Generation of Invariants

Outline

⇒ 0. Motivation
1. System Models
2. Program Analysis: Basics
3. Recurrence Technique
4. Polyhedral Analysis
5. Constraint-Based Analysis

Motivation

Trivial Example:

$$
\begin{array}{l}
\textbf{integer } i, j \textbf{ where } i = 2 \wedge j = 0 \\
\ell_0 : \ \textbf{while } (...) \textbf{ do} \\
\quad \left[\begin{array}{l}
\textbf{if } (...) \textbf{ then} \\
\qquad i := i + 4 \\
\textbf{else} \\
\qquad (i, j) := (i + 2, j + 1)
\end{array} \right]
\end{array}
$$

$\boxed{i \geq 2}$ and $\boxed{j \geq 0}$ are <u>invariants</u>.
$\boxed{i - 2j \geq 2}$ is also an invariant.

The Problem: How do we obtain such invariants $\boxed{\text{automatically}}$?

Buffer Overflow Analysis

```
int *a = malloc( sizeof(int) * n);
int i,j,k;
for(i=0; i<n; ++i)
     for(j=0;2*j<=i;++j)
         if (a[i] <= a[2*j+1])
                  .....
     ...
```

$\boxed{0 \leq i < n?}$ $\boxed{0 \leq 2j + 1 < n?}$

Check bounds for each array access.
[Dor et al. : 03], [Wagner et al. : 00]

Division by Zero

```
double a,b,c
....
while  ( b > 0 || c >= 0 ) {
          a = a + b/(c+b-1);
....
}
```

$\boxed{c + b - 1 > 0}$

Prove every divisor non-zero.

Termination

local i, j, k: **integer**

while ($i \leq 100 \;\wedge\; j \leq k$) **do**

$(i, j, k) \;:=\; (j, i+1, k-1)$

od

Termination is proved by the ranking function

$$\boxed{-i - j + k}$$

which decrements by 2 each iteration.

We need the supporting invariant

$$\boxed{-i - j + k \geq 0}$$

to establish termination.

Why Invariants?

- Program analysis
  - Buffer overflows analysis,
  - Division-by-zero analysis,
  - Runtime safety,
  - Reachability,
  - Deadlock-freedom.
- Compiler optimizations.
- Manufacturing systems,
- *Regulatory-networks* in biological systems.

Outline

0. Motivation

⇒ 1. System Models

2. Program Analysis: Basics
3. Recurrence Technique
4. Polyhedral Analysis
5. Constraint-Based Analysis

Transition Systems: Definition

[Manna+Pnueli : 96]

V	:	Set of numerical variables
L	:	Locations
ℓ_{init}	:	Initial Location
Θ	:	Initial Condition
$\mathcal{T} : \{\tau_1, \tau_2, \ldots, \tau_m\}$	:	Set of transitions
τ_i	:	$\left\langle \ell_i, \ell_j, \rho_{\tau_i} \right\rangle$

prelocation   postlocation   transition relation

Transition System: Example

integer i, j **where** $i = 2 \wedge j = 0$

l_0 : **while true do**

$$\left[\begin{array}{c} i := i + 4 \\ \textbf{or} \\ (i, j) := (i + 2, j + 1) \end{array}\right]$$

The corresponding transition system is:

$$\begin{array}{lcl} L & : & \{l_0\} \\ V & : & \{i, j\} \\ \Theta & : & (i = 2 \ \wedge \ j = 0) \\ \tau_1 & : & \langle l_0, \ l_0, \ (i' = i + 4 \ \wedge \ j' = j) \rangle \\ \tau_2 & : & \langle l_0, \ l_0, \ (i' = i + 2 \ \wedge \ j' = j + 1) \rangle \end{array}$$

Transition System: Execution

Transition Systems: $\langle V, L, \ell_{init}, \Theta, \mathcal{T} \rangle$.

$$\langle \ell_0, \boldsymbol{x_0} \rangle \xrightarrow{\tau_1} \langle \ell_1, \boldsymbol{x_1} \rangle \xrightarrow{\tau_2} \langle \ell_2, \boldsymbol{x_2} \rangle \rightarrow \cdots$$

