A LISP ENVIRONMENT FOR CREATING AND APPLYING RULES FOR MUSICAL PERFORMANCE

Anders Friberg and Johan Sundberg
Royal Institute of Technology
Department of Speech Communication and Music Acoustics
S-100 44 Stockholm, Sweden

BACKGROUND

In a previous ICMC we reported on a project where performance rules are used for automatic note-to-tone conversion (Fryden & Sundberg 1984). The input is the the string of note signs and the output the corresponding sound sequences. These rules, developed on a minicomputer system controlling a hardware formant synthesizer (MUSSE), operate on shorter and longer musical contexts.

AIM

The aim of the present work was to develop a new, more general and user oriented system containing the rules just mentioned and allowing easy development and testing of both existing and new rules. It was considered essential to use a small computer that would make the system portable. We have chosen a Macintosh microcomputer with Le_Lisp as the programming language controlling any MIDI interfaced synthesizer.

DESCRIPTION

The input is the music notation. Facilities are provided to insert it using the "mouse",the Macintosh keyboard or directly from a synthesiser for obtaining a string of note symbols, one for each voice. These strings are read into the system and each of them can be displayed in conventional notation on a note staff.

The program organizes the music in voices that can be processed either serially or in parallel. Each voice consists of a list of notes each of which have a list of properties such as duration, amplitude, MIDI key number or any other property that has been assigned to the notes. The assignment of such properties is normally realized by rules using so called property access functions, but the properties can also be edited manually. The possibility to let rules assign new properties to the notes also allows implementation of hierarchical rule structures; a rule may trigger its subject rules via the note properties.

Exampel of a property list:

(dr 793 nomdr 854 a0 90 f0 50 bar 1 phrase 1 key "D")

duration = 793 ms, nominal duration = 854 ms, level 90 amplitude units, MIDI key number = 50, first bar, first phrase and key = D major

It is possible to have several voices available in the Lisp program, but all of them need not be processed by the rules at the same time. For instance, several copies of the same input melody can be stored in the program in terms of different voices, and these voices can then be operated on by different selections of rules. This allows immediate comparisons of different performances of the same melody, e.g. for the purpose of testing.

Example of a general rule construct :

```
(DEFUN <rule name> (<input parameters>)
  (EACH-NOTE-IF
    (<condition 1>)
    (<condition 2>)
        .
    (<condition n>)
        (THEN
          (<statement 1>)
          (<statement 2>)
          .
          (<statement n>)  )))
```

Translation: Define a function called "rule name" using the "input parameters" listed within the paretheses. EACH-NOTE-IF is macro function implying that the statements 1 through n specified after THEN be executed on all notes meeting the conditions 1 through n.

Within an EACH-NOTE-IF function several helpfunctions can be used, such as:

(FIRST?)
(LAST?)

These return the value TRUE if the current note is the first or the last note, respectively; otherwise the value FALSE is returned.

Examples of some of the property access functions:

(THIS <property >)
(NEXT <property >)
(PREV <property >)

These functions return the value of the specified property pertaining to the current, the next, or the previous note respectively.
The assignment of a particular value to a property of the current, next or previous note is made by means of functions such as these:

(SET-THIS <property> <value>)
(SET-NEXT <property> <value>)
(SET-PREV <property> <value>)
(ADD-THIS <property> <value>)
(ADD-NEXT <property> <value>)
(ADD-PREV <property> <value>)

There are also other useful functions available such as different ramp functions for crescendos, ritardandos etc.

As an illustration, we quote the following performance rule, which lengthens the final note of phrases by 40 msec and inserts a micropause of 30 ms:

```
(DEFUN phrase ()
  (EACH-NOTE-IF
     (NEXT 'phrase-start)
    (THEN
        (ADD-THIS 'dr  40)
        (SET-THIS 'dron
                  (- (this 'dr) 30) )
     )))
```

Translation: Define the function "phrase", using no input parameters: to each note followed by a note possessing the property "phrase-start" a quantity of 40 ms is added to the duration; also, the sounding part, i.e. "dron", of the same note is shortened by 30 ms.

Facilities are provided for displaying graphically or numerically any mathematical function of the note properties. For example, the amplitude or the difference between nominal and actual duration can be shown graphically, see Figure 1.

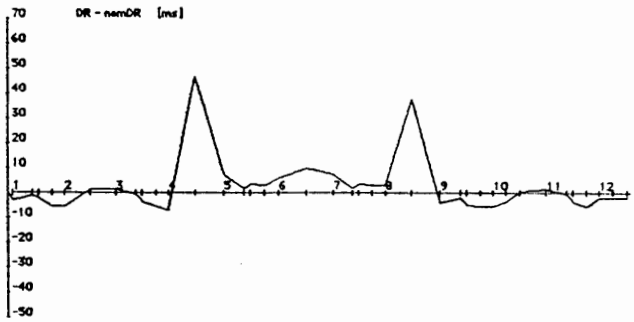

Figure 1. Difference between nominal and performed duration in the first twelve bars of a melody; the second half notes of the phrase terminating bars number four and eight have been lengthened by fourty ms.

Playing over the MIDI does not per se involve any restrictions as to tuning systems, such as the equally tempered scale. The system is designed primarily for conventionally notated music but in principle any form of control of musical events, organized in voices, can be used within the limits of the MIDI interface and the playing routine.

There are menus and dialogs for the most common operations allowing a fast user interaction, see Figure 2. These facilities are sufficient for using the system with existing rules. In other words, the system can be used without knowledge about lisp. New rules, however, are written in lisp as shown above.

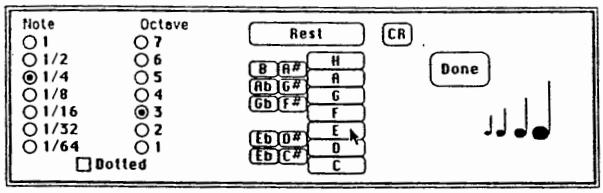

Figure 2. Display for writing the input music. As soon as the note value, the octave number, and the pitch name have been selected, the note appears on the display.

The system was successfully used in a listening experiment, for evaluating various performance rules (Thompson, Friberg & Sundberg, forthcoming). The subjects rated the musical quality of different versions of the same melodies. A simple program was written that ran the experiment automatically presenting the different performances in an order that was randomized for each subject and recorded the subjects' answers. The simplicity of realizing such a task demonstrates the strength of this lisp environment.

Acknowledgement
This work was supported by the National Bank of Sweden Tercentenary Foundation.

References

FRYDÉN, L. & SUNDBERG, J. (1984) Performance rules for melodies : origin, functions, purposes. International Computer Music Conference 1984.

THOMPSON, W., FRIBERG, A. & SUNDBERG, J. (forthcoming)