sinusoidal curves drawn on graph paper.

In the two lengthy volumes entitled The Schillinger System of Musical Composition (1941), Schillinger details strict recipes for generating rhythms, scales, melodies, chords, chordal progressions, and entire forms. Schillinger's transformations embrace intervallic expansions and rotations in addition to all of the traditional canonic devices. His productions often involve phasing, and they can be quite elaborate. Also included is a unique method of subtractive composition in which changing portions of a repeating ~~mass~~ aggregate are selectively deleted.

Figure 3-3: Joseph Schillinger's method of deriving rhythms through "interferences of periodicies". Of the abbreviations, c.d. denotes "common denominator", c.p. denotes "common product", and r denotes "resultant". Copyright 1941 Carl Fischer.

Figure 3-3 depicts a relatively simple instance of the Schillinger System: a method of generating rhythms through "interferences of periodicies". According to Schillinger, this

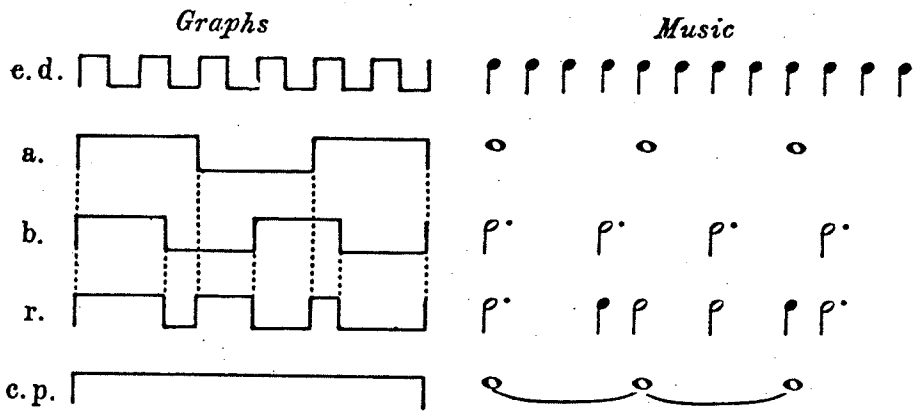


Fig. 3-3

pattern represents "the natural nucleus of a musical score, in which c.d. units are arpeggio or obligato forms, a and b are chords, r rhythms of the theme, and c.p. sustained tones ('pedal point')."

Schillinger himself composed works such as a First Airphonic Suite, Op. 21 (1929). However, his major impact was as a theorist. He was confident enough in his aesthetic to criticize Bach and Stravinsky for failing to adhere (note 4). Even so, his ideas have been influential among composers as diverse as George Gershwin and Earle Brown. Many successful film composers of the 1930's were students of Schillinger (note 5).

3.3.3 Olivier Messiaen

Olivier Messiaen (b. 1908) incorporates a curious blend of Christian mysticism, exotic material ranging from orientalisms to birdsong, conventional French modality, and literal procedures. Many of the last are documented in his book, The Technique of My Musical Language (1944). There is a strong resemblence to Schillinger -- indeed, many of Schillinger's formulations are more general than Messiaen's -- however, just as strong antecedents of Messiaen's practices may be seen in his French

predecessors, including Claude Debussy and Erik Satie.

Figure 3-4: Material for Olivier Messiaen's Mode of Values and Intensities. Copyright 1949 Durand.

One of Messiaen's most radical compositions using techniques not subsumed under Schillinger's edifice is a short etude for piano entitled Mode of Values and Intensities (1949). While the procedures used to compose this etude do not themselves fall under the heading of "literal selection", the piece constitutes a seminal influence upon the development of European serial music, so we treat it here as a matter of historic continuity. The etude consists of three ongoing contrapuntal parts. At the outset, Messiaen organizes five musical attributes -- or in serialist jargon, "parameters" -- by establishing a "mode" divided into three "divisions", one for each part. This mode is reproduced here as Figure 3-4. In each "division", Messiaen assigns a fixed register, duration, dynamic, and articulation to each of the twelve chromatic degrees. The music is constrained so that no two parts sound the same chromatic degree simultaneously, and neither do they sound the same duration, dynamic, or articulation. For example, if the top part has a dotted quarter, then neither of the other parts may begin a dotted quarter until the top part proceeds to a new note. The

original "divisions" do not appear in their entirety -- Messiaen allows himself the freedom to use their elements in any order and to break off a statement at any time -- but they do supply casual thematic material.