<u>Computation</u>= infinite sequence of <u>states</u> $\langle \ell_i, x_i \rangle$ such that:

- Initial Condition satisfied

$$\ell_0 = \ell_{init} \ \wedge \ \Theta(\boldsymbol{x_0})$$

- Consecutive states $\langle \ell_i, x_i \rangle \rightarrow \langle \ell_{i+1}, x_{i+1} \rangle$ satisfy some transition

$$\tau_k : \ \langle \ell_i, \ell_{i+1}, \rho_{\tau_k}(\boldsymbol{x_i}, \boldsymbol{x_{i+1}}) \rangle$$

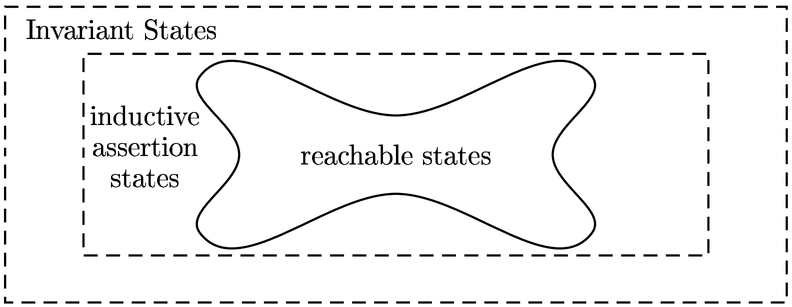
Outline

0. Motivation
1. System Models
⇒ 2. Program Analysis: Basics
3. Recurrence Technique
4. Polyhedral Analysis
5. Constraint-Based Analysis

Invariants

[Floyd : 67] [Hoare : 69]

Assertion ψ is an *invariant of P* iff
it is true at all the *reachable states* of P.



Example:

reachable states : $\langle \ell_0, 2\rangle, \ \langle \ell_0, 4\rangle, \ \langle \ell_0, 8\rangle, \ \langle \ell_0, 16\rangle, \ldots$

invariant : $at\_\ell_0 \ \rightarrow \ x$ is even

Inductive Assertions

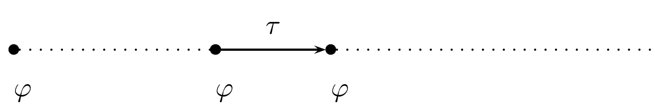
An assertion φ is inductive assertion iff

Initiation The assertion is true initially,

$$\Theta \ \models \ \varphi$$

Consecution For every transition τ, if φ is true before τ is taken, then it is true after taking τ,

$$\varphi \ \wedge \ \rho_\tau \ \models \ \varphi'$$



Invariant vs. Inductive Assertion

Inductive Assertion $\Rightarrow$ Invariant

Invariant $\not\Rightarrow$ Inductive Assertion

To prove invariant φ, find inductive assertion ψ, satisfying initiation + consecution, such that

$$\psi \ \rightarrow \ \varphi$$

Example:

local x **where** $x = 1$
while *true*
$x := -x$

$\boxed{x \leq 1}$ is invariant but not inductive assertion.

Invariant vs. Inductive Assertion (Cont)

Example:

local x **where** $x = 1$
while *true*
$x := -x$

Example: $(x \le 1 \ \wedge \ x \ge -1)$ is an inductive assertion

$$(x \le 1 \ \wedge \ x \ge -1) \ \rightarrow \ (x \le 1)$$

> To find invariants,
> we really search for inductive assertions.

Example

integer i, j **where** $i = 2 \wedge j = 0$
while true do

$$l_0 : \left[\begin{array}{c} i := i + 4 \\ \textbf{or} \\ (i, j) := (i + 2, j + 1) \end{array} \right]$$

Is $\boxed{\varphi : \ i - 2j \ge 2}$ an invariant?

