NSAD'05

Numerical Abstract Domains for Digital Filters

Jérôme Feret
École Normale Supérieure
http://www.di.ens.fr/~feret

January, 2005

Overview

1. Introduction
2. Case study
3. Generic framework
4. Simplified filters
5. Expanded filters
6. Conclusion

Context

- proving the absence of run time error in critical embedded software.

  Filter behavior is implemented at the software level, using hardware floating point numbers.

  $\Longrightarrow$ Full certification requires special care about these filters.

Issues

- Control flow detection: to locate filter resets and filter iterations.

- Invariant inference:

  We seek precise bounds on the output, using information inferred about the input. (Linear invariants do not yield accurate bounds).

- Floating-point arithmetics:

  (in the concrete semantics and when implementing the abstract domain).

Overview

1. Introduction
2. Case study
3. Generic framework
4. Simplified filters
5. Expanded filters
6. Conclusion

The second order filter (simplified)

$$V \in \mathbb{R};$$

$$E := 0;\quad S_0 := 0;\quad S_1 := 0;\quad S_2 := 0;$$

$\text{while}\ (V \geq 0)\ \{$

$\quad V \in \mathbb{R};\ T \in \mathbb{R};$

$\quad E \in [-1; 1];$

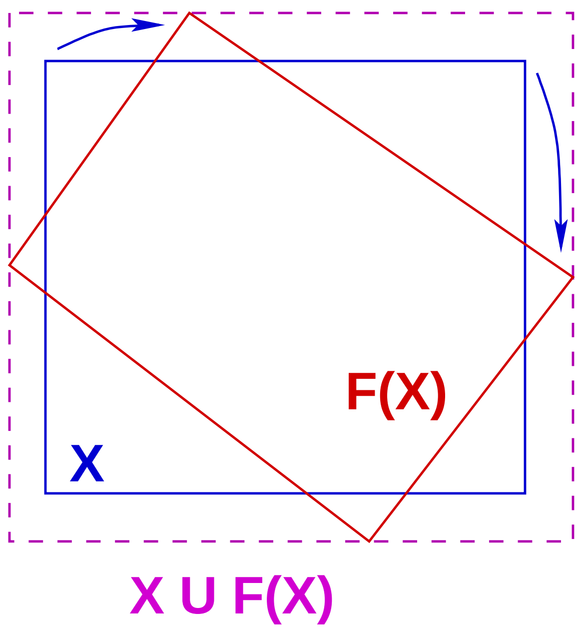
$\quad \text{if}\ (T \geq 0)\ \{S_0 := E; S_1 := E;\}$

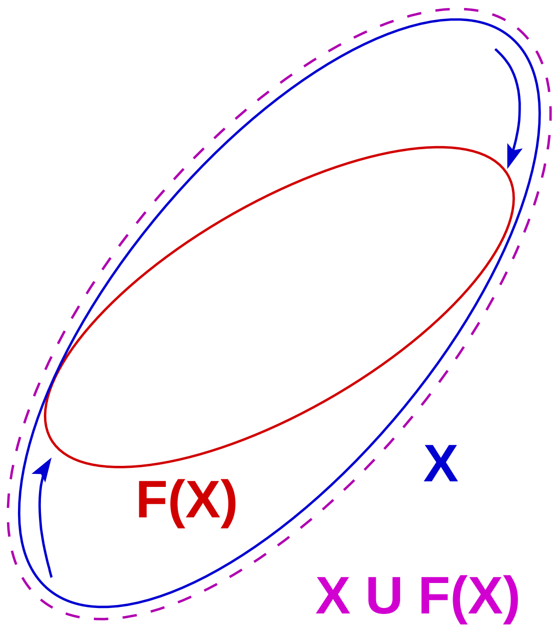
$\quad \text{else}\ \{S_0 := 1.5 \times S_1 - 0.7 \times S_2 + E;\}$

$\quad S_2 := S_1;\ S_1 := S_0;$

$\}$

Linear versus quadratic invariants





Ellipsoidal constraints

Theorem 1 (second order filter (simplified))
Let a, b, $K \geq 0$, $m \geq 0$, X, Y, Z *be real numbers such that:*

1. $a^2 + 4b < 0$,
2. $X^2 - aXY - bY^2 \leq K$,
3. $aX + bY - m \leq Z \leq aX + bY + m$.