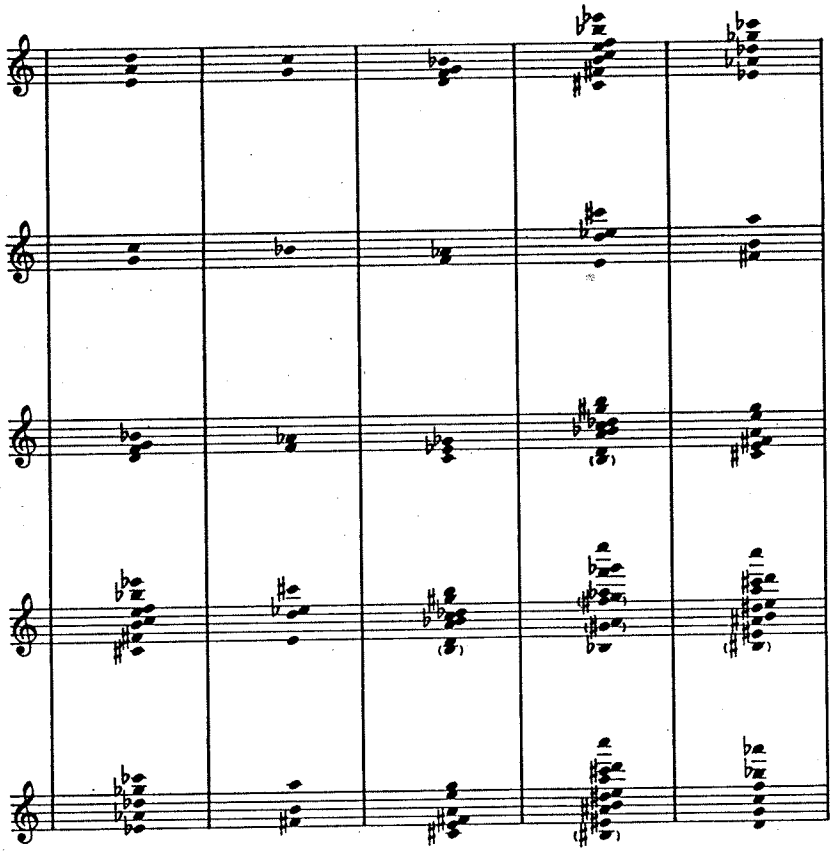
3.3.4 Pierre Boulez

The step of using a full fledged series to control each of the "parameters" was first taken in Europe by Messien's student Pierre Boulez (b. 1925). The first movement of in Book I of Boulez's Structures for two pianos (1952) has been described exhaustively by Ligeti (1958). All sequences of chromatic degrees, durations, dynamics, and articulations in this movement are derived from two 12 by 12 arrays giving the original series and its inversion in each of their transpositions; these arrays also supply the retrogrades of each transposition, reading backward. The compositional process was novel in that once Boulez had established correspondances between the integers 1-12 and the various musical attributes (degrees, durations, and so on), he ignored the latter and worked directly with the numbers. As Ligeti points out, the various "transpositions" of Boulez's duration series are impossible to hear as such, while the series

of dynamics and articulations are even more obscure.

In later works, Boulez broadens the scope of his "parameters" to include musical attributes defined over several measures, such as tempo and complexity of texture. He discusses this generalized scope in Boulez on Music Today (1963), where he also mentions a new technique of derivation which he calls multiplication (note 6). Boulez uses multiplications extensively in Book II of his Structures for two pianos (1958-1962). The analysis by Haeusler (1965), for example, reveals that the chordal vocabulary of Part 1 results from the "matrix" of chords shown in Figure 3-5, which Boulez derives by multiplying portions of a twelve tone row by itself. In order to multiply two chords to create a third, it is necessary to transposes each pitch occuring in the first chord by each of the intervals occuring in the second chord. Boulez's practice in Part 1 of Structures, Book II is to discard all duplicate degrees and to use the resulting chords in freely transposed and revoiced forms, so his procedures are not readily audible. Multiplications are much easier to hear ~~more readily discernable~~ in Part 2 of the same work.

Figure 3-5: "Multiplication" in Book II of Pierre Boulez's Structures for two pianos (1963), after Haeusler (1965) - This matrix has been derived by



multiplying each chord in the second column (also second row) by each other chord in the same column. Pitches bracketed in parenthesis indicate doublings at one or more octaves.