Example: Continued

Show

Initiation: $\underbrace{(i = 2 \ \wedge \ j = 0)}_{\Theta} \ \models \ \underbrace{i - 2j \ge 2}_{\varphi}$

Consecution: Two transitions τ_1 and τ_2.

$$\underbrace{i - 2j \ge 2}_{\varphi} \ \wedge \ \underbrace{(i' = i + 4 \ \wedge \ j' = j)}_{\rho_{\tau_1}} \ \models \ \underbrace{i' - 2j' \ge 2}_{\varphi'}$$

$$\underbrace{i - 2j \ge 2}_{\varphi} \ \wedge \ \underbrace{(i' = i + 2 \ \wedge \ j' = j + 1)}_{\rho_{\tau_2}} \ \models \ \underbrace{i' - 2j' \ge 2}_{\varphi'}$$

$\boxed{\varphi : \ i - 2j \ge 2}$ is inductive assertion $\Rightarrow$ invariant.

Preliminaries: Assertion Domains

Assertion Domain:
A class of assertions, containing: the initial assertion,
the transition relations, and
the target invariants.

Common Examples:

1. Linear Equalities over Reals $\quad 2i + j - 3 = 0$,
2. Linear Inequalities over Reals $\quad 2i + 3j - 2 \le 0$,
   (convex polyhedra)
3. Integer Arithmetic $\quad (\exists\ k)[2j = i \ \wedge \ i = j + k]$,
   (Presburger Arithmetic)
4. Multiplication over Reals $\quad (\exists a, b)[ai^3 + bi = j \ \vee \ i = j^2]$.

Linear Invariants

Invariants in the domain of Linear Equalities or Inequalities are called *Linear Invariants*.

All variables and constants are assumed to range over the reals.

Example:

$$\textbf{integer } i, j \textbf{ where } i = 2 \land j = 0$$

$$\ell_0: \textbf{ while } (...) \textbf{ do} \left[\begin{array}{l} \textbf{if } (...) \textbf{ then} \\ \quad i := i + 4 \\ \textbf{else} \\ \quad (i, j) := (i + 2, j + 1) \end{array} \right]$$

$\boxed{\varphi_1 : \ i \geq 2}$ and $\boxed{\varphi_2 : \ i - 2j \geq 2}$ are linear invariants.

"i is even" is an invariant but not linear.

Outline

Recurrence Equation Techniques

[German + Wegbreit : 74], [Katz + Manna : 75]

For each program variable y,

$\underline{y(n)}$ represents <u>the value of y</u> at step n of the computation.

1. Use the transition relations to <u>deduce</u> recurrence relations.
2. <u>Solve</u> the recurrence relations.
3. <u>Eliminate</u> the step parameter n to obtain invariants.

Example

$$\textbf{integer } i, j \textbf{ where } i = 2 \land j = 0$$

$$l_0: \textbf{ while true do} \left[\begin{array}{c} i := i + 4 \\ \textbf{or} \\ (i, j) := (i + 2, j + 1) \end{array} \right]$$

<u>Base Case</u>: $i(0) = 2, \ j(0) = 0.$

<u>Recurrence Equations</u>:

$$i(n), \ \ j(n) = \begin{cases} i(n-1) + 4, \ \ j(n-1) \\ or, \ i(n-1) + 2, \ \ j(n-1) + 1 \end{cases}$$

Example

Recurrence Equations:

$$i(n),\ \ j(n) = \begin{cases} i(n-1)+4, \ \ j(n-1) \\ or,\ i(n-1)+2, \ \ j(n-1)+1 \end{cases}$$

$$\Downarrow$$

$$\begin{array}{rcccl} i(n-1)+2 & \le & i(n) & \le & i(n-1)+4 \\ j(n-1) & \le & j(n) & \le & j(n-1)+1 \end{array}$$

$$\Downarrow$$

$$2n+2 \ \le\ i(n) \ \le\ 4n+2 \quad \wedge \quad 0 \ \le\ j(n) \ \le\ n$$

$$\Downarrow$$

$$\boxed{i \ge 2} \wedge \boxed{i - 2j \ge 2} \wedge \boxed{j \ge 0}$$

Pros and Cons

Advantages:

