Approximating the Transitive Closure of a Boolean Affine Relation

Paul Feautrier

ENS de Lyon
Paul.Feautrier@ens-lyon.fr

January 22, 2012
Definitions and Motivations

The Basic Algorithm
  Characterization
  Frakas Lemma
  Comparison to the ACI Method

A Piecewise Extension

Conclusions
Definitions

- A relation on a set $E$ is a subset of $E \times E$.
- A Boolean expression on $\mathbb{N}^d$ or $\mathbb{Z}^d$ is a Boolean combination of affine inequalities $\sum_{i=1}^d a_i x_i + x_0 \geq 0$ or $\sum_{i=1}^d a_i x_i + x_0 > 0$ on $d$ variables.
- A Boolean affine relation is a Boolean affine expression in which one has distinguished input and output variables, e.g. with primes.
- Relation union, relation composition
  $(R \circ S)(x, y) = \exists z : R(x, z) \& S(z, y)$.
- Transitive closure of $R$: the smallest reflexive and transitive relation which includes $R$:

$$R^+ = R \cup R^2 \cup \ldots \cup R^k \ldots ; \quad R^* = I \cup R^+$$

$$R^1 = R ; \quad R^{n+1} = R \circ R^n$$
Motivation

Boolean affine relations are ubiquitous in static program analysis:
- loop invariants
- “transformers”
- dependences and value-based dependences

Transitive closures are useful in many cases:
- program verification and termination
- loop scheduling (Pugh)
- communication-free parallelism
Over-Approximations

Unfortunately, the transitive closure of a Boolean affine relation is not always Boolean affine:

The transitive closure of
\[(x' = x + y) \& (y' = y) \& (i' = i + 1)\]
is:

\[(i' > i) \& (x' - x = y.(i' - i)) \& y' = y),\]

which is not affine.

One has to resort to over- or under-approximations. This talk concentrates on over-approximations. A common over-approximation is to ignore the fact that variables may be integral.
Related Works

- Kelly, Pugh et. al. introduced the idea of d-relations, i.e. relations on $x' - x$, which can be summed to build the transitive closure.

- Ancourt, Coelho and Irigoin generalized the idea by introducing the distance set: $(\Delta R)(d) = \exists x : R(x; x + d)$.

- Sankaranarayanan et. al. applied Farkas lemma to the conditions $R \subseteq R^+$ and $R \circ R^+ \subseteq R^+$ but the result was a bilinear system, to be solved by quantifier elimination or rewriting.

Kelly, Pugh et. al.: LCPC’95
Ancourt, Coelho, Irigoin: NSAD’2010
Sankaranarayanan, Sipma, Manna: SAS’2004
If $R$ is reflexive and transitive, then
\[ \approx_R \equiv \{ x, x' \mid R(x; x') \land R(x'; x) \} \]
is an equivalence relation.

The quotient relation $R/ \approx_R$ is an order.

Hence $R$ can be written as $R(x; x') \equiv f_R(x) \prec_R f_R(x')$ where $f_R$ is the mapping from the universe to the equivalence classes of $\approx_R$, and $\prec_R$ is the quotient order.

For finite graphs, the equivalence classes are the strongly connected components, and $\prec_R$ is the transitive closure of the reduced graph.
Application, I

Select a shape for $f$ – for instance, a linear function $f(x) = f \cdot x$ – and an order – for instance the ordinary order $\leq$ – and solve the constraint:

$$R(x; x') \Rightarrow f \cdot x \leq f \cdot x'$$

- The resulting relation $S(x; x') \equiv f \cdot x \leq f \cdot x'$ is an over approximation of $R^*$.
- An improved result is $S(x; x') \cap (\mathcal{D}(R) \times \mathcal{C}(R))$, the domain and codomain of $R$.
- If $R$ is Boolean affine, then the constraint can be solved using Farkas lemma.
Farkas Lemma

If the system of constraints $Ax + b \geq 0$ is feasible, then:

$$\forall x. (Ax + b \geq 0 \Rightarrow c.x + d \geq 0) \equiv \exists \Lambda \geq 0: c = \Lambda A \ & \ d \geq \Lambda b$$

