Antique SEMINAR

Static Analysis of Digital Filters
ESOP 2004, NSAD 2005

Jérôme Feret
Laboratoire d’Informatique de l’École Normale Supérieure
INRIA, ÉNS, CNRS

http://www.di.ens.fr/~feret

May, 17th 2024.
Overview

1. Introduction
2. Case studies
3. Concrete semantics
4. Generic approximation
5. Filter domains
6. Basic simplified filters
7. Higher order simplified filters
8. Bounded expansion
9. Conclusion
Context

We want to **prove run time error absence**, in **critical embedded software**. Filter behaviour is implemented at the software level, using hardware floating point numbers.

**Full certification** requires special care about these filters.
Issues

• **Detection**: to locate filter resets and filter iterations.

• **Invariant inference**: we are not interested in functional properties. We seek precise bounds on the output, using information inferred about the input. *(Linear invariants do not yield accurate bounds).*

• To take into account floating-point rounding:
  – in the semantics,
  – when implementing the abstract domain.
Overview

1. Introduction
2. Case studies
3. Concrete semantics
4. Generic approximation
5. Filter domains
6. Basic simplified filters
7. Higher order simplified filters
8. Bounded expansion
9. Conclusion
The high band-pass filter

\[ V \in \mathbb{R}; \]
\[ E_1 := 0; \quad S := 0; \]
\[ \text{while } (V \geq 0) \{ \]
\[ \quad V \in \mathbb{R}; \quad T \in \mathbb{R}; \]
\[ \quad E_0 \in [-1;1]; \]
\[ \quad \text{if } (T \geq 0) \{ S := 0 \} \]
\[ \quad \text{else } \{ S := 0.999 \times S + E_0 - E_1 \} \]
\[ \quad E_1 := E_0; \]
\[ \} \]
The analyzer infers the following sound counterpart $\mathbb{F}^\#$:

$$\mathbb{F}^\#(X) = \{0.999s + e_0 + e_1 \mid s \in X, \ e_0, e_1 \in [-1; 1]\}$$

to the loop body.
Abstract iteration

1. The analyzer starts iterating $F^\#:$
   \[
   F^\#(\{0\}) = [-2; 2],
   F^\#([-2; 2]) = [-3.998; 3.998],
   \ldots;
   \]

2. then it widens the iterates:
   \[
   F^\#([-10; 10]) \not\subseteq [-10; 10],
   F^\#([-100; 100]) \not\subseteq [-100; 100],
   \ldots;
   \]

3. until it discovers a stable threshold:
   \[
   F^\#([-10000; 10000]) = [-9992; 9992];
   \]

4. finally, it keeps iterating to refine the solution:
   \[
   F^\#([-9992; 9992]) = [-9984.008; 9984.008].
   \]
Driving the analysis

Theorem 1 (High band-pass filter (history-insensitive))
Let $D \geq 0$, $m \geq 0$, $a$, $X$ and $Z$ be real numbers such that:

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

then we have:

1. $|Z| \leq |a|D + m$;
2. $|a| < 1$ and $D \geq \frac{m}{1 - |a|} \implies |Z| \leq D$.

Theorem 1 implies that 2000 can be used as a widening threshold.
History sensitive approximation

Theorem 2 (High band-pass filter (history-sensitive version))
Let $\alpha \in \left[\frac{1}{2}; 1\right]$, $i$ and $m > 0$ be real numbers.
Let $E_n$ be a real number sequence, such that $\forall k \in \mathbb{N}, \ E_k \in [-m; m]$.
Let $S_n$ be the following sequence:

$$
\begin{align*}
S_0 &= i \\
S_{n+1} &= \alpha S_n + E_{n+1} - E_n.
\end{align*}
$$

We have:

1. $S_n = \alpha^n i + E_n - \alpha^n E_0 + \sum_{l=1}^{n-1} (\alpha - 1)\alpha^{l-1}E_{n-l}$
2. $|S_n| \leq |\alpha|^n|i| + (1 + |\alpha|^n + |1 - \alpha^{n-1}|)m$;
3. $|S_n| \leq 2m + |i|$.