1. Many algorithms are analyzed this way by humans.
2. Can produce powerful invariants.

Disadvantages:

1. Recurrence equations are hard to solve.
2. Can be automated only under restricted settings.

Outline

0. Motivation
1. System Models
2. Program Analysis: Basics
3. Recurrence Technique
4. ⇒ Polyhedral Analysis
5. Constraint-Based Analysis

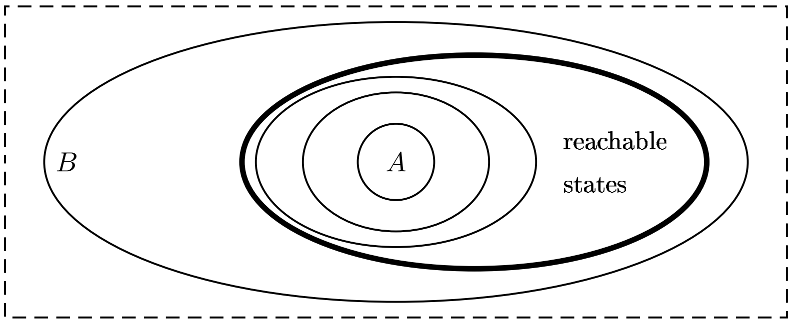
Polyhedral Analysis

Conventional Approach: *Compute* polyhedra containing reachable states. [Cousot+Halbwachs : 78]

These techniques are based on *Abstract Interpretation*.

1. **Initiation:** Start with the initial region and iterate over sets of states.
2. **Iteration:** Expand current iterate with all the states reachable in one step (*post* operator).
3. **Termination:** Enforce convergence by Widening.

Polyhedral Analysis (Cont)



A initial states

B superset of reachable states $\Rightarrow$ invariant

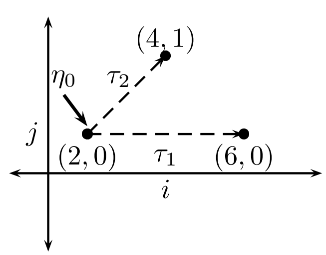
Example

Consider the following transition system:

$$
\begin{aligned}
&L = \{l_0\},\\
&V = \{i, j\},\\
&\Theta:\ (i = 2 \wedge j = 0),\\
&\mathcal{T} = \{\tau_1, \tau_2\},\ \textit{where}\\
&\qquad \tau_1:\ \langle l_0,\ l_0,\ (i' = i + 4 \wedge j' = j)\rangle\\
&\qquad \tau_2:\ \langle l_0,\ l_0,\ (i' = i + 2 \wedge j' = j + 1)\rangle
\end{aligned}
$$

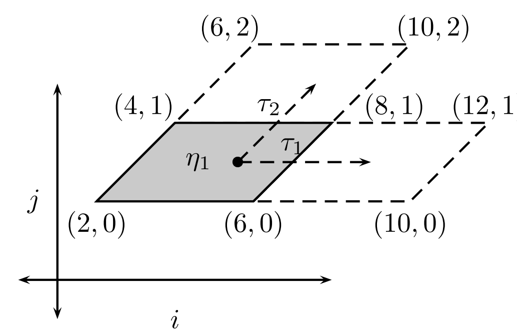
Step 1: Iteration

$$
\begin{aligned}
\eta_0 :&\quad (j = 0) \quad \wedge \quad (i = 2)\\
post(\eta_0, \tau_1) :&\quad (j = 0) \quad \wedge \quad (i = 6)\\
post(\eta_0, \tau_2) :&\quad (j = 1) \quad \wedge \quad (i = 4)\\
\eta_1 :&\quad (0 \le j \le 1) \quad \wedge \quad (2 \le i - 2j \le 6)
\end{aligned}
$$



