Faust Term Rewriting Extension

Albert Gräf

Dept. of Music Informatics

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

```
// tone.dsp

vol      = nentry("vol", 0.3, 0, 10, 0.01);
pan      = nentry("pan", 0.5, 0, 1, 0.01);
freq     = nentry("pitch", 440, 20, 20000, 0.01);

// simple sine tone generator

process  = osci(freq)*vol : panner(pan);
```

Signal Processing with Faust

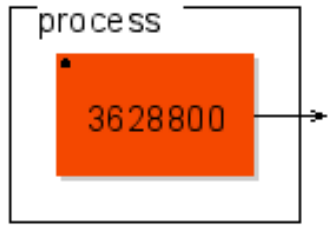
- *Functional signal processing* language, processing of *synchronous streams* of samples.
- Formal semantics means that Faust can be used as a *specification language*.
- Specifications are *executable*, sophisticated optimizations, generates competitive C++ code.
- Works with *different platforms and environments*, just recompile.

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

```
fact(0) = 1;
fact(n) = n*fact(n-1);
process = fact(10);
```

Term Rewriting Extension

- Faust signal processors are essentially **terms** in the **block diagram algebra** (BDA)
- **Term rewriting** provides us with a means to manipulate BDA terms in an **algebraic fashion** at **compile time**



JOHANNES GUTENBERG UNIVERSITÄT MAINZ

Brief Digression: Term Rewriting

$$\begin{aligned} top(push(s,x)) &\rightarrow x \\ pop(push(s,x)) &\rightarrow s \end{aligned}$$

term rewriting system

reduce

terms as “data”

$$top(pop(push(empty,1))) \rightarrow top(empty)$$

normal form

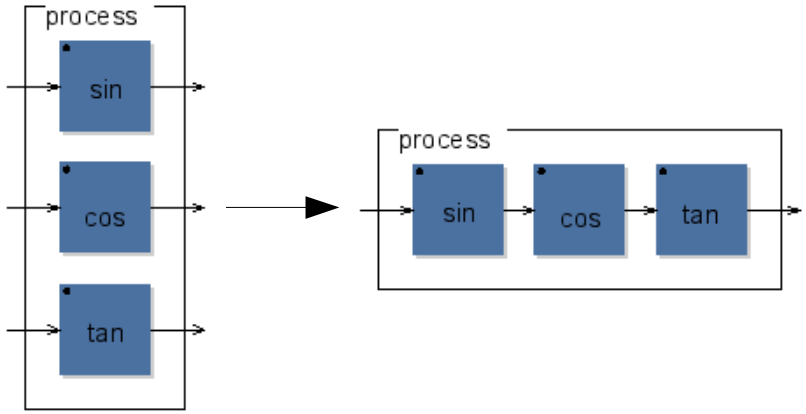
- Whitehead et al: *universal algebra*
- *Equational logic*: equality of normal forms
- O'Donnell et al: term rewriting as *programming language*
- Goguen, Mahr et al: *algebraic specification*
- Milner, Turner et al: *modern functional programming*

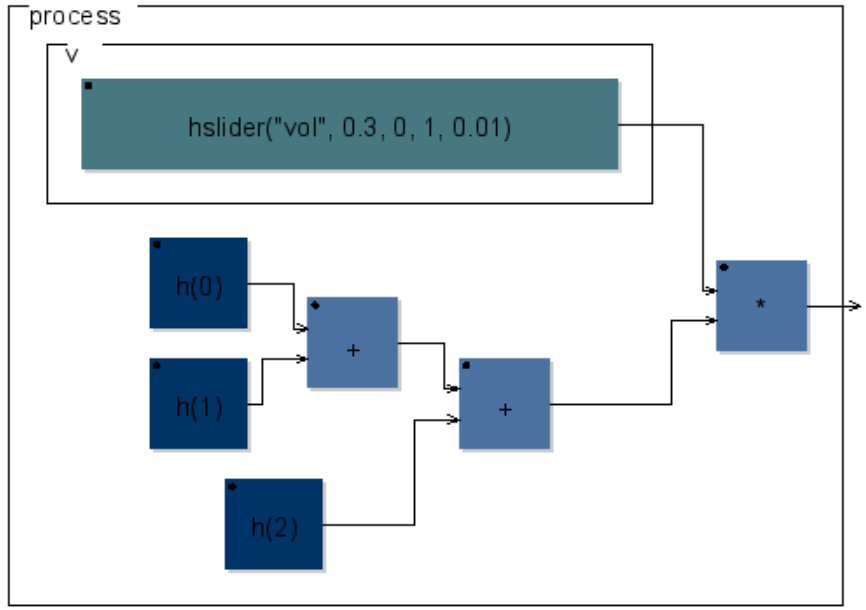
Faust Term Rewriting Extension

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

```
serial((x,y))     = serial(x) : serial(y);
serial(x)         = x;
process           = serial((sin,cos,tan));
```

BDA Term Rewriting






```
fold(1,f,x) = x(0);
fold(n,f,x) = f(fold(n-1,f,x),x(n-1));
fsum(n)     = fold(n,+);

f0 = 440; a(0) = 1; a(1) = 0.5; a(2) = 0.3;
h(i)        = a(i)*osc((i+1)*f0);
v           = hslider("vol", 0.3, 0, 1, 0.01);
process     = v*fsum(3,h);
```