Theorem 2 implies that 2 is a sound bound on $|S|$. □
The second order filter

\[
V \in \mathbb{R};
\]
\[E_1 := 0; \quad E_2 := 0; \quad S_0 := 0; \quad S_1 := 0; \quad S_2 := 0;\]
while \((V \geq 0)\) {
  \[
  V \in \mathbb{R}; \quad T \in \mathbb{R};
  \]
  \[
  E_0 \in [-1; 1];
  \]
  if \((T \geq 0)\) \{ \[
    S_0 := E_0; \quad S_1 := E_0; \quad E_1 := E_0
  \} \]
  else \{ \[
    S_0 := 1.5 \times S_1 - 0.7 \times S_2 + 0.5 \times E_0 - 0.7 \times E_1 + 0.4 \times E_2
  \} \]
  \[
  E_2 := E_1; \quad E_1 := E_0;
  \]
  \[
  S_2 := S_1; \quad S_1 := S_0 \]
}
Quadratic constraints

Theorem 3 (second order filter (history insensitive))
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. $Z^2 - aZX - bX^2 \leq (\sqrt{-bK} + m)^2$;
2. $\begin{cases} \sqrt{-b} < 1 \\ K \geq \left(\frac{m}{1-\sqrt{-b}}\right)^2 \Rightarrow Z^2 - aZX - bX^2 \leq K. \end{cases}$
Proof

We define \( Q(X, Y) \overset{\Delta}{=} X^2 - aXY - bY^2 \) and \( Z \overset{\Delta}{=} aX + bY + e \).
We have:

\[
Q(Z, X) = (aX + bY + e)^2 - a(aX + bY + e)X - bX^2
\]
\[
Q(Z, X) = -b(X^2 - aXY - bY^2) + e(aX + 2bY + e)
\]
\[
Q(Z, X) = -bQ(X, Y) + e(aX + 2bY + e)
\]
\[
Q(Z, X) \leq -bQ(X, Y) + m|aX + 2bY| + m^2 \quad \text{since } |e| \leq m
\]

\[
(aX + 2bY)^2 = -4b\left(\frac{a^2}{4b}X^2 - aXY - bY^2\right)
\]
\[
(aX + 2bY)^2 \leq -4bQ(X, Y) \quad \text{since } a^2 + 4b < 0
\]
\[
|aX + 2bY| \leq 2\sqrt{-bQ(X, Y)}
\]

\[
Q(Z, X) \leq \left(\sqrt{-bQ(X, Y)} + m\right)^2
\]
Linear versus quadratic invariants

[Diagram showing the comparison between linear and quadratic invariants with labeled axes and regions labeled as $X$, $F(X)$, and $X \cup F(X)$.]
Second order filter approximation

1. without relational domain, we cannot limit $|S_2|$

2. with quadratic constraints (history insensitive abstraction), we can infer that $|S_2| < 22.111$

3. by formally expanding the output as a sum of all previous inputs, we can prove that $|S_2| < 1.41824$
Overview

1. Introduction
2. Case studies
3. Concrete semantics
4. Generic approximation
5. Filter domains
6. Basic simplified filters
7. Higher order simplified filters
8. Bounded expansion
9. Conclusion
Let $\mathcal{V}$ be a finite set of variables.
Let $\mathcal{I}$ be the set of real intervals (including $\mathbb{R}$).
Expressions $\mathcal{E}$ are affine forms of variables $\mathcal{V}$ with real interval coefficients:

$$ E ::= I + \sum_{j \in J} I_j V_j $$

Programs are given by the following grammar:

$$ P ::= \text{skip} $$
$$ P;P $$
$$ V ::= E $$
$$ \text{if} \ (V \geq 0) \ \{ P \} \ \text{else} \ \{ P \} $$
$$ \text{while} \ (V \geq 0) \ \{ P \} $$
Semantics

We define the semantics of a program $P$:

$$\llbracket P \rrbracket : (\mathcal{V} \to \mathbb{R}) \to \wp(\mathcal{V} \to \mathbb{R})$$

by induction over the syntax of $P$:

$$\llbracket \text{skip} \rrbracket (\rho) = \{ \rho \},$$

$$\llbracket P_1; P_2 \rrbracket (\rho) = \{ \rho'' \mid \exists \rho' \in \llbracket P_1 \rrbracket (\rho), \rho'' \in \llbracket P_2 \rrbracket (\rho') \},$$

$$\llbracket V := I + \sum_{j \in J} I_j.V_j \rrbracket (\rho) = \{ \rho[V \mapsto i + \sum_{j \in J} i_j.\rho(V_j)] \mid i \in I, \forall j \in J, i_j \in I_j \},$$