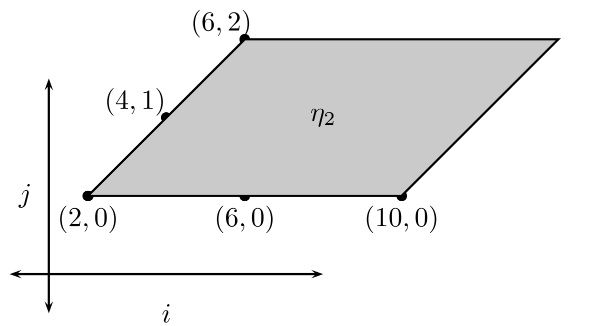
Step 2: Iteration

$$
\begin{aligned}
\eta_1 :&\quad (0 \le j \le 1) \quad \wedge \quad (2 \le i - 2j \le 6)\\
post(\eta_1, \tau_1) :&\quad (0 \le j \le 1) \quad \wedge \quad (6 \le i - 2j \le 10)\\
post(\eta_1, \tau_2) :&\quad (1 \le j \le 2) \quad \wedge \quad (2 \le i - 2j \le 6)\\
\eta_2 :&\quad (0 \le j \le 2) \quad \wedge \quad (2 \le i - 2j \le 10)
\end{aligned}
$$



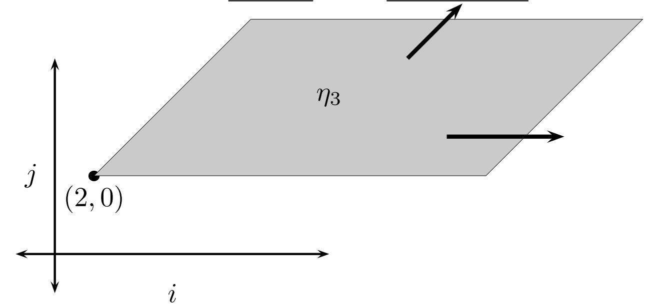
Step 3: Widening Iteration

$$\begin{array}{rlcl}
\eta_1: & (\underline{0 \le j} \le 1) & \wedge & (\underline{2 \le i - 2j} \le 6) \\
\eta_2: & (\underline{0 \le j} \le 2) & \wedge & (\underline{2 \le i - 2j} \le 10) \\
\eta_3(\text{widening}): & (0 \le j) & \wedge & (2 \le i - 2j)
\end{array}$$



Iteration: Step 4

$$\begin{array}{rlcl}
\eta_3: & (0 \le j) & \wedge & (2 \le i - 2j) \\
post(\eta_3, \tau_1): & (0 \le j) & \wedge & (2 \le i - 2j) \\
post(\eta_3, \tau_2): & (0 \le j) & \wedge & (2 \le i - 2j) \\
\eta_4: & \underline{(0 \le j)} & \wedge & \underline{(2 \le i - 2j)}
\end{array}$$



Note: Termination of iteration, $\eta_4 = \eta_3$.
The final invariants are $\boxed{0 \le j} \wedge \boxed{2 \le i - 2j} \Rightarrow \boxed{i \ge 2}$.

Pros and Cons

Advantages:

1. Can be implemented across numerous assertion domains,

   Linear Inequalities ([Cousot+Halbwachs : 78]
   HyTech [Alur+Henzinger : 90s]
   [Henzinger + Ho : 95])
   Presburger Arithmetic
   ([Bultan++ : 97]
   [Wolper+Boigelot : 03])

2. Can scale to larger programs over simple domains.

Disadvantages:

1. Widening operation involves heuristic guess.

Outline

0. Motivation
1. System Models
2. Program Analysis: Basics
3. Recurrence Technique
4. Polyhedral Analysis

$\Rightarrow$ 5. Constraint-Based Analysis

Overview

[Colón++ : 2003]

Linear Inequalities over reals:

1. Target invariant (Template):

$$\psi : \ \underline{c_1}x_1 + \cdots + \underline{c_n}x_n + \underline{d} \leq 0.$$

2. Generate constraints on the coefficients by encoding

$$\begin{array}{rcll} \Theta & \models & \psi & \text{(initiation)} \\ \psi \wedge \rho_\tau & \models & \psi' & \text{(consecution)} \end{array}$$

Use Farkas Lemma.