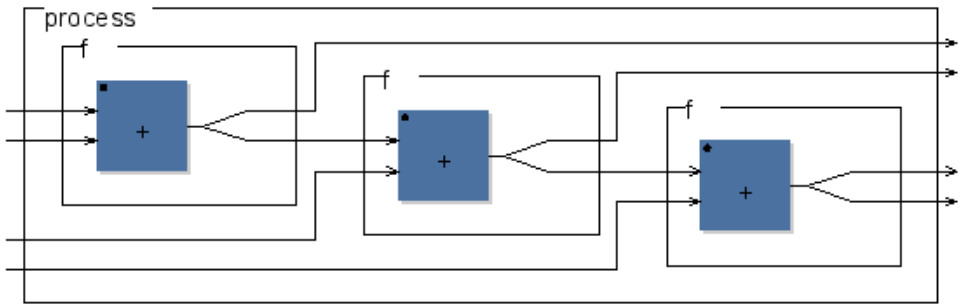
Custom BDA Ops

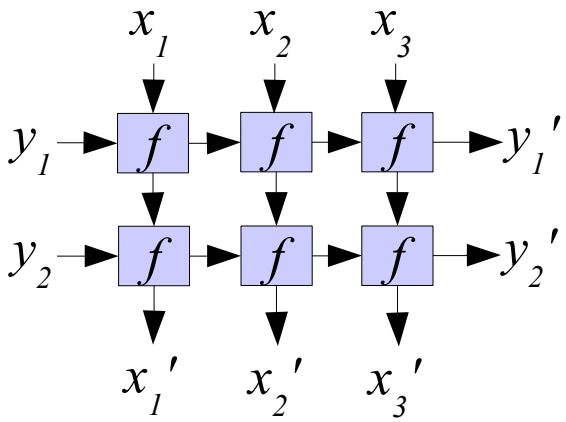
Faust Term Rewriting Extension

```
g(1,f)   = f;
g(m,f)   = (f, r(m-1)) : (_, g(m-1,f));

h(1,m,f) = g(m,f);
h(n,m,f) = (r(n+m) <: (!,r(n-1),s(m),
             (_,s(n-1),r(m) : g(m,f)))) :
            (h(n-1,m,f), _);

r(1) = _; r(n) = _,r(n-1); // route through
s(1) = !; s(n) = !,s(n-1); // skip

f        = + <: _,_; // cell function
process  = h(2,3,f);
```

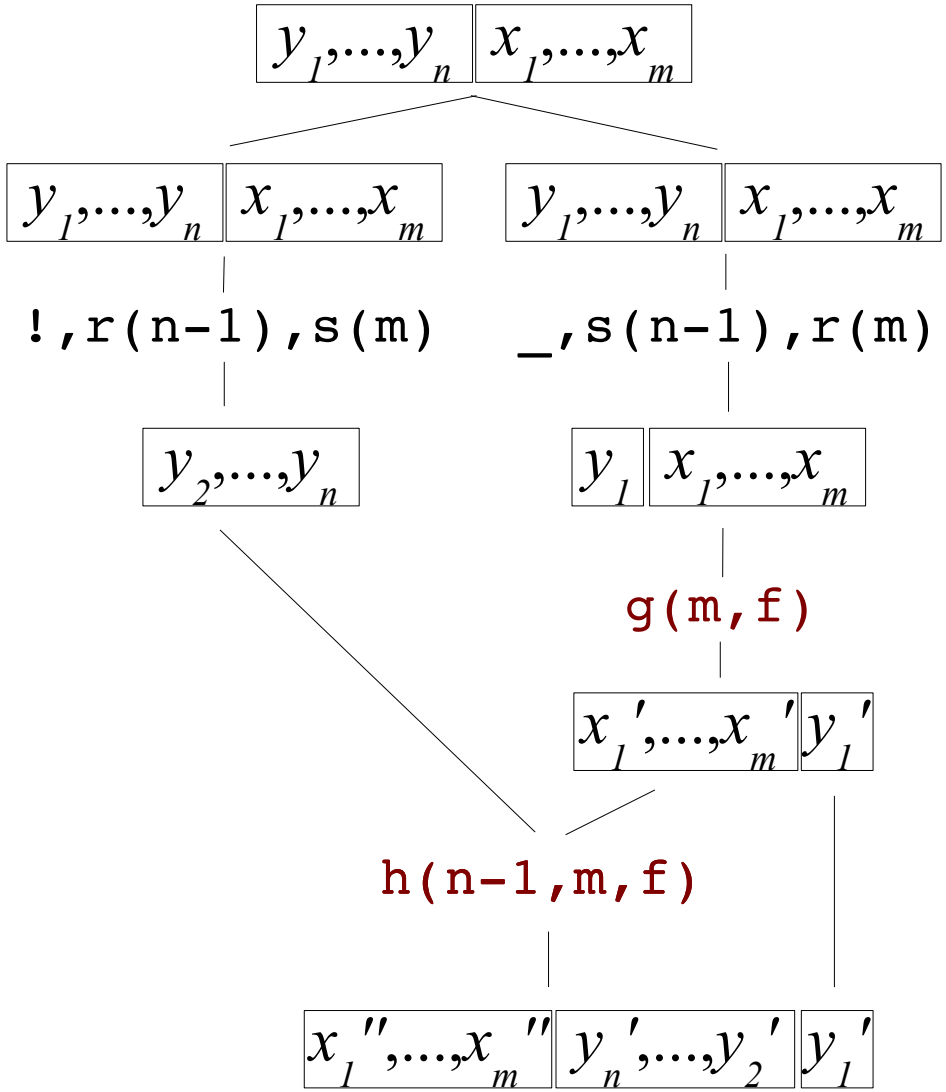
Systolic Array: parallel processing in a 2D grid

g(3,+):





Faust Term Rewriting Extension



Systolic Array: arranging the rows

```
h(n,m,f)
  = (r(n+m) <: (!,r(n-1),s(m),
     (_,s(n-1),r(m) : g(m,f)))) :
    (h(n-1,m,f), _);
```