$$\llbracket \text{if } (V \geq 0) \{ P_1 \} \text{ else } \{ P_2 \} \rrbracket (\rho) = \begin{cases} \llbracket P_1 \rrbracket (\rho) & \text{if } \rho(V) \geq 0 \\ \llbracket P_2 \rrbracket (\rho) & \text{otherwise,} \end{cases}$$

$$\llbracket \text{while } (V \geq 0) \{ P \} \rrbracket (\rho) = \{ \rho' \in \text{Inv} \mid \rho'(V) < 0 \}$$

where $\text{Inv} = \text{lfp} (X \mapsto \{ \rho \} \cup \{ \rho'' \mid \exists \rho' \in X, \rho'(V) \geq 0 \text{ and } \rho'' \in \llbracket P \rrbracket (\rho') \}).$
Abstract domain

An abstract domain ENV♯ is a set of environment properties.
A concretization map γ relates each property to the set of its solutions:

$$\gamma : \text{ENV}^♯ \rightarrow \wp(\mathcal{V} \rightarrow \mathbb{R}).$$

Some primitives simulate concrete computation steps in the abstract:

- an abstract control path merge ∪;
- an abstract guard GUARD and an abstract assignment ASSIGN;
- an abstract least fixpoint lfp♯ operator, which maps sound counterpart $f^♯$ to monotonic function $f$, to an abstraction of the least fixpoint of $f$.

lfp♯ is defined using extrapolation operators (⊥, ▽, △).

Soundness follows from the monotony of the concrete semantics.
Abstract semantics

\[ [\text{skip}]^\#(a) = a \]

\[ [P_1; P_2]^\#(\rho^\#) = [P_2]^\#([P_1]^\#(\rho^\#)) \]

\[ [V := E]^\#(a) = \text{ASSIGN}(V, E, a) \]

\[ [\text{if } (V \geq 0) \{P_1\} \text{ else } \{P_2\}]^\#(a) = a_1 \sqcup a_2, \]
\[ \text{with} \quad \begin{cases} a_1 = [P_1]^\#(\text{GUARD}(V, [0; +\infty[, a))) \\ a_2 = [P_2]^\#(\text{GUARD}(V, ]-\infty; 0[, a)) \end{cases} \]

\[ [\text{while } (V \geq 0) \{P\}]^\#(a) = \text{GUARD}(V, ]-\infty; 0[, \text{Inv}^\#) \]
\[ \text{where } \text{Inv}^\# = \text{lfp}^\#(X \mapsto a \sqcup [P]^\#(\text{GUARD}(V, [0; +\infty[, X))) \]
We prove by induction over the syntax:

**Theorem 4 (Soundness)** For any program $P$, environment $\rho$, abstract element $a$, we have:

$$\rho \in \gamma(a) \implies \llbracket P \rrbracket(\rho) \subseteq \gamma(\llbracket P \rrbracket^{\sharp}(a)) .$$
Overview

1. Introduction
2. Case studies
3. Concrete semantics
4. Generic approximation
5. Filter domains
6. Basic simplified filters
7. Higher order simplified filters
8. Bounded expansion
9. 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 the second order filter:

- $p = 2, \; q = 3$;
- $F(V, W, X, Y, Z) = a \times V + b \times W + c \times X + d \times Y + e \times Z$;
- $a^2 + 4b < 0$. 
Filter domain

A filter constraint is a couple in $\mathcal{T} \times \mathcal{B}$ where:
- $\mathcal{T} \in \wp_{\text{finite}}(V^m \times \mathbb{R}^n)$ 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 a set of environments:

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

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 ‘radius’ of an ellipsis.
\[ \vec{Y} = F(\vec{X}) \]
\[ \vec{X} = \vec{Y} \]
Iterations

$\vec{X} \equiv \text{BUILD}$

$\vec{Y} = F(\vec{X})$

$\vec{X} = \vec{Y}$
\[ \vec{Y} = F(\vec{X}) \]
\[ \vec{X} = \vec{Y} \]
$\vec{X}$

$\vec{Y} = F(\vec{X})$

$\vec{X} = \vec{Y}$

**Iterations**
\[ \vec{X} = \vec{Y} = F(\vec{X}) \]

\[ \vec{X} = \vec{Y} \]
Iterations

\[
\vec{X} = \vec{Y} = F(\vec{X})
\]