3. Solve the constraints.

4. Any solution is an invariant.

Farkas' Lemma

[Farkas : 1902]

Let S be a system of linear inequalities over real-valued variables $x_1, \ldots, x_n$,

$$S : \left[\begin{array}{ccccccccc} a_{11}x_1 & + & \cdots & + & a_{1n}x_n & + & b_1 & \leq 0 \\ \vdots & & & & \vdots & & \vdots & \\ a_{m1}x_1 & + & \cdots & + & a_{mn}x_n & + & b_m & \leq 0 \end{array}\right]$$

and ψ a linear inequality,

$$\psi : \ \underline{c_1}x_1 + \cdots + \underline{c_n}x_n + \underline{d} \leq 0$$

then, $S \models \psi$ iff S is unsatisfiable, or there exist *real multipliers* $\underline{\lambda_1}, \ldots, \underline{\lambda_m} \geq 0$ such that:

$$\underline{c_1} = \sum_{i=1}^{m} \underline{\lambda_i} a_{i1} \quad \ldots \quad \underline{c_n} = \sum_{i=1}^{m} \underline{\lambda_i} a_{in} \ \ \underline{d} \leq (\sum_{i=1}^{m} \underline{\lambda_i} b_i)$$

Example: Program

integer i, j **where** $i = 2 \wedge j = 0$

$\ell_0 :$ **while true do**

$$\left[\begin{array}{c} i := i + 4 \\ \textbf{or} \\ (i, j) := (i + 2, j + 1) \end{array}\right]$$

Example: Transition System

$L = \{\ell_0\}$,

$V = \{i, j\}$,

$\Theta : \ (i = 2 \wedge j = 0)$,

$\mathcal{T} = \{\tau_1, \tau_2\}$, *where*

$\tau_1 : \ \langle \ell_0, \ \ell_0, \ (i' = i + 4 \wedge j' = j) \rangle$

$\tau_2 : \ \langle \ell_0, \ \ell_0, \ (i' = i + 2 \wedge j' = j + 1) \rangle$

Example: Initiation

Target invariant $\boxed{\psi :\ \underline{c_1} i + \underline{c_2} j + \underline{d} \le 0}$.

$$
\left.\begin{array}{c|ccccc}
\underline{\lambda_1} & i & & -2 & = 0 \\
\underline{\lambda_2} & & j & & = 0 \\
\hline
 & \underline{c_1} i & + \ \underline{c_2} j & + \ \underline{d} & \le 0 & \leftarrow \psi
\end{array}\right\} \Theta \quad
\begin{array}{|ccccc|}
\hline
i & - & 2 & \le & 0 \\
-i & + & 2 & \le & 0 \\
\hline
\end{array}
$$

$$\exists\ \underline{\lambda_1}, \underline{\lambda_2}\ \left[\underline{\lambda_1} = \underline{c_1}\ \wedge\ \underline{\lambda_2} = \underline{c_2}\ \wedge\ \cdots\cdots\right]$$

The constraint after elimination of the $\underline{\lambda}$'s is

$$\boxed{2\underline{c_1} + \underline{d} \le 0}$$

Example: Consecution

Encoding Consecution for τ_1 : $\boxed{\psi\ \wedge\ \rho_{\tau_1}\ \models\ \psi'}$

$$
\begin{array}{c|cccccccl}
\underline{\mu_1} & \underline{c_1} i & + \ \underline{c_2} j & & & & + \ \underline{d} & \le 0 & \leftarrow \psi \\
\underline{\lambda_1} & i & & - \ i' & & & + \ 4 & = 0 & \\
\underline{\lambda_2} & & j & & - \ j' & & & = 0 & \\
\hline
 & & & \underline{c_1} i' & + \ \underline{c_2} j' & & + \ \underline{d} & \le 0 & \leftarrow \psi'
\end{array}
\quad \left.\begin{array}{l} \\ \\ \end{array}\right\} \rho_{\tau_1}
$$

$$\exists\ \underline{\lambda_1}, \underline{\lambda_2}, \underline{\mu_1}\ \left[\underline{\mu_1} \ge 0\ \wedge\ \underline{\mu_1}\underline{c_1} + \underline{\lambda_1} = 0\ \wedge\ \cdots\cdots\right]$$