3.3.5 Milton Babbitt

In the United States, the most prominent composer employing Schoenberg's twelve-tone method has been Milton Babbit (b.1916). Babbitt has been an active proponent of using modular arithmetic to describe twelve-tone procedures. His 3 Compositions for Piano (1948) antedate Boulez's Structures, Book I in their serialization of attributes other than pitches. Since then Babbitt has derived a unique method for generating serial rhythms called the time-point system, which he has used in compositions realized on the RCA Mark II Synthesizer (Babbitt, 1962).

3.4 AUTOMATED LITERAL SELECTION

Though serial procedures lost their momentum among the

avant-garde by the end of the 1950's, they have enjoyed continued attention among practitioners of automated composition. This development is unsurprising in view of the relative ease of implementing serial procedures on a computer. Much activity during the 1970's has also been directed toward developing generalized utilities for composition by computer, and such utilities have naturally provided a wide range of literal transformations.

3.4.1 Direct Implementations: Barbaud, Hiller

The first serial composing programs were developed by Pierre Barbaud in 1961. Barbaud's programs generated tone rows at random and then transformed them using the classic twelve-tone transformations along with several others including interval expansions and rotations (Barbaud, 1966; Hiller, 1970).

Barbaud was followed ~~in followed~~ in 1963 by Hiller and Baker, who implemented the procedures of Boulez's Structures, Book I, "almost in toto" to compose the "Prolog to Strophe II" and "Epilog to Strophe IV" of their eclectic Computer Cantata (described in Hiller and Baker, 1964). In these movements, the prime material was a series consisting of the integers from 1 to 12, randomly shuffled; the transformations existed as closed

subroutines which Hiller and Baker drew their MUSICOMP package (heading 1.1.1), while the productions were provided by the composers as an explicit program. Hiller has continued to employ literal procedures in compositions up to the present. For example, Hiller uses change-ringing procedures in Algorithms II (197?), composed in collaboration with Raveesh Kumra for 9 instruments and tape. This work has four "versions" which serve for all practical purposes as movements; versions 3 and 4 apply pairwise exchanges to transform both melodic and rhythmic material (Hiller and Kumra, 1979).

3.4.2 Real-Time Composition: Emmanuel Ghent and Laurie Spiegel

The extreme simplicity of implementing literal procedures makes them especially attractive to real-time applications. Such procedures have been used very extensively in conjunction with Max Mathew's GROOVE system (the name stands for Generating Realtime Operations On Voltage-controlled Equipment), which allows composers to supply their own programs in order to interpret data obtained in real-time from devices such as a keyboard, a drawing tablet, or a bank of knobs.

Two of the composers associated with GROOVE during the

middle 1970's have been Laurie Spiegel and Emmanuel Ghent. Laurie Spiegel's compositions such as Patchwork (1974) were created as 'high-level' performances in which programs written by Spiegel monitored GROOVE's input devices in order to "derive from it much more complex music than [Spiegel] actually played." (Spiegel, 1980) Patchwork derives from four melodic patterns and four rhythms, which Spiegel manipulated in real time using the traditional canonic transformations.

Figure 3-6: "Pitch set" for Emmanuel Ghent's Lustrum.

Figure 3-7: Emmanuel Ghent: Lustrum, measures 207-211 - The material being played is: violin, F1 (original); viola, F2 (original); 'cello, F1 (inversion); bass, F2 (retrograde inversion); trumpet 1, F2 (retrograde); trumpet 2, F2 (inversion).

An example of Emmanuel Ghent's work with GROOVE is Lustrum for electronic string quintet, brass quintet, and computer-generated tape (1974). Lustrum also exists in an all-computer version entitled Brazen. The work is based entirely on the 15-pitch mode (Ghent uses the term "pitch set") depicted in Figure 3-6. From this mode, Ghent employs an

