– Euterpea –

From Signals to Symphonies Using Haskell

Paul Hudak
Yale University
Department of Computer Science

Workshop on Functional Art, Music, Modeling, and Design

(FARM'13)
Boston, MA
September 28, 2013

Euterpea

- Haskell library for computer music.
- Support for both “note level” and “signal level.”
- Named after *Euterpe*, one of nine Greek *muses*, or goddesses of the arts, specifically the muse of *music.*
- Basis for textbook: *The Haskell School of Music* (available on-line, almost 400 pages).
- Used to teach two-term computer music sequence in Computing and the Arts major at Yale.
- Goal: Teach FP and Computer Music in symbiosis.
- Free download from haskell.cs.yale.edu.

Demo

- Basics
- (Musical) Performance
- Algorithmic Composition
- Musical User Interface (MUI)
- Sound Synthesis

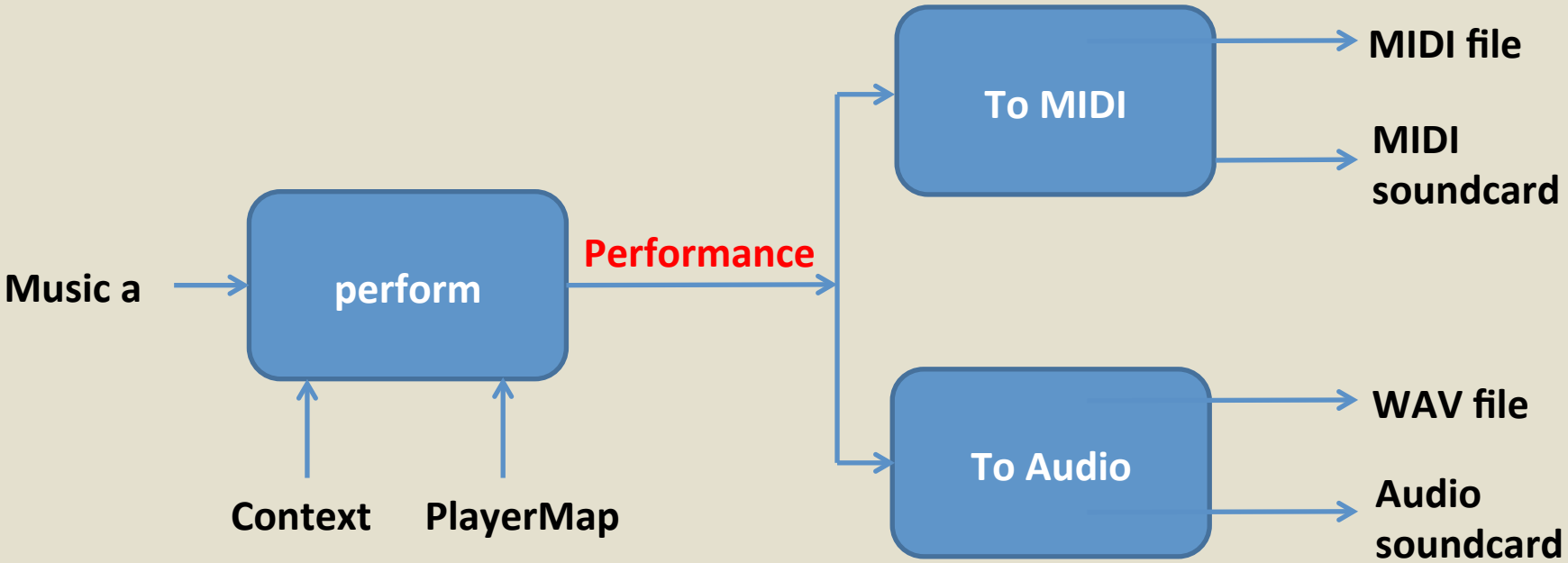
(to demo)

Handy Musical Functions

- delayM d m = rest d :+: m
- timesM 0 m = rest 0
  timesM n m = m :+: timesM (n – 1) m
- repeatM m = m :+: repeatM m
- takeM d m = … -- truncates m's duration to d
- m1 (/=:) m2 = … -- like (:=:), but stops after shortest
- line [ ] = rest 0 -- turns list of notes into Music
  line (m:ms) = m :+: line ms
  [better: line = foldr (:+:) (rest 0)]

