Abstract Acceleration of General Linear Loops

Bertrand.Jeannet, Peter.Schrammel, Sriram Sankaranarayanan

Motivation and Challenge

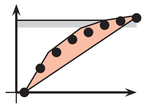
Practical motivation: inferring polyhedral invariants on programs containing **non-trivial numerical loops** (*eg.* control programs).

This requires precise analysis of the behavior of those numerical loops.

Theoretical challenge: numerical loops involving only linear expressions may generate non-linear trajectories.

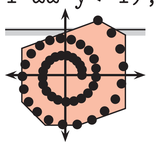
Exponentials:

```
assert(x>=1 && x<=3 && y>=0 && y<=2);
while(y<=3){
   x = 1.5*x;
   y = y+1;
}
```



Spirals:

```
assert(x>=0 && x<=2 && y>=-1 && y<=1);
while(y<=10){
   x = 1.1*x - 0.05*y;
   y = 0.05*x + 1.1*y;
}
```



Our goal is to compute a convex polyhedron containing all such trajectories.

Over-approximate the Set A^*

Closed-form expression of A^k

- Use of real normal Jordan form J of A

$$A^k = Q^{-1}J^kQ$$

- Coefficients of J^k of the form

$$\binom{k}{p}\,\lambda^{k-p}\cos((k-p)\theta - r\tfrac{\pi}{2})$$

Over-approximate $\bigcup_k J^k$ with a set $\mathcal{J}$

- $\vec{m}(k)$: vector of coefficients of J^k
- Bound linear expressions $e_i(\vec{m}(k))$ for $k \geq 0$
- Set of matrices $\mathcal{J}$ defined by a template polyhedron $\{\vec{m} \mid \bigwedge_i a_i \leq e_i(\vec{m}) \leq b_i\}$

Over-approximate A^* with $\mathcal{A} = Q^{-1}\mathcal{J}Q$

Compute $\mathcal{A}X$

$\mathcal{A}$ and X: polyhedra defined by vertices and rays

$$\mathcal{A}X \subseteq \text{ConvexSpan}(\overbrace{V_1V_2}^{\text{vertices}}, \overbrace{V_1R_2 \cup R_1V_2 \ \cup R_1R_2}^{\text{rays}})$$

Approach

Linear loop modelled as a discrete function

$$\tau : G\vec{x} \leq \vec{h} \ \rightarrow \ \vec{x'} = A\vec{x} + \vec{b}$$

Problem: compute $\tau^*(X) = \bigcup_{k\geq 0} \tau^k(X)$ where X is the set of initial states before the loop

Context: abstract interpretation with convex polyhedra as abstract properties

Approach: abstract acceleration: compute "in a single step" some over-approximation

$$\tau^{\otimes}(X) \supseteq (\alpha \circ \tau^* \circ \gamma)(X)$$

Example: Gonnord & al proposed for **translations** $\tau : G\vec{x} \leq h \rightarrow \vec{x'} = \vec{x} + \vec{b}$ the closed-form formula

$$\tau^{\otimes}(X) = X \ \sqcup \ \left(\left((X \sqcap G) \nearrow \{\vec{b}\}\right) \sqcap G\right) + \{\vec{b}\}$$

Existing Techniques

Standard polyhedral analysis
compute iteratively $(\alpha \circ \tau \circ \gamma)^*(I)$

- not precise (at all in case of non-linear behaviour)
- delivers only inductive invariant (good ones are not inductive, see examples)

Previous abstract acceleration methods

- computationally very efficient
- very precise . . . when it applies
- restricted to translations with resets and inputs (linear behaviour)

Ellipsoid methods

- computationally efficient
- restricted to stable loops (A diagonalizable and modules of eigenvalues strictly less than 1)
- ignores the loop condition

Method

Loop without guard: $\tau : \ true \ \rightarrow \ \vec{x'} = A\vec{x}$

(1) Over-approximate the set $A^* = \{A^k \mid k \geq 0\}$ by a template polyhedron matrix $\mathcal{A}$
(2) Given an input polyhedron X, compute $\mathcal{A}X \supseteq \bigcup_{k\geq 0} A^kX$