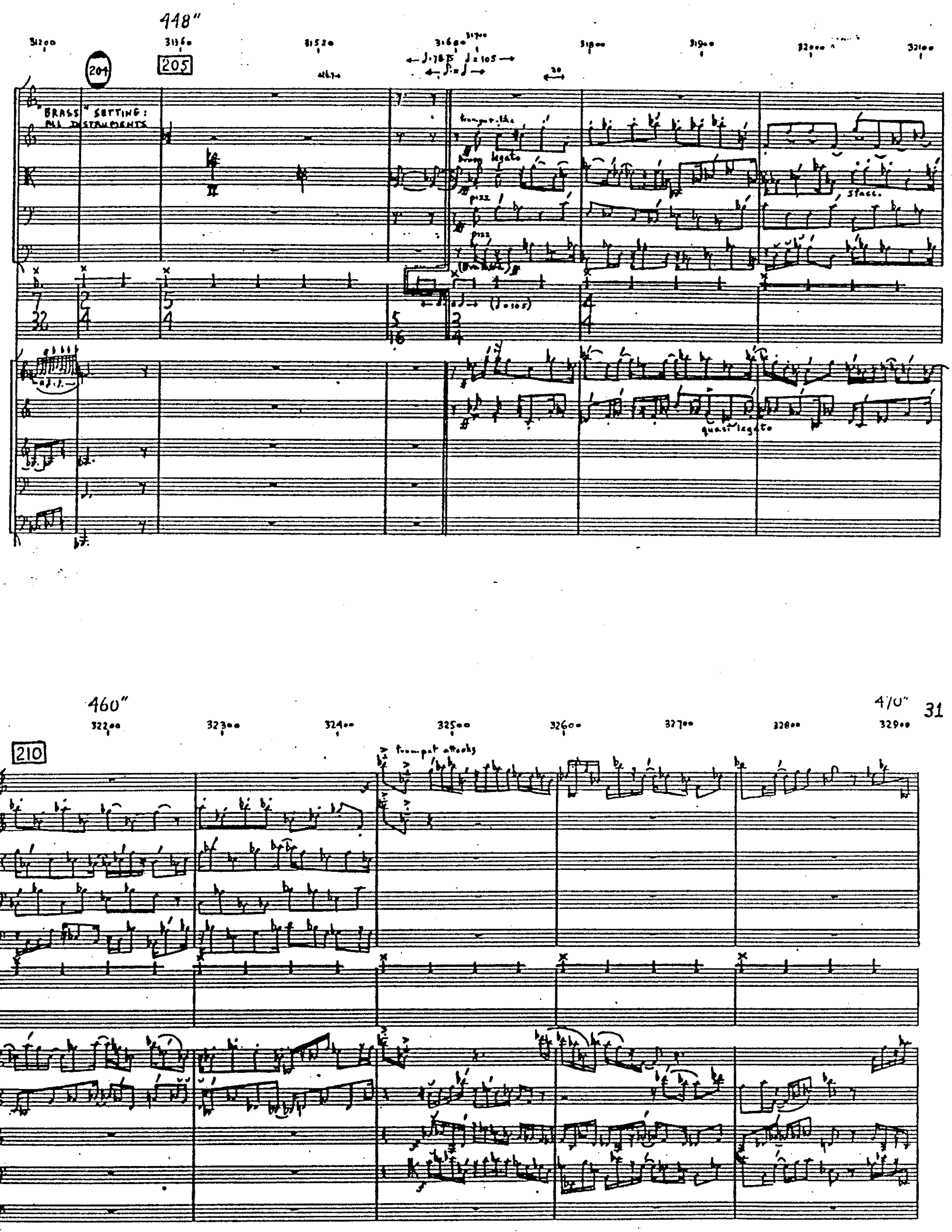

448"
204
205
"BRASS" SETTING:
ALL INSTRUMENTS
trumpet-like
bring legato
pizz
stacc.
quasi legato
460"
470"
31
210
trumpet attacks

Figure 3-7

algorithm which selects pitches and durations according to weighted probabilities in order to generate ten "functions" which Ghent denotes using the symbols F1 through F10. The "functions" in turn provide all of the material appearing in Lustrum. Each may occur either in its original form or in a transformed version. Included among Ghent's repertory of transformations are inversion, retrograde, and transposition; all of these transformations may be applied either directly (subsequent to consulting the pitch set in Figure 3-6) or modally (prior to consulting the set).

3.4.3 Literal Utilities

The compositional features of Leland Smith's SCORE program, originally designed as a "preprocessor" for digital synthesis but later converted into a music-printing program, reflect the serial practice of treating each of the musical "parameters" independently. A user describes each musical voice separately, and encodes each attribute (pitch, duration, articulation, loudness, and so on) as a separate string of mnemonics. In addition to mnemonics for letter names and rhythmic neumes, Smith's code provides shorthands for musical patterns which the

user can subject to various literal transformations. It also provides features for limited random selection and for describing gradual evolutions (Smith, 1972).

