Signal Rate Inference for Multi-Dimensional Faust

Yann Orlarey[1] Pierre Jouvelot[2]

[1]Grame, France

[2]MINES ParisTech, PSL Research University, France

IFL'16 – Leuven, August, 31, 2016

Towards Multirate Faust

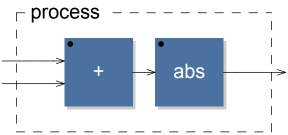
Faust (Functional Audio Stream) key features:

- Real-time (audio) signal processing DSL
- Two layers: lambda-calculus macros and *signal processors*
- High-level, functional paradigm for signals (*timed samples*)
- Efficient, multiplatform implementations

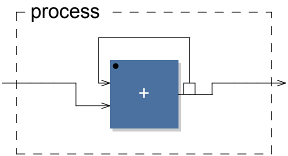
... but need for multiple rates:

- Control vs. audio rates
- Efficient spectral processing (FFT, wavelets, oversampling...)
- Multi-dimensional data structures (arrays)

Core Faust



process = + : abs ;



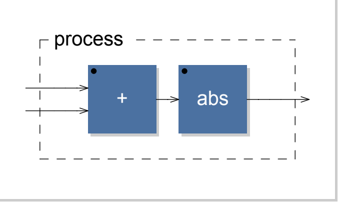
process = + ~ _ ;

Example: Organ

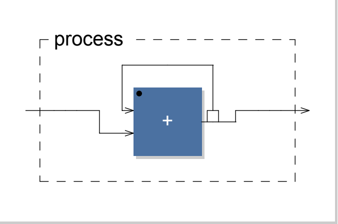

```
// organ.dsp
// http://faust.grame.fr

voice(gate, gain, freq) = envelop(gate, gain) * timbre(freq);

// Envelop
envelop(gate, gain) = gate * gain : smooth(0.9995)
   with { smooth(c) = * (1-c) : + ~ * (c); };

// Timbre
phasor(f) = f/fconstant(int fSamplingFreq, <math.h>) : (+,1.0:fmod) ~ _;
osc(f)    = phasor(f) * 6.28318530718 : sin;

timbre(freq) = osc(freq) + 0.5*osc(2.0*freq) + 0.25*osc(3.0*freq);

// Organ
process = voice(midigate, midigain, midifreq) *
          hslider("volume", 0, 0, 1, 0.01)
   with { midigate = button ("gate");
          midifreq = hslider("freq[unit:Hz]", 440, 20, 8000, 1);
          midigain = hslider("gain", 0.5, 0, 10, 0.01); };
```

Agenda

Signal Examples



$\text{abs}(\mathrm{I}_0 + \mathrm{I}_1)$



$$X_0, \text{ with } \quad \begin{aligned} D(X) &= \langle X_0@1 + \mathrm{I}_0 \rangle \\ X &= \langle X_0 \rangle \end{aligned}$$

Signal Definition

- Step-wise approximation of continuous functions...
- ... from sampled times in $\mathbb{T}_r$ ($r \in \mathbb{Q}^*$) to values in $\mathsf{V} \cup \{0_\mathsf{V}\}$
- $\mathbb{T}_r = \frac{1}{r}\mathbb{Z} = \{\frac{i}{r} \mid i \in \mathbb{Z}\}$
- Examples :

$$\begin{aligned}
\mathbb{T}_1 &= \{\ldots, -2, -1, 0, 1, 2, \ldots\} \; ; \\
\mathbb{T}_2 &= \{\ldots, -1, -0.5, 0, 0.5, 1, \ldots\} \; ; \\
\mathbb{T}_{1/3} &= \{\ldots, -6, -3, 0, 3, 6, \ldots\} \; .
\end{aligned}$$

- Negative times always yield 0_V

Syntax and Dynamic Semantics

- Signal expressions $E \in S$:

$$E ::= k \mid f \mid \mathrm{I}_n \mid X_i \mid E_1 \star E_2$$
$$E \uparrow^n \mid E \downarrow_n \mid \mathbf{v}(E, n) \mid \mathbf{s}(E) \mid E_1 \# E_2 \mid E_1[E_2]$$

- ... and $D(\langle X_0, ..., X_{n-1} \rangle) \in S^*$
- Denotational semantics:

$$\begin{aligned} \mathbb{S}_a(I_n)t &= a(I_n)t \ , \\ \mathbb{S}_a(X_i, D)t &= \mathbb{S}_a(\pi_i(D(X)))t \ . \end{aligned}$$

- Multidimensional constructs (with $s_i(t) = \mathbb{S}_a(E_i^{(r_i)})t$):

$$\begin{aligned} \mathbb{S}_a(\mathbf{v}(E_1, n))t &= [s_1(t - (n-1)/r_1), ..., s_1(t - 1/r_1), s_1(t)] \\ \mathbb{S}_a(\mathbf{s}(E_1))t &= s_1(\lfloor tr_1 \rfloor / r_1)[\mathrm{mod}(rt, n)], \text{ if } |s_1(t)| = n \ . \end{aligned}$$