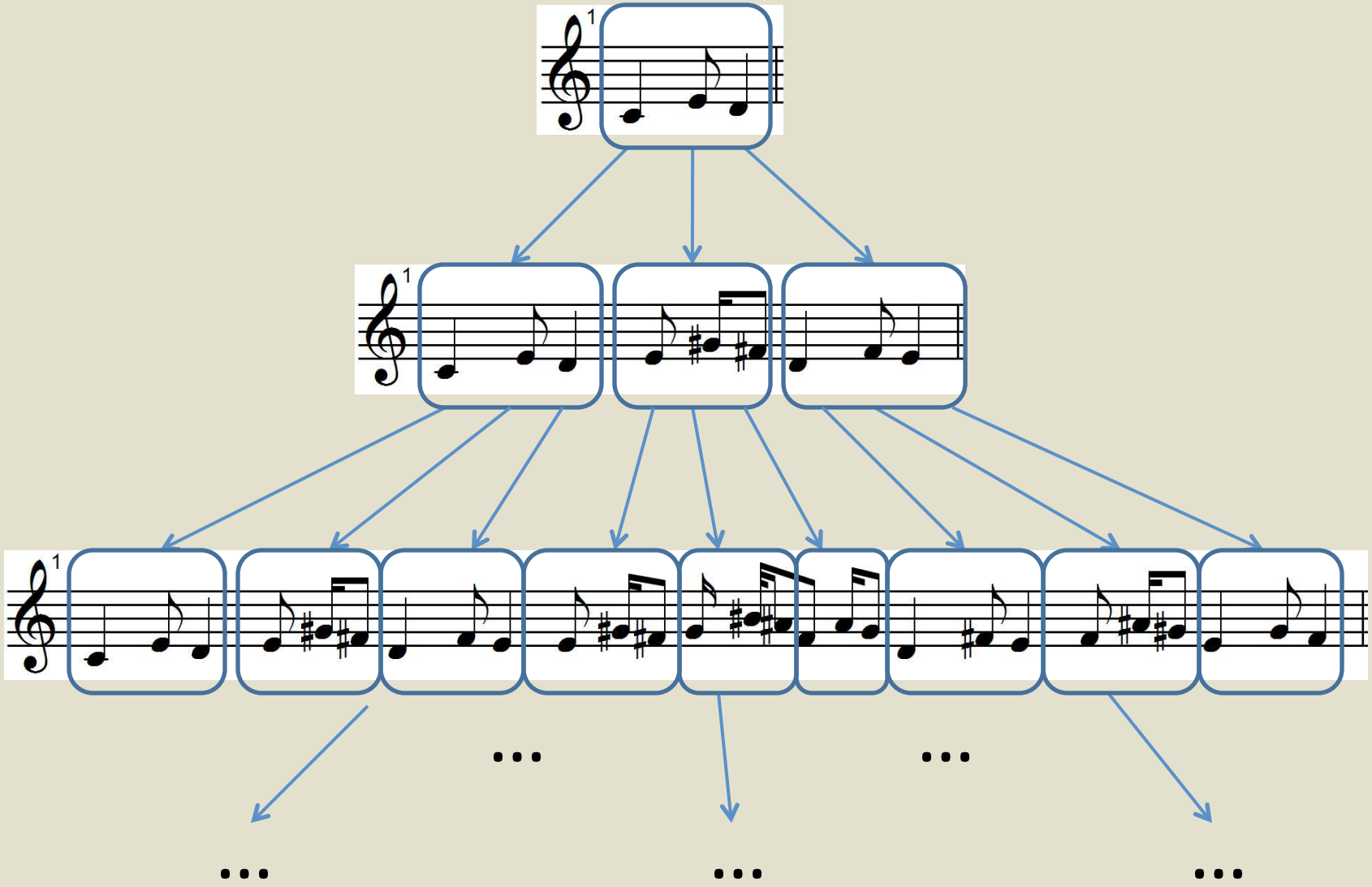
(to demo)

(Musical) Performance



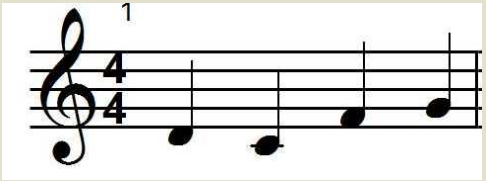
(to demo)

Example of Self-Similar Music



Example

- Use this 4-note motif as the seed:



- Traverse 4 levels in the tree.
- Play together with itself in reverse and transposed up a perfect fourth:

```
m  :=: transpose 5 (revM m)
```

(to demo)

Bifurcate Me, Baby!

- Consider the recurrence equation:

  $$x_{n+1} = r * x_n * (1 - x_n)$$

  Start with an initial population x_0 and iteratively apply the growth function to it, where r is the growth rate. For certain values of r, the population stabilizes, but as r increases, the period doubles, quadruples, and eventually leads to chaos. It is one of the classic examples in chaos theory.
- In Euterpea:

```
grow r x = r * x * (1-x)
```