William Buxton and Barry Truax have developed utilities for literal procedures which, though independently developed, bear many strong resemblences. In both Buxton's SCED ("SCore EDitor"), and Truax's PODX composing environment, a composer begins by defining his prime material as a limited collection of motives. This material is stored as files called "subscores" (Buxton's term). The source of a prime subscore is arbitrary; prime subscores may be manually composed or, in the case of PODX, randomly generated. Both PODX and SCED provide repertories of basic "operators" (again Buxton's term) for deriving new subscores from old ones. Included are such productions as mixing and splicing along with such transformations as inversion, retrograde, transposition, augmentation, and so on; random transformations are also available. In SCED, more complex operations may be derived by linking two or more basic operators to create a "macro".

Otto Laske (1980) describes a number of very ambitious macros used with SCED to create his Terpsichore (1980) for computer-synthesized tape. The prime material for Terpsichore was derived externally from SCED using Gottfried Michael Koenig's PROJECT1 program. It consists of seven subscores numbered for

reference from 1 to 7; the styles of these subscores ranged along a continuum from serial-style "non repetition" in subscore 1 to "utmost recurrance" in subscore 7. Terpsichore as a whole spans three movements, each derived from the prime material by a different macro:

Movement A derives from subscores 7 and 6. The basic process mixes each half of a subscore with its own retrograde.

Movement B derives from subscores 5, 4, and 3. The operations consist of Boulez's procedure of "multiplication", retrograding, and splicing.

Movement C derives from subscores 2 and 1. The operations are very elaborate; they include scaling of durations (as opposed to periods) in order to create detached and slurred effects. Other operations include inversion, retrograde, and imitative stretto.

3.5 DEMONSTRATION 1: LITERAL SELECTION

Demonstration 1 illustrates the basic design of a complete composing program. In particular, it shows how a composer might implement literal strategies to derive not only the musical details, but also the broader level of musical structure.

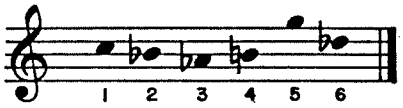
3.5.1 Compositional Directives

The piece derives entirely from the three rows depicted in Figure 3-8. It has three levels of structure: the local level consists of individual notes; the median level consists of 24-note phrases; and the global consists of the piece as a whole, which spans four entire phrases.

Figure 3-8: Compositional Data for Demonstration 1.

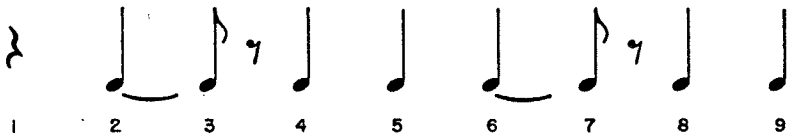
Figure 3-9: Derivation of a melodic resultant.

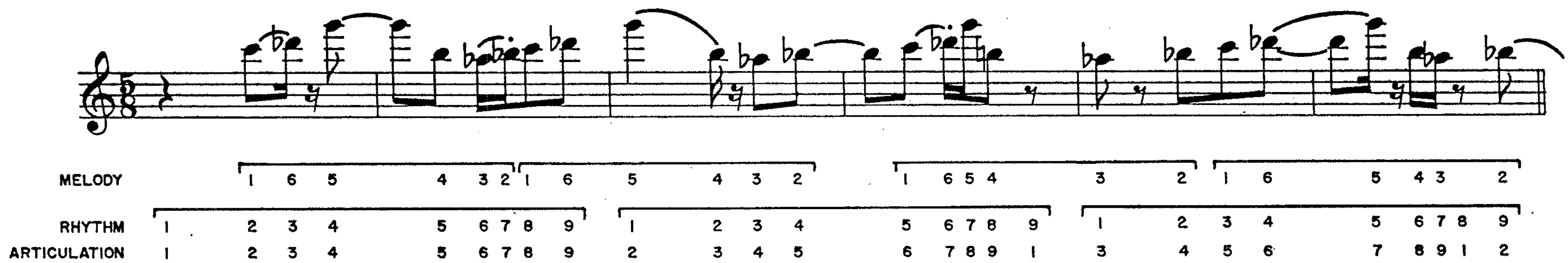
Prime melody



Prime rhythm

Prime articulations




MELODY 1 6 5 4 3 2 1 6 5 4 3 2 1 6 5 4 3 2 1 6 5 4 3 2
RHYTHM 1 2 3 4 5 6 7 8 9 1 2 3 4 5 6 7 8 9 1 2 3 4 5 6 7 8 9
ARTICULATION 1 2 3 4 5 6 7 8 9 2 3 4 5 6 7 8 9 1 3 4 5 6 7 8 9 1 2

