Guide to BOOK Chapter 13 Examples:
John ffitch

Using C to Generate Scores

Background

These notes assume that you are using a command-line interface and you have understood the methods described in the Introductory C chapters.

The techniques in this chapter develop the musical ideas in Richard Boulanger's piece *Trapped in Convert.* In order to appreciate the musical changes we make here, I recommend that you listen to the original. This can be synthesised in real-time using the command:

```
csound TrappedInConvert.orc TrappedInConvert.orc -odac
```

It may be useful to create a sound file of this piece so it can be listened to repeatedly and even in parts using a sound editor. On a Windows or GNU/Linux system run

```
csound TrappedInConvert.orc TrappedInConvert.orc \

-W -o trapped.wav
```

On a Macintosh it may be better to create an AIFF file:

```
csound TrappedInConvert.orc TrappedInConvert.orc \

-A -o trapped.aif
```

A recommended sound editor is **Audacity** which is cross platform, but you may have an alternative favorite.

Building and Running the *range* Programs

The simple first program, to test the range of the Henon process, is called `henonMinMax.c` and can be compiled and run with the commands:

```
gcc henonMinMax.c

a.out
```

This should print the output as described in the text.

The second range checking program is `henonValues.c`. It can be compiled and run using the same commands with the filename substituted. The program to collect the statistics of the frequencies that the integer values occur is `henonInRange.c`. It can be treated the same way.

Building – The First Attempt

The first attempt at a score generator requires us to build the generator and then create the score that will be used as part of the input to Csound. This gives the sequence of commands:

```
gcc trappedFirst.c

a.out > trappedFirst.sco

csound TrappedInConvert.orc trappedFirst.sco
```

that will generate the sound file as `test.wav` or `test.aif` depending on the settings of your Csound system. The sound file can then be played with your media player or command-line program, or loaded into a sound editor.

Creating the Final Piece

The C program for the final version can be found in `algoTrappedFinal.c`. The orchestra we will use for the final is `algoTrappedFinal.orc`. The complete process is sufficiently complex that a Makefile was written for it. The targets provided by this Makefile are: `DDC4.create –` to build the program from the C source; `algoTrappedFinal.sco –` to run the program and make the score; `DDC4.wav` – to run Csound and create the audio file; and `dac` – to run Csound in real-time mode. The default target is to create the audio. As above, the sound file can then be played with your media player or command-line program, or loaded into a sound editor.