We have:

1. $Z^2 - aZX - bX^2 \leq \left(\sqrt{-bK} + m\right)^2$;
2. $$\begin{cases} \sqrt{-b} < 1 \\ K \geq \left(\frac{m}{1-\sqrt{-b}}\right)^2 \end{cases} \implies Z^2 - aZX - bX^2 \leq K.$$

Overview

1. Introduction
2. Case study
3. Generic framework
4. Simplified filters
5. Expanded filters
6. Conclusion

Filter family

A filter class is given by:

- the number p of outputs and the number q of inputs involved in the computation of the next output;
- a (generic/symbolic) description of F with parameters;
- some conditions over these parameters

In the case of a second order filter using the last three inputs:

- $p = 2, \ \ q = 3;$
- $F(S_{n+1}, S_n, E_{n+2}, E_{n+1}, E_n) = a.S_{n+1} + b.S_n + c.E_{n+2} + d.E_{n+1} + e.E_n;$
- $a^2 + 4b < 0.$

Filter domain

A filter constraint is a couple in $\mathcal{T}_{\mathcal{B}} \times \mathcal{B}$ where:

- $\mathcal{T}_{\mathcal{B}} \in \wp_{\text{finite}}(\mathcal{V}^{m} \times \mathbb{R}^{n})$ with:
  - m, the number of variables that are involved in the computation of the next output. m depends on the abstraction;
  - n, the number of filter parameters;
- $\mathcal{B}$ is an abstract domain encoding some “ranges”.

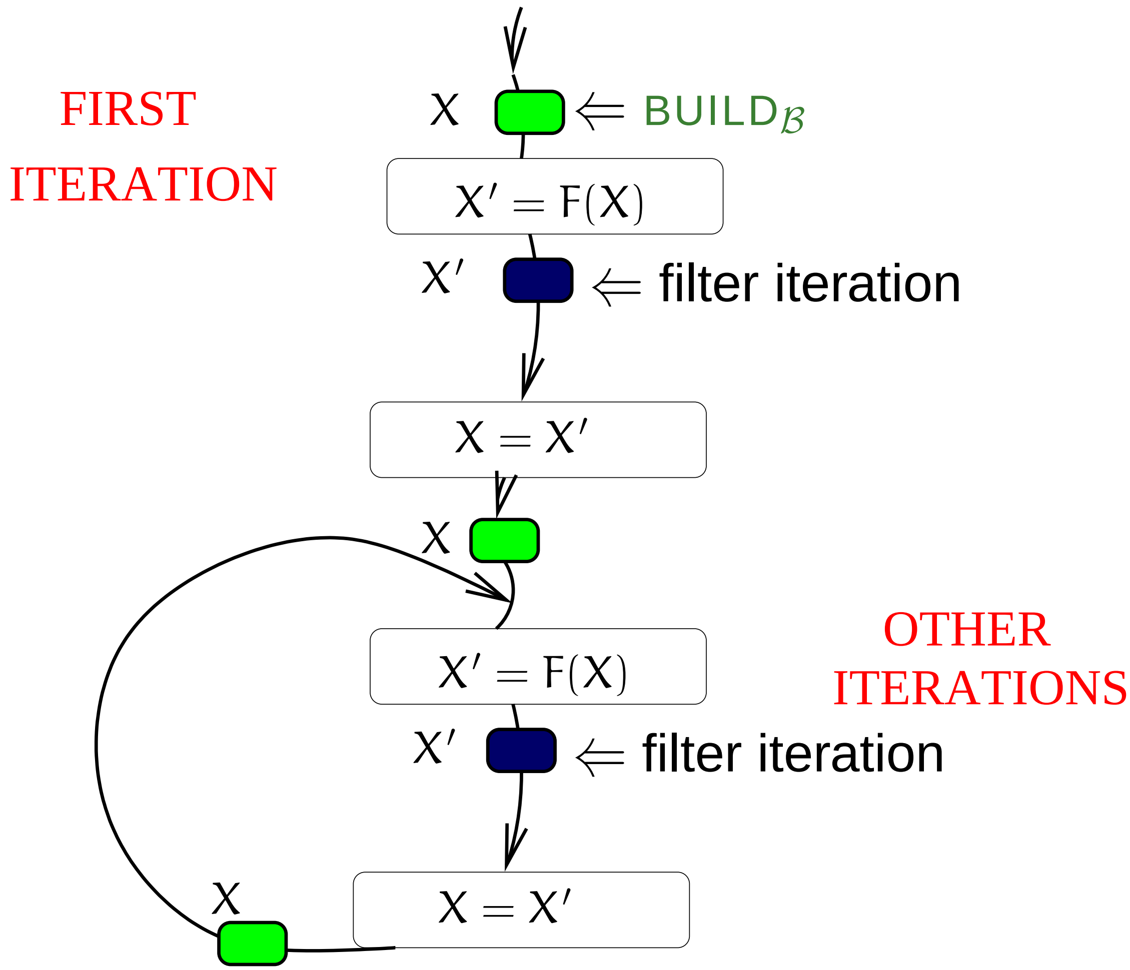
A constraint (t, d) is related to $\wp(\mathcal{V} \to \mathbb{R})$, by a concretization function:

$$\gamma_{\mathcal{B}} : \mathcal{T}_{\mathcal{B}} \times \mathcal{B} \to \wp(\mathcal{V} \to \mathbb{R}).$$