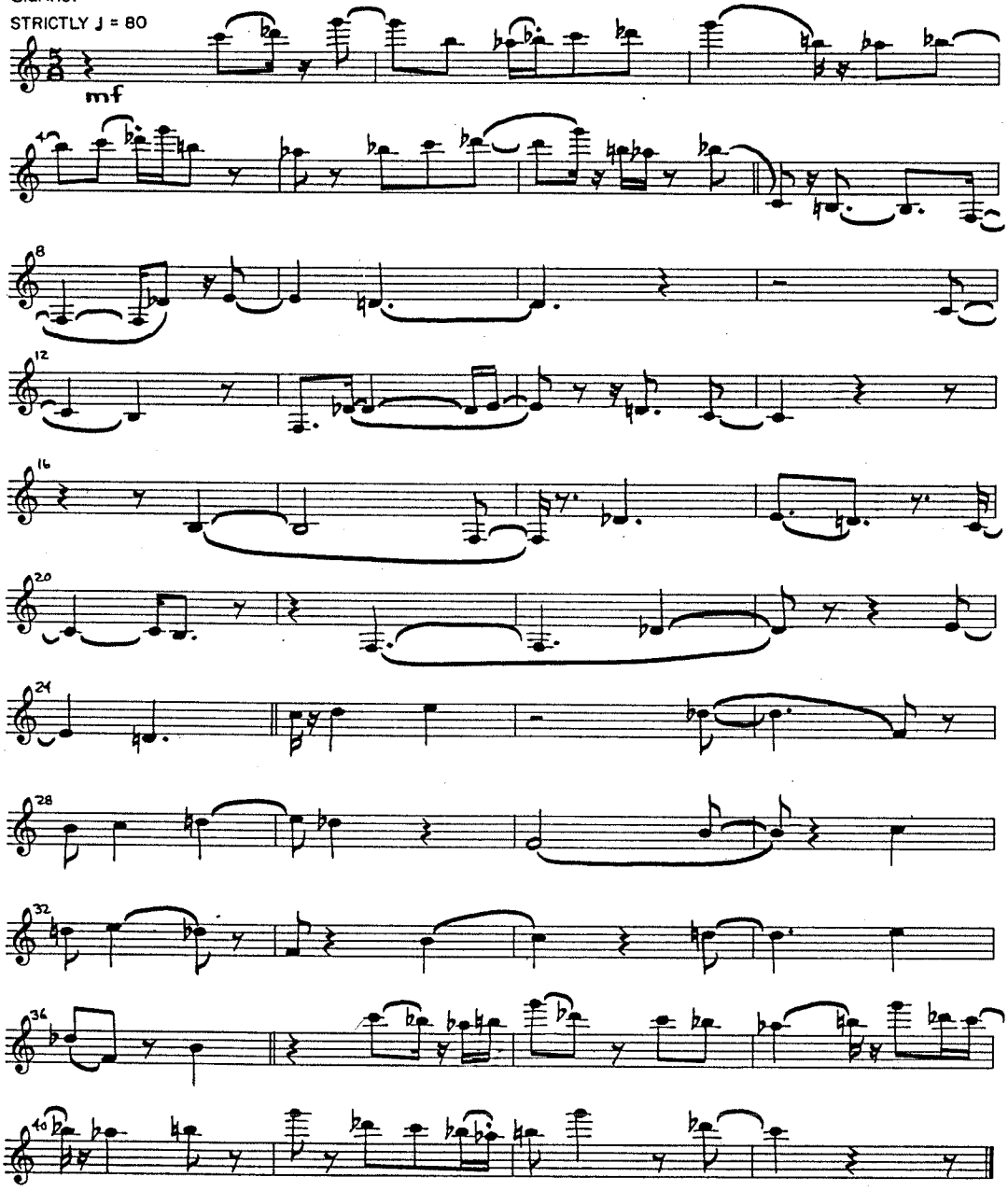
Measure	Pitches	Periods
1	High register Retrograde	No augmentation Prime form
7	Low register Retrograde - inversion	Augmentation by 2 Inversion
25	Medium register Inversion	Augmentation by 3 Retrograde - inversion
37	High register Prime form	No augmentation Retrograde

Demonstration I

Clarinet

Charles AMES

STRICTLY ♩ = 80



Each phrase in Demonstration 1 constitutes a resultant created by phasing four cycles of the pitch row against three cycles of the period row. Figure 3-9 illustrates how the opening resultant of the piece is derived. Concurrent with each cycle of periods is a cycle of the nine articulations. Notice that the cycles of pitches and periods come back into alignment at the end of the phrase because one element in each cycle of articulations is always a rest.