Loop with guard: $\tau : \ G\vec{x} \leq 0 \ \rightarrow \ \vec{x'} = A\vec{x}$

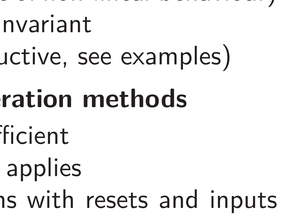
(2') Given an input polyhedron X, over-approximate the maximum number of iterations N of the loop
(3') Over-approximate the set $A^{[0,N-1]} = \{A^k \mid 0 \leq k < N\}$ by a template polyhedron matrix $\mathcal{A}^{[0,N-1]}$
(4') Compute $X \sqcup \tau(\mathcal{A}^{[0,N-1]}(X \sqcap G))$

Example: Exponential

Loop:

```
assert(x>=1 && x<=3 && y>=0 && y<=2);
ξ = 1;
while(y<=3){
   x = 1.5*x;
   y = y+1; }
```

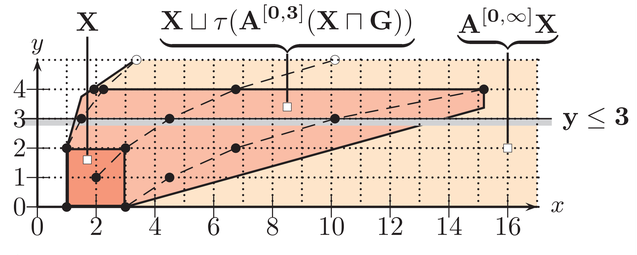
with $\tau : \overbrace{(0\ 1 {-}3)}^{G} \begin{pmatrix} x \\ y \\ \xi \end{pmatrix} \leq 0 \ \rightarrow \ \begin{pmatrix} x' \\ y' \\ \xi' \end{pmatrix} = \overbrace{\begin{pmatrix} 1.5 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}}^{A} \begin{pmatrix} x \\ y \\ \xi \end{pmatrix}$



Over-approximate A^*:

$$\left\{ A^k = \begin{pmatrix} 1.5^k & 0 & 0 \\ 0 & 1 & k \\ 0 & 0 & 1 \end{pmatrix} \middle| \ k \geq 0 \right\} = A^*$$

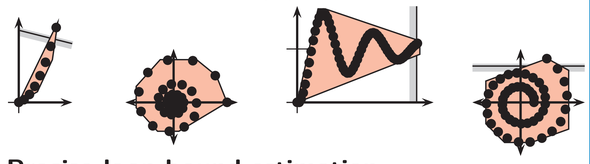
$$\subseteq \left\{ \begin{pmatrix} m_1 & 0 & 0 \\ 0 & 1 & m_2 \\ 0 & 0 & 1 \end{pmatrix} \middle| \begin{array}{l} m_1 \geq 1 \\ m_2 \geq 0 \\ m_1 - m_2 \geq 0.25 \end{array} \right\} = \mathcal{A}^{[0,\infty]}$$



Maximum number of iterations N: $y_0 \in [0,2] \wedge y' = y+1 \wedge y \leq 3 \implies N = 4$

Contributions

Precise polyhedral approximation of any linear loop, based on (i) Jordan normal form (ii) matrix abstract domain and (iii) asymptotic analysis



Precise loop bound estimation
Smooth integration into a static analyzer
Rationale: using non-linear deductions within polyhedra analysis

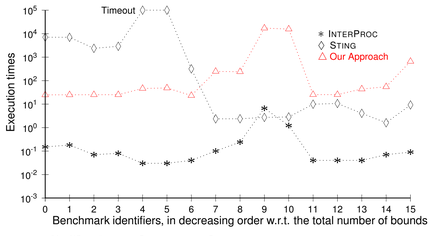
Experiments

Prototype implementation in OCaml based on Apron (polyhedra) and Sage (Jordan form)

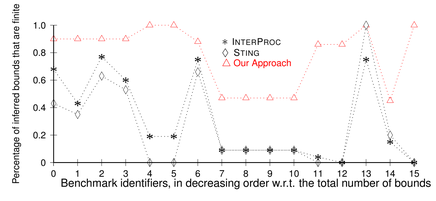
- Loops with conditionals ⇒ multiple self-loops
- Acceleration of inner loop, widening of outer loop

Benchmarks: small filters (single loops), forced oscillators with stable and unstable switching modes (nested loops), thermostat system (nested loops)

Analysis Time:



Number of finite bounds:



Reasonable analysis time in the view of Tremendous gain in precision

Applications and Extensions

- Open systems with inputs (ongoing work)
- Stability/termination analysis
- Hybrid systems with linear differential equations