The final result after elimination is:

$$\boxed{(\underline{c_1} \le 0)\quad \vee\quad (\underline{c_1} = 0\ \wedge\ \underline{c_2} = 0)}$$

Example: Combined Constraint

The overall constraint is:

$$
\begin{array}{cl}
(2\underline{c_1} + \underline{d} \le 0) & \leftarrow \textit{Initiation} \\
\wedge & \\
\left[\begin{array}{l} (\underline{c_1} \le 0) \vee \\ (\underline{c_1} = 0 \wedge \underline{c_2} = 0) \end{array}\right] & \leftarrow \tau_1\ \textit{consecution} \\
\wedge & \\
\left[\begin{array}{l} (2\underline{c_1} + \underline{c_2} \le 0) \vee \\ (\underline{c_1} = 0 \wedge \underline{c_2} = 0) \end{array}\right] & \leftarrow \tau_2\ \textit{consecution}
\end{array}
$$

This simplifies to

$$\boxed{2\underline{c_1} + \underline{d} \le 0\ \wedge\ \underline{c_1} \le 0\ \wedge\ 2\underline{c_1} + \underline{c_2} \le 0}$$

Summary

1. Fix a *template assertion* with unknown coefficients,

$$\underline{c_1} i + \underline{c_2} j + \underline{d} \le 0 \qquad \cdots \text{Target Invariant}$$

2. Compute constraints on the unknown coefficients,

$$2\underline{c_1} + \underline{d} \le 0\ \wedge\ \underline{c_1} \le 0\ \wedge\ 2\underline{c_1} + \underline{c_2} \le 0$$

3. Solve these constraints

$$\langle \underline{c_1}, \underline{c_2}, \underline{d} \rangle = \langle 0, -1, 0 \rangle\ ,\ \langle \underline{c_1}, \underline{c_2}, \underline{d} \rangle = \langle -1, 2, 2 \rangle$$

4. Generate the invariants

$$
\begin{array}{ccc}
\langle 0, -1, 0 \rangle & \leftrightarrow & j \ge 0 \\
\langle -1, 2, 2 \rangle & \leftrightarrow & i - 2j \ge 2
\end{array}
$$

Invariants: $\boxed{j \ge 0} \wedge \boxed{i - 2j \ge 2} \Rightarrow \boxed{i \ge 2}$

Constraint Solving

- Use exact quantifier elimination over the reals.
  - Decidable [Tarski : 31]
  - Cylindrical-Algebraic Decomposition (CAD) [Collins : 75]
    - QEPCAD [Hong : 93]
    - Mathematica
  - QE for quadratic constraints [Weispfenning : 92]
    - REDLOG [Dolzmann+Sturm : 97]
- Specialized techniques:
  - Heuristic techniques using CLP-style solvers. Scales to medium-sized systems
  - Linear closed form for Petri nets and extensions. Scales to large systems

  Prototype implementation available.

Pros and Cons

Advantages:

1. No widening heuristic.
2. All inductive invariants are generated, or obtained as consequences.

Disadvantages:

1. Constraints may be hard to solve completely.

Related Approaches

- P-invariants in Petri Nets. [Frances : 76], [Manna+Pnueli : 80], [Murata : 89]
- Set constraints in Program Analysis. [Heintze; Aiken;... : 90s]
- Symbolic bounds analysis of pointer arithmetic. [Rugina+Rinard : 00]
- Ranking function generation. [Colón+ Sipma : 02]
- Lyapunov functions. [Forsman : 93]
- Barrier sets for reachability analysis. [Prajna+Jadbabaie : 04]

Non-linear?

Problem: Discover *non-linear equality* invariants over reals?

Solution: Algebraic Geometry:

Gröbner Bases and the Theory of Ideals.

[Büchberger : 65] [Cox et al. : 96] [Sankaranarayanan++ : 04]