Table 3-1: Global Recipe for Demonstration 1.

Figure 3-10: Transcription of Demonstration 1.

Table 3-1 gives the layout of Demonstration 1 and details which transformations have been applied to the pitch and period rows in each phrase. The complete musical product appears in Figure 3-10.

3.5.2 Implementation

-- Programming example 3-7: program DEM01 --

```
      program DEMO1
C
C     Demonstration of literal selection
C
      open (2,file='DEMO1.DAT',status='NEW')
      ITIME = 0
      call PHRASE(ITIME,72,1,-1, 1, 1, .true.)
      call PHRASE(ITIME,48,3,-1, 1,-1,.false.)
      call PHRASE(ITIME,60,2, 1,-1,-1,.false.)
      call PHRASE(ITIME,72,1, 1,-1, 1, .true.)
      call WNOTE(ITIME,4,72,2)
      close (2)
      end

      subroutine PHRASE(ITIME,IREG,IAUG,INCPCH,INCPER,ISGN,RHYFLG)
      parameter (MPCH=6,MPER=9,MART=9)
      integer VALPCH(MPCH),VALPER(MPER),VALART(MART)
      logical RHYFLG
      data VALPCH/0,-2,-4,-1,7,1/
      data VALPER/4,2,2,4,2,1,1,2,2/
      data VALART/0,3,1,2,2,3,1,2,2/
      data IDXART/0/
C
      IDXPCH = 1 - INCPCH
      IDXPER = 0
      do (27 times)
C       Determine articulation
        if (IDXART.eq.MART) call ROTATE(VALART,-1)
        call SEQUEN(IART,VALART,IDXART,1,MART)
C       Determine period
        call SEQUEN(IPER,VALPER,IDXPER,1,MPER)
C       Invert period (where applicable)
        if (RHYFLG) IPER = 4 / IPER
C       Augment period
        IPER = IAUG * IPER
C       Test for rest (IART=0)
        if (IART.eq.0) then
          IPCH = 0
        else
C         Determine relative pitch
          call SEQUEN(IPCH,VALPCH,IDXPCH,INCPCH,MPCH)
C         Invert pitch (if applicable) and apply transposition
          IPCH = ISGN*IPCH + IREG
        end if
C       Write out note
        call WNOTE(ITIME,IDUR,IPCH,IART)
      repeat
      return
      end

      subroutine WNOTE(ITIME,IDUR,IPCH,IART)
C
C     Subroutine for converting note data to mnemonic form
C
      character*2 MNEDEG(12)
      character*4 MNEART(3)
      data MNEDEG/' C','Db',' D','Eb',' E',' F',
     :            'F#',' G','Ab',' A','Bb',' B'/
      data MNEART/'stac','norm','slur'/
C
      MEAS = ITIME / 10
      IBEAT = ITIME - 10*MEAS
      MEAS = MEAS + 1
      if (IART.eq.0) then
        write (2,10) MEAS,IBEAT,IDUR
   10   format (I5,':',I1,2X,I2,2X,'Rest')
      else
        IOCT = IPCH / 12
        IDEG = IPCH - 12*IOCT + 1
        write (2,20) MEAS,IBEAT,IDUR,MNEDEG(IDEG),IOCT,
     :               MNEART(IART)
   20   format (I5,':',I1,2X,I2,2X,A2,I1,2X,A4)
      end if
      ITIME = ITIME + IDUR
      return
      end
```

The design of program DEMO1 reflects this three-tiered structure in that attributes affecting entire phrases are determined by the main program, DEMO1 proper, while attributes affecting individual notes are calculated by subroutine PHRASE with assistance from the library subroutine SEQUEN. Data describing each note is then converted into mnemonics by subroutine WNOTE, which also writes these mnemonics out to a mass-storage file called "DEMO1.DAT".

