ALGORITHMIC CONTROL OF SIGNAL PROCESSING

Roger B. Dannenberg

Generating Control and Audio Algorithmically

• Xenakis: GENDYN
• Roads: Microsound
• Granular Synthesis

• Signals (Sounds) controlled by Patterns
• Patterns controlled by Signals
Pat-ctrl

define function pat-ctrl(durpat, valpat)
  return seq(const(next(valpat),
               next(durpat)),
             pat-ctrl(durpat, valpat))

• What is the duration of a sound returned by pat-ctrl?
Controlling Frequency with Patterns

```plaintext
define function pat-fm(durpat, valpat, pitch, dur)
begin
    with hz = step-to-hz(pitch + pat-ctrl(durpat, valpat))
    return pwl(0.01, 1, dur - 0.1, 1, dur) *
        hzosc(hz + 4.0 * hz * hzosc(hz))
end
```

Using Scores

```plaintext
exec score-play(
    {{ 0 30 {pat-fm-note grain-dur: 8 spread: 1
        pitch: c3 fixed-dur: t
        vel: 50}}
    {10 20 {pat-fm-note grain-dur: 3 spread: 10
        pitch: c4 vel: 75}}
    {15 18 {pat-fm-note grain-dur: 1 :spread: 20
        pitch: c5}}
    {20 13 {pat-fm-note grain-dur: 1 spread: 10
        grain-dur: 20 pitch: c1}}})
```

Using Scores (2)

```lisp
(exec score-play
  (! 0 30 (pat-fm-note grain-dur: 8 spread: 1
            pitch: c3 fixed-dur: t
            vel: 55))
  (! 10 20 (pat-fm-note grain-dur: 3 spread: 10
            pitch: c4 vel: 75))
  (! 15 18 (pat-fm-note grain-dur: 1 spread: 20
            pitch: c5))
  (! 20 13 (pat-fm-note grain-dur: 20 pitch: c1))))
```

- Key ideas:
  - Scores do not have to consist of "notes"
  - Packaging a complex behavior as a Nyquist instrument (a behavior with keyword parameters) supports hierarchical composition
  - Via scores, programs, even score-gen

Using Nyquist SOUNDs for Global Control

- Scores are fine for events
- What about continuous change?
- Example from before: Tendency Masks:
Accessing Sound Values

• Solution: use SOUNDs to specify global continuous evolution of parameter values
• To access a sound: sref(sound, time)
  • sound is any SOUND type
  • time is relative to environment, so time=0 means "now"
• Remember that while behaviors start "now", existing sounds have a definite start time

Template for Global Control using Sounds

```lisp
define variable pitch-contour =
  pwl(10, 25, 15, 10, 20, 10, 22, 25, 22)
define function get-pitch()
  return sref(pitch-contour, 0)
define function pwl-pat-fm()
  begin
    ... 
    make-eval({get-pitch}),
    ... 
  end
play pwl-pat-fm()
```

Note: must be LISP expression
Contours in Score-Gen

begin
with pitch-contour = pwl(10, 25, 15, 10, 20, 10, 22, 25, 22),
ioi-pattern = make-heap({0.2 0.3 0.4})
exec score-gen(save: quote(pwl-score),
   score-dur: 22,
pitch: truncate(c4 +
   sref(pitch-contour,
   sg:start) +
   #if(oddp(sg:count), 0, -5)),
ioi: next(ioi-pattern),
dur: sg:ioi - 0.1,
vel: 100)
end
Examples (code_12.sal)