Static Semantics

Y. Orlarey
P. Jouvelot

- Domain types as rates r in $\mathbb{Q}^*$
- Codomain types $\mathrm{T} ::= \mathrm{int}[l, h] \mid \mathrm{float}[l, h] \mid [n]\mathrm{T}$
- Signal types: T^r
- Typing rules, with $\Gamma(\mathrm{I}_n) = \mathrm{float}[-\infty, +\infty]^r$:

$$\frac{}{\Gamma \vdash \mathrm{I}_n : \Gamma(\mathrm{I}_n)} \qquad \frac{\Gamma \vdash (E_i, D) : \Gamma(X_i) \qquad D(X) = \langle E_0, E_1 \ldots E_{n-1} \rangle}{\Gamma \vdash (X_i, D) : \Gamma(X_i)}$$

$$\frac{\Gamma \vdash E : [n]\mathrm{T}^r}{\Gamma \vdash \mathbf{s}(E) : \mathrm{T}^{nr}} \qquad \frac{\Gamma \vdash E : \mathrm{T}^{nr}}{\Gamma \vdash \mathbf{v}(E, n) : [n]\mathrm{T}^r}$$

- Subject Reduction theorem

Rate Inference Algorithm

- Predefined typing environment Ω
- Possibly-scalable ($v \in \{0,1\}$) rate environments Δ^v, with $\Delta^v(\mathrm{I}_n) = r$ (idem for X_i)
- Set of confluent rewrite rules $(E,\Omega) \rightarrow (\Delta^v, r)$:

$$\frac{}{k \rightarrow (\perp^0, 1)} \qquad \frac{}{X_i \rightarrow (\perp[X_i \rightarrow 1]^1, 1)}$$

$$\frac{\Omega \vdash E:[n]T \quad E \rightarrow (\Delta^v, r)}{\mathbf{s}(E) \rightarrow (\Delta^v, nr)} \qquad \frac{E \rightarrow (\Delta^1, r) \quad m = \mathrm{lcm}(n,r)}{\mathbf{v}(E,n) \rightarrow ((\frac{m}{r}\Delta)^1, m/n)}$$

- Composition $\Delta_1^{v_1} + \Delta_2^{v_2}$ of rate environments for $E_1 \star E_2$:

$$\frac{r_i = \Delta_i(x) \quad m = \mathrm{lcm}(r_1, r_2) \quad (\frac{m}{r1})^{v_1}\Delta_1 \simeq (\frac{m}{r2})^{v_2}\Delta_2}{\Delta_1^{v_1} + \Delta_2^{v_2} \rightarrow ((\frac{m}{r1})^{v_1}\Delta_1 \cup (\frac{m}{r2})^{v_2}\Delta_2)^{v_1 v_2}}$$

Local Rate Inference Correctness

Theorem (Soundness (simplified))

For all integers p, if we have

- $(E, \Omega) \rightarrow (\Delta^v, r)$ *and*
- $(\Omega, p^v\Delta) \sqsubset \Gamma$,

then, $\Gamma \vdash (E, D) : \mathrm{T}^{p^v r}$.

Theorem (Integer Completeness (simplified))

If $\Gamma \vdash_{\mathbb{N}} E : \mathrm{T}^R$, *then there exist* Δ, v, r *and k such that:*

- $(E, \Omega(\Gamma)) \rightarrow (\Delta^v, r)$ *and*
- $R = rk^v$.

Global Rate Inference Algorithm (1/2)

rates(L, D) :

% Input: List L of n signal outputs E_i
%   Function D for recursive signal definitions
% Output: Typing environment Γ ($\Gamma(o_i)$ = type/rate of E_i)

% Infer types and local rates
for each E_i in L
    (Ω_i, T_i) = sample_type((E_i, D));
    $(\Delta_i^{v_i}, r_i)$ = local_rate((E_i, Ω_i));

% Compute the global sample type environment
Ω = $\bigcup_{i=0}^{n-1} \Omega_i[o_i \rightarrow \mathrm{T}_i]$;

Global Rate Inference Algorithm (2/2)


```
% Compute the global rate environment
Rs = ⋃_{i=0}^{n−1} {Δ_i[o_i → r_i]^{v_i}};
while ∃ intersecting R_1 and R_2 in Rs
    Rs = Rs ∪ {R_1 + R_2} − {R_1, R_2};
Δ = ⋃_{R∈Rs} R;

% Build the global signal type environment
Γ = [];
for each x in Dom(Δ)
    Γ = Γ[x → Ω(x)^{Δ(x)}];

% Check recursive signals
for each X in Dom(D)
    for each i from 0 to length(D(X)) − 1
        T_i^{r_i} = type/rate(Γ, (π_i(D(X)), D));
        check (T_i^{r_i} == Γ(π_i(X)));

return Γ;
end
```

Related Work

Functional languages
Music languages (Csound)
Synchronous languages:

- Synchronous Dataflow Model, SDF (Lee and Messerschmitt, 87)
- Lustre (Caspi et al, 87), Signal (Benveniste et al, 91), Lucid synchrone (Caspi et al, 07)
- "Clocks as abstract types" (Colaço and Pouzet, 03)
- Clock inference (Talpin and Shulka, 05)
- Array primitives as clock mechanisms (Jouvelot and Orlarey, 11)

Conclusion

- Faust (signal) language extended with array and sampling primitives
- Multirate inference algorithm
- Correctness theorems and proofs (for a DSL)
- Existing prototype in C++ (Grame)

Future work: relaxed type/rate constraints, explicit rates

`organ.dsp`