\[
\vec{X} = \vec{Y}
\]
\[ \vec{Y} = F(\vec{X}) \]
\[ \vec{X} = \vec{Y} \]
\[ \vec{X} \rightarrow \vec{Y} = F(\vec{X}) \rightarrow \vec{X} = \vec{Y} \]
Iterations

\[ \vec{X} = \vec{Y} \]

\[ \vec{Y} = F(\vec{X}) \]

\[ \vec{X} = \vec{Y} \]
Merging computation paths

\[ \vec{X} = \vec{Y} \]

\[ \vec{Y} = F(\vec{X}) \]

\[ \vec{X} = \vec{Y} \]

\[ \vec{X} = \vec{I} \]
Merging computation paths

\[ \vec{X} = \vec{Y} = F(\vec{X}) = \vec{I} = \vec{Y} = \vec{X} \]
Merging computation paths

\[ \vec{X} = \vec{Y} \]

\[ \vec{X} = \vec{I} \]

\[ \vec{Y} = F(\vec{X}) \]

\[ \vec{X} = \vec{Y} \]
Merging computation paths

$\vec{X} = \vec{Y}$

$\vec{Y} = F(\vec{X})$

$\vec{X} = \vec{Y}$

$\vec{X} = \vec{I}$
Merging computation paths

\[ \vec{X} = \vec{Y} \]

\[ \vec{X} = \vec{I} \]

\[ \vec{Y} = F(\vec{X}) \]

\[ \vec{X} = \vec{Y} \]
Merging computation paths

\[ \vec{X} = \vec{Y} \]
\[ \vec{X} = \vec{I} \]
\[ \vec{Y} = F(\vec{X}) \]

BUILD
Merging computation paths

\[
\vec{X} = \vec{Y} = F(\vec{X}) = \vec{I} = \vec{Y} = \vec{X}
\]
Overview

1. Introduction
2. Case studies
3. Concrete semantics
4. Generic approximation
5. Filter domains
6. Basic simplified filters
7. Higher order simplified filters
8. Bounded expansion
9. Conclusion
Simplified second order filter

**Theorem 5 (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. \[
\left\{ \begin{array}{l}
\sqrt{-b} + \delta < 1 \\
K \geq \left( \frac{m}{1 - \sqrt{-b} - \delta} \right)^2 \quad \implies \quad Z^2 - aZX - bX^2 \leq K,
\end{array} \right.
\]

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

• The domain relates the variables describing the last two outputs and the four filter parameters to the square root of the ellipsis 'radius':
  \( \gamma_{B_1}(X, Y, a, \varepsilon_a, b, \varepsilon_b, k) \) is given by the set of environments \( \rho \) that satisfy:
  \[
  (\rho(X))^2 - a\rho(X)\rho(Y) - b(\rho(Y))^2 \leq k^2;
  \]

• in order to interpret assignment \( Z = E \) under range constraints \( \rho^\# \), we test whether \( E \) matches:
  \[
  [a - \varepsilon_a; a + \varepsilon_a] \times X + [b - \varepsilon_b; b + \varepsilon_b] \times Y + E'
  \]
with \( a^2 + 4b < 0 \),
and capture:
  – filter parameters: \( (a, \varepsilon_a, b, \varepsilon_b) \);
  – variables tied before \( (X, Y) \) and after the iteration \( (Z, X) \),
  – an approximation of the current input: \( \text{EVAL}^\#(E', \rho^\#) \).
Approximated reduced product

Initial conditions

Output refinement
Overview

1. Introduction
2. Case studies
3. Concrete semantics
4. Generic approximation
5. Filter domains
6. Basic simplified filters
7. Higher order simplified filters
8. Bounded expansion
9. Conclusion
Higher order simplified filters

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

\[
S_{n+p} = a_1 S_n + \ldots + a_p S_{n+p-1} + E_{n+p},
\]

where the polynomial \(P = X^p - a_p X^{p-1} - \ldots - a_1 X^0\) 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 \((x^i_n)_{n \in \mathbb{N}}\) and \((y^j_n)_{n \in \mathbb{N}}\) such that:

\[
\begin{align*}
S_n &= \left( \sum_{i=1}^{k} x^i_n \right) + \left( \sum_{j=1}^{l} y^j_n \right) \\
x^i_{n+2} &= \alpha_i.x^i_{n+1} + \beta_i.x^i_n + F^i(E_{n+2}, E_{n+1}) \\
y^j_{n+1} &= \delta_j.y^j_n + G^j(E_{n+1}).
\end{align*}
\]

The initial outputs \((x^i_0, x^i_1, y^j_0)\) 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

Whenever we meet an assignment $V_{n+p} = E_{n+p} + \sum_{k=1}^{p} I_k \times V_{n+k-1}$,

1. we consider the characteristic polynomial $P = X^p - \sum_{k=1}^{p} I_k \cdot X^{p-k}$,

2. we take a polynomial $Q$ of the form $\prod_{i=1}^{k} (X^2 - A_i X - B_i) \prod_{j=1}^{l} (X - D_j)$ with $2k + l = p + 1$.