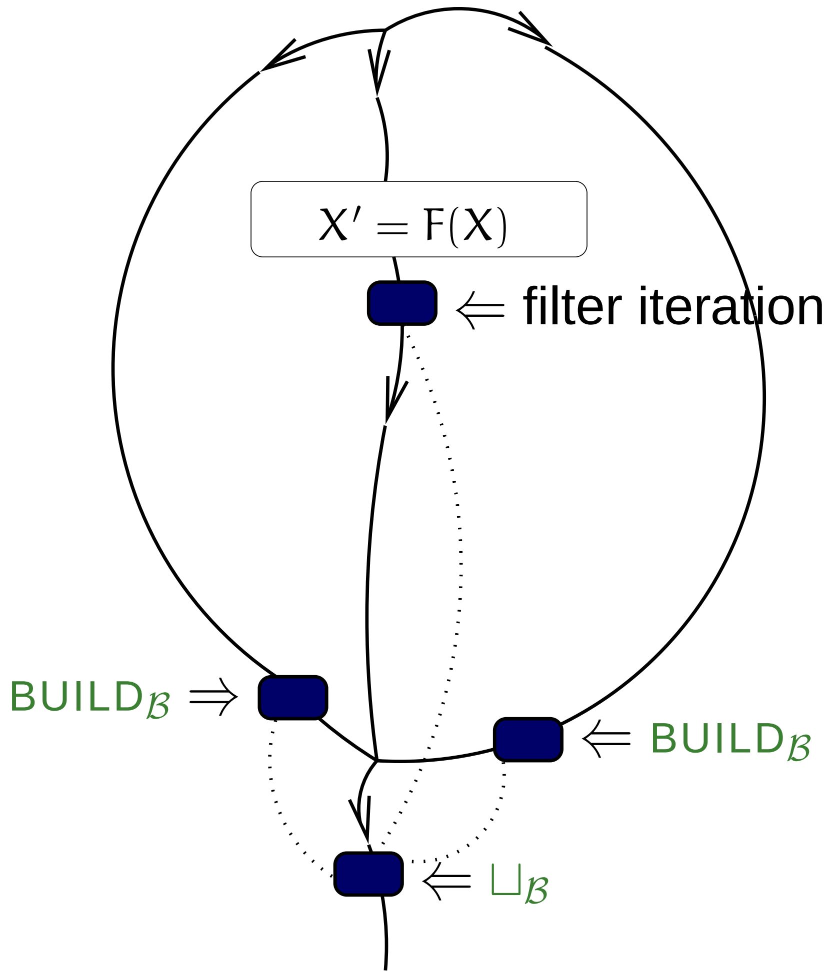
An approximation of second order filter may consist in relating:

- the last two outputs and the first two coefficients of the filter (a and b)
- to the ‘ratio’ of an ellipsoid.

Assignment


FIRST
ITERATION
X
⇐ $\text{BUILD}_{\mathcal{B}}$
$X' = F(X)$
X'
⇐ filter iteration
$X = X'$
X
OTHER
ITERATIONS
$X' = F(X)$
X'
⇐ filter iteration
X
$X = X'$

Merging computation paths



Overview

1. Introduction
2. Case study
3. Generic framework
4. Simplified filters
5. Expanded filters
6. Conclusion

Simplified first order filter

Theorem 2 (Including rounding errors)
Let a, $\varepsilon_a \geq 0$, $D \geq 0$, $m \geq 0$, X *and* Z *be real numbers such that:*

1. $|X| \leq D$;
2. $aX - (m + \varepsilon_a|X|) \leq Z \leq aX + (m + \varepsilon_a|X|)$.