3.5.2.1 Phrases - Program DEMO1 implements the layout detailed in Table 3-1 through four calls to subroutine PHRASE (lines 7-9 of DEMO1) which provide this information in the form of explicit arguments. A final call to subroutine WNOTE (line 10) acts to close the piece on a quarter-note C5 with normal articulation.

3.5.2.2 Notes - The three rows controlling pitches, periods, and articulations are declared in lines 5-7 of subroutine PHRASE. The associated symbols in PHRASE adhere to three mnemonic 'roots':

1. PCH - pitch, expressed as an offset from the first note in the row.
2. PER - period between consecutive attacks.
3. ART - articulation (slurred, normal, staccato, or rest).

The length of each row is given by a parameter starting with the letter M; values of the prime row-forms reside in arrays beginning with VAL. Each row also has an associated index beginning with IDX.

The transformations used to generate additional material for Demonstration 1 break down as follows:

1. Pitches

   a. Transpositions: transpositions are determined by variable IREG (line 29 of PHRASE. All transpositions in Demonstration 1 occur at the octave.

   b. Intervallic inversions: the variable ISGN (line 29) determines whether offsets from array VALPCH go upward (ISGN * IPCH positive) or downward (ISGN * IPCH negative).

   c. Retrogrades: the variable INCPCH determines whether

SEQUEN selects offsets by sampling forward or backward through array VALPCH (line 27).

2. Period between consecutive attacks

   a. Retrogrades: the variable INCPER (line 17 of PHRASE) determines whether SEQUEN selects periods by sampling forwards or backwards through array VALPER.

   b. Rhythmic inversions: the logical variable RHYFLG (line 19) determines whether or not PHRASE should invert the relative period IPER.

   c. Augmentations: multiplying by the variable IAUG (line 21) converts the relative period IPER into an absolute period.

3. Articulations are determined by samping array VALART (line 15). Each complete cycle through VALART causes PHRASE to rotate each element of VALART by one position by initiating a call to the library subroutine ROTATE (line 14).

3.5.2.3 Mnemonic Listing - The mechanics of subroutine WNOTE bear close attention because they serve for most of the Demonstrations in this book. When it calls WNOTE to append a note to the mnemonic listing, DEMO1 provides an action time in sixteenths, ITIME, a duration in sixteenths, IDUR, and a pitch, IPCH. The first task of WNOTE is to place ITIME in a measure (lines 8-19). The author finds it most convenient to begin measures on beat 0. Conversion of numeric degrees to letter names is easily accomplished using the character array LETTER. The convention is to express registers in octaves above 32'C (making 32' C C0; middle C is C4; A 440 is A4, and so on). The character array ARTIC converts numeric articulations into mnemonic indications.

3.6 NOTES

1. The terms <u>transformation</u> and <u>production</u> derive from linguistic concepts examined more fully in Chapter 11. In the immediately following text, the terms <u>translation</u> and <u>reflection</u> come from analytic geometry (e.g., "translation of axes"), while <u>permutation</u> comes from the mathematical theory of groups.

2. In Alban Berg's words: "rich in degrees, and bold -- the two are much the same" (Reich, 1968, page 32).

3. For all his numerological superstitions, Schoenberg believed very strongly that the final arbiter of all compositional procedures -- systematic or otherwise -- is the ear.

4. An opposing sentiment may by found in Stravinsky's statement (1958, p. 20): "... to be perfectly symmetrical is to be perfectly dead".

5. The editor's footnote on page 1477 of Schillinger's book gives a list of names and films.

6. Boulez's draws his term <u>multiplication</u> from the mathematical theory of groups. It is quite legitimately used.

7. Be wary of the MOD function! As implemented in most computer languages, this function uses the formula:

$$MOD(N,M) = N - M*INT(N/M).$$

This formula deviates from the "modulo" relationship described in discrete mathematics; for example, -14 modulo 12 yields 10 while

MOD(-14,12) yields -2.

3.7 RECOMMENDED READING

Ghent, Emmanuel, 1978. "Real-time interactive compositional procedures", Proceedings of the American Society of University Composers.

Laske, Otto, 1981. "Subscore Manipulation as a tool for compositional and sonic design", Proceedings of the 1980 International Computer Music Conference (Computer Music Association).

Schillinger, Joseph, 1941. The Schillinger System of Musical Composition (New York: Carl Fischer). 941).