3. we expand $Q$ into $X^p - \sum_{k=1}^{p} J_k \cdot X^{p-k}$.

4. we bound the expression $| \sum_{k=1}^{p} (I_k - J_k) \times V_{n+k-1} | \leq \text{err}(V_n, \ldots, V_{n+p-1})$;

5. we take the following assignment:

$$V_{n+p} = E_{n+p} + [-\text{err}(V_n, \ldots, V_{n+p-1}), +\text{err}(V_n, \ldots, V_{n+p-1})] + \sum_{k=1}^{p} J_k \times V_{n+k-1}$$

instead.

A sound factoring algorithm is not required!
Overview

1. Introduction
2. Case studies
3. Concrete semantics
4. Generic approximation
5. Filter domains
6. Basic simplified filters
7. Higher order simplified filters
8. Bounded expansion
9. Conclusion
Other filters

We consider sequences of the following form:

\[
\begin{align*}
S_k &= i_k, \quad 0 \leq k < p \\
S_{n+p} &= F(S_n, \ldots, S_{n+p-1}) \mp G(E_{n+p+1-q}, \ldots, E_{n+p})
\end{align*}
\]

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

- the contribution of the errors ($\varepsilon_n$);

\[
\begin{align*}
\varepsilon_k &= 0, \quad 0 \leq k < p; \\
\varepsilon_{n+p} &= F(\varepsilon_n, \ldots, \varepsilon_{n+p-1}) + \text{err}_{n+p}
\end{align*}
\]

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

\[
\begin{align*}
R_k &= i_k, \quad 0 \leq k < p \\
R_{n+p} &= F(R_n, \ldots, R_{n+p-1}) + G(E_{n+p+1-q}, \ldots, E_{n+p})
\end{align*}
\]
Bounding $R_n$

To refine the output, we need to bound the sequence $R_n$:

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

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

2. Since the filter is linear, we have, for $n > N + p$:

- $\text{last}_n^N(X_1, \ldots, X_N) = \text{last}_{N+p}^N(X_1, \ldots, X_N)$;
- $\text{res}_n^N$ satisfies:

$$\text{res}_{n+p}^N = F(\text{res}_n^N, \ldots, \text{res}_{n+p-1}^N) + G'_{[F,G]}(E_{n+p-N+1-q}, \ldots, E_{n+p-N})$$
Abstract gain with respect to $N$
Overview

1. Introduction
2. Case studies
3. Concrete semantics
4. Generic approximation
5. Filter domains
6. Basic simplified filters
7. Higher order simplified filters
8. Bounded expansion
9. Conclusion
Overview

1. Introduction
2. Case studies
3. Concrete semantics
4. Generic approximation
5. Filter domains
6. Basic simplified filters
7. Higher order simplified filters
8. Bounded expansion
9. Conclusion
Benchmarks

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

<table>
<thead>
<tr>
<th>lines of C</th>
<th>70,000</th>
<th>216,000</th>
<th>379,000</th>
</tr>
</thead>
<tbody>
<tr>
<td>global variables</td>
<td>13,400</td>
<td>7,500</td>
<td>9,000</td>
</tr>
<tr>
<td>iterations</td>
<td>72</td>
<td>41</td>
<td>37</td>
</tr>
<tr>
<td>time/iteration</td>
<td>52s</td>
<td>1mn18s</td>
<td>1mn16s</td>
</tr>
<tr>
<td>analysis time</td>
<td>1h02mn</td>
<td>53mn</td>
<td>47mn</td>
</tr>
<tr>
<td>false alarms</td>
<td>574</td>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>

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 (not necessarily sound) polynomial reduction algorithm for each filter instance.
- filters are detected up to:
  - term recombination
  - and some laws of the real fields;

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

http://www.astree.ens.fr