Compositional abstraction and analysis of digital linear filters using the Z-transform

David Monniaux

`http://www.di.ens.fr/~monniaux`

Centre National de la Recherche Scientifique

École Normale Supérieure
Département d'Informatique
45, rue d'Ulm
75230 Paris cedex 5

Introduction

The problem

Discrete-time digital filters implemented in software (general-purpose CPUs, DSPs) or in hardware.

A lot of the filtering linear: what we'll deal with

Implemented in fixed- or floating-point.

Need to provide sound assurance of absence of runtime errors in the program, including overflows.

Thus need to bound all filter outputs (and all intermediate values).

Discrete-time causal linear filters

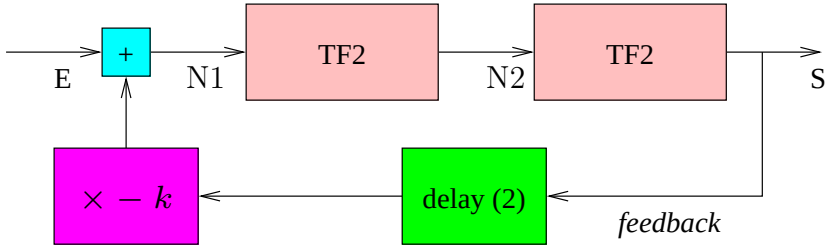
Inputs and outputs streams of data on "wires"

Complex filters made of elementary blocks connected by wires:

- delays (buffer for 1 or more clock tick(s))
- multiplication by a scalar
- addition of 2 streams

Network topology may contain feedback loops going through a delay

Example of complex filter



Each TF2 element itself a complex filter with internal filter feedback loop.

Causal, time-invariant linearity

Causal: values at clock tick $n \geq 0$ depend on those at clock ticks $\geq n$ only

(Non causal filters typically need entire buffering of the data — we do not cover them here.)

Linearity: outputs a linear function of the inputs (over the reals)

$\Rightarrow$ each output at time n a linear function of the inputs at times $\geq n$

Time invariance: this function is always the same convolution, i.e.

$$o^{(n)} = \sum_k t^{(k)} i^{(n-k)}$$

Response of the system over the reals

Transfer function

Several inputs and initial values in the "delay" operators:

$$o^{(n)} = \sum_x \sum_k r_x^{(k)} i_x^{(n-k)} + \sum_y k_y d_y^{(n)}$$

Define $O = \sum_{n=0}^{\infty} o^{(n)} z^n$ a formal power series. The equation becomes

$$O = \sum_x T_x.I_x + \sum_y k_y.D_y$$

k_y initial value of delay labeled y;
I_x series for input stream labeled x; T_x unit response for input x (Z-transform)

Bounding the transfer

We know some bound $[-B_x, B_x]$ on input I_x: $\|I_x\|_\infty \leq B_x$.

What is the $\|I \mapsto T.I\|$ operator norm w.r.t $\|\cdot\|_\infty$?
I.e. the least M such that $\|T.I\|_\infty \leq M.\|I\|_\infty$.

Answer: $\|I \mapsto T.I\| = \|T\|_1$ with $\|T\| = \sum_n |t_n|$.

Then

$$\|O\|_\infty \leq \sum_x \|T_x\|_1.\|I_x\|_\infty + \left\| \sum_y k_y.D.y \right\|_\infty$$

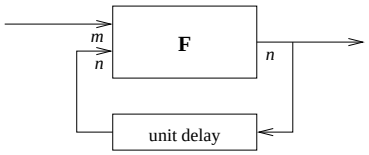
Examples

Multiplication by a scalar: $T = \alpha$,

Addition: $T_1 = 1, T_2 = 1$

Delay: by n clock ticks, $T = z^n$

Feedback loop



Filter F with two inputs I and L, output L fed back into L through unit delay (we leave out the initialization): $O = T_I.I + T_L.L = T_I.I + T_L.zO$ and thus $O = T.I$ with

$$T = (1 - zT_L)^{-1}.T_I.I$$

Rational functions

All the T power series that we construct are the developments around 0 of rational functions $P(z)/Q(z)$ (P, Q polynomials, $Q(0) = 1$) — ring $\mathbb{R}[z]_{(z)}$.