We have:

- $|Z| \leq (|a| + \varepsilon_a)D + m$;
- $\left[|a| + \varepsilon_a < 1 \text{ and } D \geq \frac{m}{1-(|a|+\varepsilon_a)}\right] \implies |Z| \leq D$. $\square$

Simplified second order filter

Theorem 3 (Including rounding errors)
Let a, b, $\varepsilon_a \geq 0$, $\varepsilon_b \geq 0$, $K \geq 0$, $m \geq 0$, X, Y, Z *be real numbers, such that:*

1. $a^2 + 4b < 0$,
2. $X^2 - aXY - bY^2 \leq K$,
3. $aX + bY - (m + \varepsilon_a|X| + \varepsilon_b|Y|) \leq Z \leq aX + bY + (m + \varepsilon_a|X| + \varepsilon_b|Y|)$.

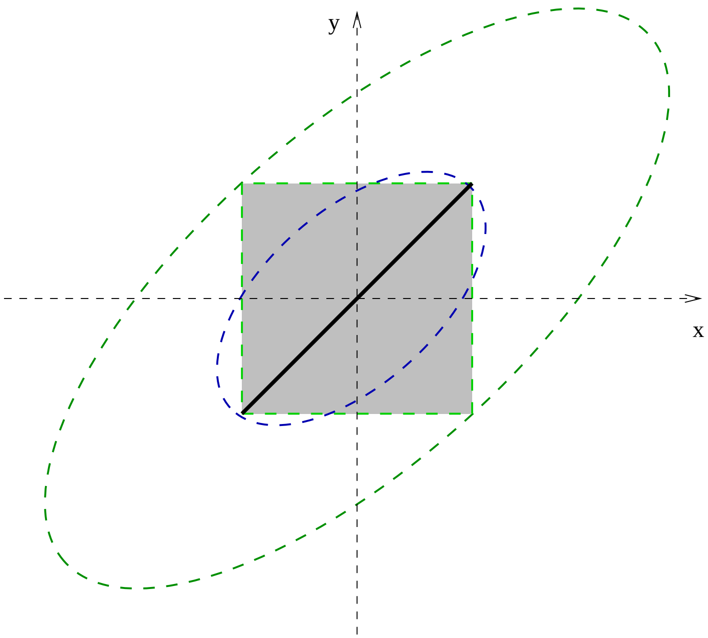
We have

1. $Z^2 - aZX - bX^2 \leq \left((\sqrt{-b} + \delta)\sqrt{K} + m\right)^2$;
2. $\begin{cases} \sqrt{-b} + \delta < 1 \\ K \geq \left(\frac{m}{1-\sqrt{-b}-\delta}\right)^2 \end{cases} \implies Z^2 - aZX - bX^2 \leq K$,

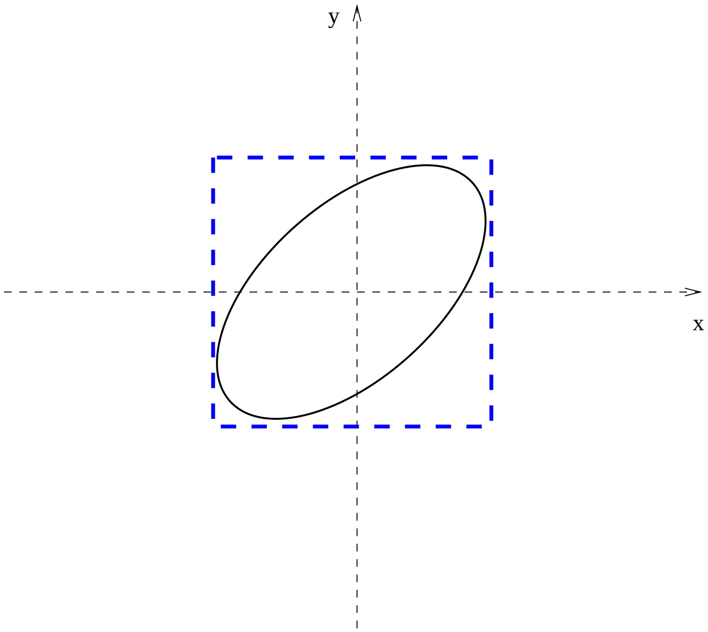
where $\delta = 2\frac{\varepsilon_b + \varepsilon_a\sqrt{-b}}{\sqrt{-(a^2+4b)}}$.