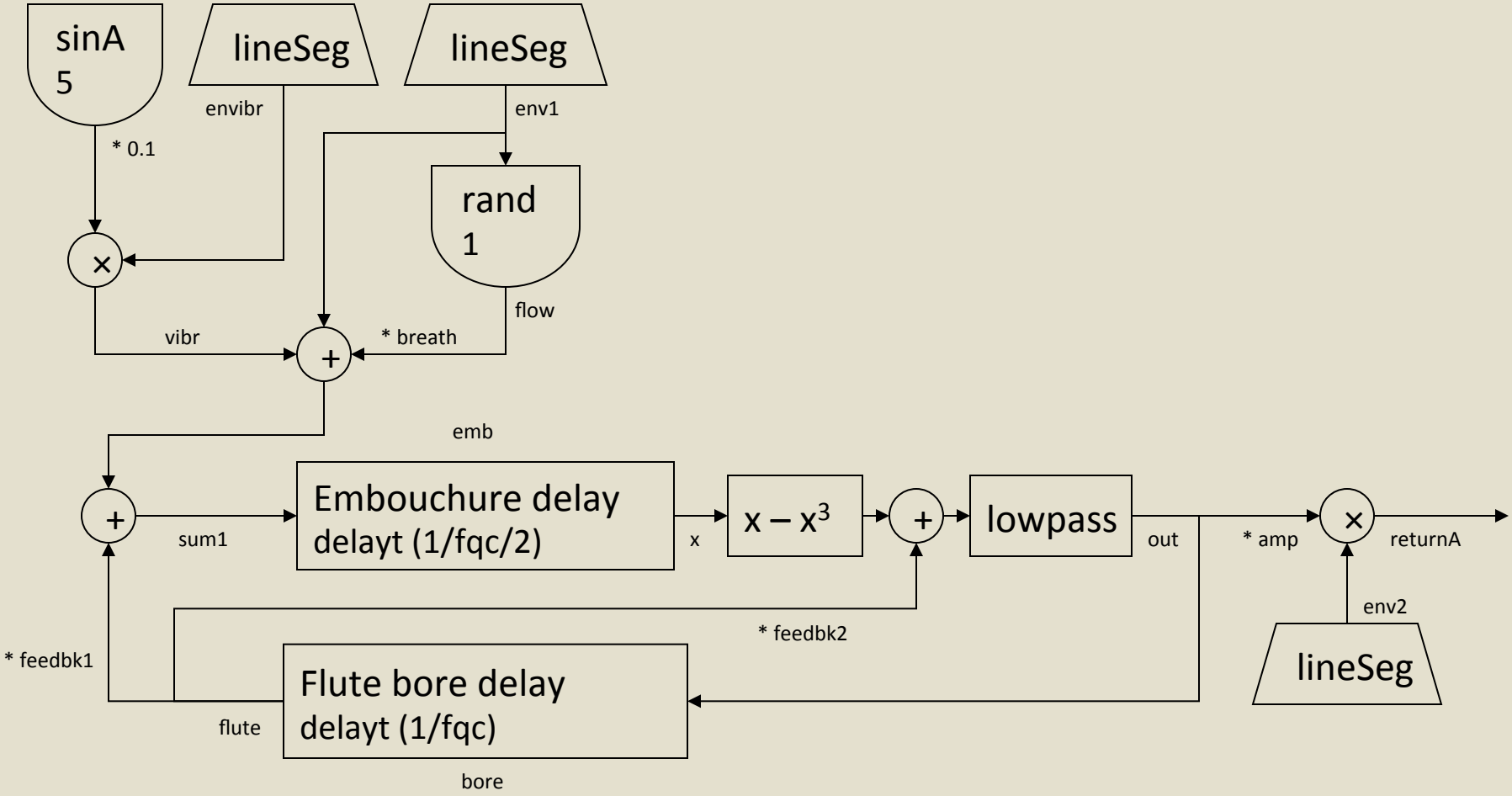
Bifurcate Me, Baby!

Gary Lee Nelson

1995

(to demo)

Physical Model of a Flute



Flute Model in Euterpea

```
flute dur freq amp vfreq =
    in proc () -> do
        amp1  <- linseg …       -< ()
        amp2  <- linseg …       -< ()
        ampv  <- linseg …       -< ()
        flow   <- rand 1         -< amp1
        vibr   <- oscils vfreq -< 0.1 * ampv
        rec
            let feedbk = body * 0.4
            body <- delay (1/freq)    -< out
            x      <- delay (1/freq/2) -< breath*flow
                                          + amp1 + vibr + feedbk
            out   <- tone                -< (x - x*x*x + feedbk, 2000)
        returnA -< out*amp*amp2
```

Flute Demo

- f0 and f1 demonstrate the change in the breath parameter.

```
f0 = flute 3 0.35 440 0.93 0.02
f1 = flute 3 0.35 440 0.83 0.05
```


- f2 has a weak pressure input so only plays the blowing noise.

```
f2 = flute 3 0.35 440 0.53 0.04
```

- f3 takes in a gradually increasing pressure signal.

```
f3 = flute 4 0.35 440
          (lineSeg [0.53, 0.93, 0.93] [2, 2])
          0.03
```

- Sequence of notes:


Thank You!!

(any questions?)

For more information about Euterpea, see:

haskell.cs.yale.edu