If a filter has m inputs $I_1, \ldots, I_n$, r initialization values $k_1, \ldots, k_r$, and n outputs $O_1, \ldots, O_m$, then

T is a $n \times m$ matrix over $\mathbb{R}[z]_{(z)}$; D is a $n \times r$ matrix over $\mathbb{R}[z]_{(z)}$;
K is a m-vector of $\mathbb{R}$;
I is a m-vector of $\mathbb{R}[z]_{(z)}[[z]]$ (series); O is a n-vector of $\mathbb{R}[z]_{(z)}[[z]]$ (series) and

$$O = T.I + D.K$$

Feedback loops

Take a filter F (T^F, $D^F \ldots$), feedback its n outputs into the last of its m inputs.

$O = T_1^F.I + T_2^F.zO + D.K$ and thus

$$O = (\mathrm{Id}_n - z.T_2^F)^{-1}.(T_1^F.I + D.K)$$

(this matrix is necessarily invertible)

Computations doable over $\mathbb{Q}[z]_{(z)}$!

Summary

Any of the filters can be summarized by matrices of rational functions over the rationals.

These matrices can be computed simply from the coefficients of the various elementary blocks or from the matrices of whole sub-filters (compositional design).

Example: filter $O^{(n)} = \sum_{k=0}^{d} \alpha_k I^{(n-k)} + \sum_{k=1}^{e} \beta_k O^{(n-k)}$ has transfer function

$$\frac{\alpha_0 + \alpha_1 z + \cdots + \alpha_d z^d}{1 - \beta_1 z - \cdots - \beta_e z^e}$$

$d = e = 2$: TF2 filter in our example

Bounding the transfer functions

Bounding the output

Let N_x apply $\|\cdot\|_x$ to all coordinates in a matrix or vector.
Then $N_\infty(O) \leq N_1(T).N_\infty(I) + N_\infty(D.K)$

The main problem: given a rational function $P(z)/Q(z)$, $Q(0) \neq 0$, give an upper bound on $\|P/Q\|_1$ (of its development around 0).

Idea (similar to J. Feret's scheme for 1st and 2nd order filters):

- compute explicitly the first N terms of this development, compute a bound for $\|P/Q\|_1^{<N}$
- bound the tail: $\|P/Q\|_1^{\geq N}$

Explicit development

Development of P/Q: division by ascending powers of $P(z)$ by $Q(z)$ (eqv. to running a filter $O^{(n)} = \sum_{k=0}^{d} p_k I^{(n-k)} - \sum_{k=1}^{e} q_k O^{(n-k)}$).

Problem: numerical instability using interval arithmetics on floating-point numbers.

After a while, error on the same order as the coefficients, sign of the coefficients unknown, then quick amplification.

Solution: develop until sign of the coefficients unknown. (Can go further with extended-precision arithmetic, see GNU MP's MPFR).

Tail bounding

Poles: the zeroes of $Q(z)$ are called *poles* of $P(z)/Q(z)$

The poles determine the behavior of the system.

Theorem: system stable ($\|P/Q\|_1 < \infty$) iff all poles have absolute value > 1

Intuition: (distinct poles) $[P(z)/Q(z)]^{(n)} = \sum_i \alpha_i \xi^{-n}$, ξ_i poles.

Theorem: let R be the remainder of the division by ascending powers of P/Q up to order N, then

$$\|P/Q\|_1^{\geq N} \leq \frac{\|R\|_1}{(|\xi_1| - 1) \dots (|\xi_n| - 1)}$$

Tail bounding

Implementation: Good algorithms and libraries (GSL...) for finding approximate roots $\tilde{\xi}_i$ of polynomial Q

Methods for sound bounds on the error on a root ($|\tilde{\xi}_i - \xi| \leq e(Q, \tilde{\xi}_i)$)

Improvements: Better bounds than the preceding are available (but error on tail generally much smaller than imprecisions on development), all use $|\xi_i|$.

In general: partial fraction decomposition. Easy on 2nd order.

Fixed-point or floating-point filters

Decomposition

Let $\tilde{O}$ be the output of the filter implemented in fixed- or floating- point, O the ideal real output, then we obtain for single input, output, no initialization:

$$\|\tilde{O} - O\|_\infty \leq \varepsilon_{\text{rel}}.\|I\|_\infty + \varepsilon_{\text{abs}}$$

ε_{rel} relative error, ε_{abs} absolute error.