$\square$

Reduced product



Initial conditions



Output refinement

Higher order simplified filters

A simplified filter of class (k, l) is defined as a sequence:

$$S_{n+p} = a_1.S_n + ... + a_p.S_{n+p-1} + E_{n+p},$$

where the polynomial $P = X^p - a_p.X^{p-1} - ... - a_1.X^0$ has no multiple roots (in $\mathbb{C}$) and can be factored into the product of k second order irreducible polynomials $X^2 - \alpha_i.X - \beta_i$ and l first order polynomials $X - \delta_j$.
Then, there exists sequences $(x_n^i)_{n\in\mathbb{N}}$ and $(y_n^j)_{n\in\mathbb{N}}$ such that:

$$\begin{cases} S_n = (\sum\limits_{1\leq i\leq k} x_n^i) + (\sum\limits_{1\leq j\leq l} y_n^j) \\ x_{n+2}^i = \alpha_i.x_{n+1}^i + \beta_i.x_n^i + F^i(E_{n+2}, E_{n+1}) \\ y_{n+1}^j = \delta_j.y_n^j + G^j(E_{n+1}). \end{cases}$$

The initial outputs (x_0^i, x_1^i, y_0^j) and filter inputs F^i, G^j are given by solving symbolic linear systems, they only depend on the roots of P.

Higher order simplified filters

Soundness of the factoring algorithm into irreducible polynomials is not required.

Whenever we meet a higher order filter assignment τ,

1. we compute the characteristic polynomial P,
2. we compute a potentially unsound factoring P' of P,
3. we expand P',
4. we consider the filter assignment τ' such that the characteristic polynomial of τ' is P',
5. we bound the difference between τ and τ'
   (by using symbolic computation),
6. we integrate this bound into the input stream.

Overview

1. Introduction
2. Case study
3. Generic framework
4. Simplified filters
5. Expanded filters
6. Conclusion

Other filters

We have:

$$\begin{cases} S_k = i_k, 0 \leq k < p \\ S_{n+p} = \overline{F}(S_n, ..., S_{n+p-1}) \mp \overline{G}(E_{n+p+1-q}, ..., E_{n+p}) \end{cases}$$

Having bounds:

- on the input sequence (E_n),
- and on the initial outputs $(i_k)_{0 \leq k < p}$;

we want to infer a bound on the output sequence (S_n).

Splitting S_n

We split the output sequence $S_n = R_n + \varepsilon_n$ into

- the contribution of the errors (ε_n);

$$\begin{cases} \varepsilon_k = 0, 0 \leq k < p; \\ \varepsilon_{n+p} = F(\varepsilon_n, \ldots, \varepsilon_{n+p-1}) + \mathit{err}_{n+p} \end{cases}$$

  we can use the simplified filter domain to limit (ε_n).

- the ideal sequence (R_n) (in the real field);

$$\begin{cases} R_k = i_k,\ 0 \leq k < p \\ R_{n+p} = F(R_n, \ldots, R_{n+p-1}) + G(E_{n+p+1-q}, \ldots, E_{n+p}) \end{cases}$$

Limiting R_n

To refine the output, we need to limit the sequence R_n:

1. We isolate the contribution of the N last inputs:

$$R_n = \mathit{last}_n^N(E_n, \ldots, E_{n+1-N}) + \mathit{res}_n^N.$$

2. Since the filter is linear, we have, for $n > N + p$:
   - $\mathit{last}_n^N = \mathit{last}_{N+p}^N$;
   - res_n^N can be limited by using the corresponding simplified filter domain.

Overview

1. Introduction
2. Case study
3. Generic framework
4. Simplified filters
5. Expanded filters
6. Conclusion

Benchmarks

We analyze three programs in the same family on a AMD Opteron 248, 8 Gb of RAM (analyses use only 2 Gb of RAM).

lines of C	70,000			216,000			379,000		
global variables	13,400			7,500			9,000		
iterations	72	41	**37**	161	75	**53**	151	187	**74**
time/iteration	52s	1mn18s	**1mn16s**	3mn07s	5mn08s	**4mn40s**	4mn35s	9mn25s	**8mn17s**
analysis time	1h02mn	53mn	**47mn**	8h23mn	6h25mn	**4h08mn**	11h34mn	30h26mn	**10h14mn**
false alarms	574	3	**0**	207	0	**0**	790	0	**0**

1. without filter domains;
2. with simplified filter domains;
3. with expanded filter domains.

Conclusion

- a highly generic framework to analyze programs with digital filtering:

  a technical knowledge of used filters allows the design of the adequate abstract domain;

- the case of linear filters is fully handled:

  We need to solve a symbolic linear system for each filter family. We need an unsound polynomial reduction algorithm for each filter instance.

- filter detection is left as a parameter:

  - term rebuilding can be used [MinéPhD];

This framework has been used and was necessary in the full certification of the absence of runtime error in industrial critical embedded software.

http://www.astree.ens.fr