In general: $N_\infty(\tilde{O} - O) \leq \varepsilon_{\text{rel},T}.N_\infty(I) + \varepsilon_{\text{rel},D}.N_\infty(K) + \varepsilon_{\text{abs}}$ with $\varepsilon_{\text{rel},T}$, $\varepsilon_{\text{rel},D}$ matrices in $\mathbb{R}_+$, ε_{abs} vector in $\mathbb{R}_+$

Error in elementary blocks

$x \oplus y$ in fixed- or floating-point $= r(x+y)$, $x \otimes y$ in fixed- or floating-point $= r(x.y)$; r rounding function such that $|r(x) - x| \leq \varepsilon_{\text{rel}}.|x| + \varepsilon_{\text{abs}}$

Fixed-point: $\varepsilon_{\text{rel}} = 0$, $\varepsilon_{\text{abs}} = u/2$ (round-to-nearest), $\varepsilon_a bs = u$ (other modes) with u the least representable positive number

Floating-point: ε_{rel} error at the n-th binary position after the point

ε_{abs} is the denormalization error: ex, if u is the least representable positive number, then $0.25 \otimes u = 0$ (ε_{abs} very small, but included for soundness)

$|r(x) - x| \leq \max(\varepsilon_{\text{rel}}.|x|, \varepsilon_{\text{abs}})$ overapproximated by affine form

Error propagation

Simple blocks: With the above: easy for addition and multiplication by scalar, no error on delays

Feedback loop: around filter F: somewhat complex computation ending up with $\varepsilon_{\text{rel}} = A.\varepsilon_{\text{rel}} + B$ with A with very small coefficients $\Rightarrow$ resolution by fixpoint iteration

Simplification: with a P/Q filter (P, Q with rational coefficients, can be large), replace by $\tilde{P}/\tilde{Q}$ (approximations for P and Q) with some larger ε_{rel}

To summarize

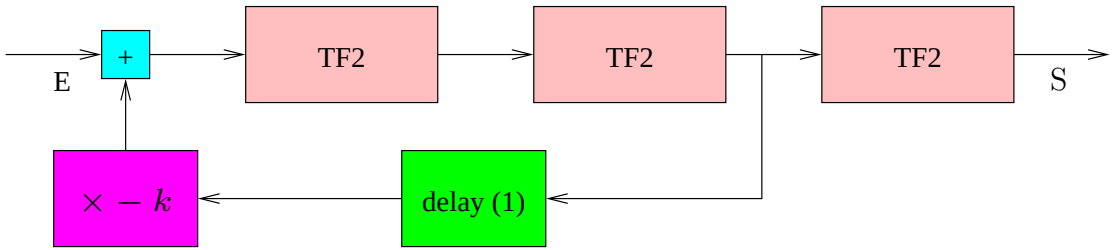
Any causal linear filter F with finite memory implemented over fixed-point or floating-point (or a mix thereof) can be summarized into:

- matrices T^F and D^F over $\mathbb{Q}[z]_{(z)}$ expressing the ideal, real input-output relationship by Z-transform: T wrt input streams, D wrt values initially in the delay memories
- matrices $\varepsilon^F_{\mathrm{rel},T}$ and $\varepsilon^F_{\mathrm{abs},D}$ over $\mathbb{R}_+$ expressing the relative errors wrt input streams and delay memories
- vector $\varepsilon^F_{\mathrm{abs}}$ of absolute error

This is compositional: computation for complex filters from the analysis results of sub-filters.

Conclusion and prospects

Implementation results



Defined compositionally in the analyzer

Computation in 0.1s (could be optimized), P/Q P of 6th degree, Q of 7th degree

$\varepsilon_{\text{rel}} \leq 4.781.10^{-13}$, $\|O\|_\infty \leq 2.0576\|I\|_\infty$.

Reconstruction of filters

From C or similar program (SSA form)

```
while (1) {
  ...
  filter
}
```

Read all lines in filter, obtain $v = e$ equations from $v := e$ assignment; variables v in e not already initialized in loop become $z.v$

In case of nonlinearity: use approximation to remove the linear part and get large $\varepsilon_{\mathrm{rel}}$.

Solve the resulting system.

Conclusions

Compositional abstract semantics for fixed- and floating-point digital linear filters with fixed memory of arbitrary complexity.

Sound bounds on the output obtained as affine function of bounds on the inputs.

Simple implementation already gives good results.

Good for analysis of data-flow languages for automatic systems (graphical Scade etc.). Extension for imperative languages.