- If $R$ is convex: $R(x; x') \equiv Ax + A'x' + a \geq 0$, then application of Farkas lemma gives the system:

  $$\Lambda A = -f, \ \Lambda A' = f, \ \Lambda a \leq 0.$$

- If $R$ is non convex, apply Farkas to each clause in its DNF. The result is a system of inequalities in positive unknowns.
Application, II

- Eliminate \( \Lambda \) (the Farkas multipliers) independently for each subsystem
- The resulting system for \( \mathbf{f} \) is homogeneous and hence defines a cone
- Let \( r_1, \ldots, r_n \) be the rays of this cone. Each ray \( r_i \) define a valid function \( f_i(x) = r_i . x \); all other vectors in the cone define redundant functions.
- The resulting approximation to \( R^* \) is:

\[
S(x; x') \equiv \bigwedge_{i=1}^{n} f_i(x) \leq f_i(x').
\]

- \( \prec \) is the Cartesian product order \( \leq^n \).
An Example

Consider the following relation from Sankaranarayanan et al.:

\[(x' = x + 2y \& y' = 1 - y) \lor (x' = x + 1 \& y' = y + 2)\]

Let \(f(x) = f_1x + f_2y\) be the unknown.

- The first clause gives the constraint \(f_1 = f_2 \geq 0\)
- The second clause gives the constraint \(f_1 + 2f_2 \geq 0\)
- One can take \(f_1 = f_2 = 1\) and the transitive closure is \(x + y \leq x' + y'\).
Relation to the ACI method

Starting from:

\[ \Lambda A = -f, \quad \Lambda A' = f, \quad \Lambda a \leq 0. \]

one can eliminate \( f \) instead of \( \Lambda \), giving \( \Lambda (A + A') = 0 \)

In the definition of the distance set

\[ (\Delta R)(d) = \exists x : Ax + A'(x + d) + a \geq 0 \]

elimination of \( x \) means finding – e.g. by Fourier-Motzkin – a positive matrix \( L \) such that \( L(A + A') = 0 \). \( L \) can be chosen equal to \( \Lambda \). If \( L.a \leq 0 \) the ACI method gives \( LA'(x' - x) \geq -La \).

The basic algorithm gives \( f = \Lambda A' \) and \( \Lambda A'(x' - x) \geq 0 \).

The two methods gives equivalent results, one giving an approximation for \( R^+ \) and the other for \( R^* \).
When the number of clauses increases, the method fails ($f(x) = 0$) since the number of constraints increases but not the number of unknowns. An example:

$$(x < 100 \ & \ x' = x + 1) \lor (x \geq 100 \ & \ x' = 0).$$

One possible solution: take $f$ as a piecewise affine function:

$$f(x) = \text{if } \sigma(x) \geq 0 \text{ then } g(x) \text{ else } h(x),$$

where $\sigma$, the split function, is taken to be affine:

$$\sigma(x) = \sigma.x + \sigma_0$$
The hyperplanes $\sigma(x) \geq 0$ and $\sigma(x') \geq 0$ split $E \times E$ into 4 regions, in which Farkas lemma can be applied, giving 4 systems of constraints. For instance:

$$R(x; x') \land \sigma(x) \geq 0 \land \sigma(x') \geq 0 \Rightarrow g(x) \leq g(x').$$

If $\sigma$ is known, the systems are still linear, and can be solved as above.
Another Example

For:

\[ R(x; x') \equiv (x < 100 \ & \ x' = x + 1) \lor (x \geq 100 \ & \ x' = 0). \]

and taking \( \sigma(x) = x \), one obtain (after simplification):

\[ R^*(x; x') \equiv (x = x') \lor ((x' < 101) \ & \ ((x \leq x') \lor (0 \leq x'))). \]
How to Choose the Split

- Note that $\sigma(x)$ and $a.\sigma(x)$ gives equivalent systems, whatever the sign of the constant multiplier $a$.

- By manipulating the resulting systems, one can prove that for each clause in the DNF of $R$, either $\sigma$ has a zero Farkas multiplier, or $\sigma$ must belong to the cone generated by the rows of $A + A'$.

- There are only a finite number of possibilities, which can be explored systematically. When the homogeneous part $\sigma.x$ is selected, one obtain a linear system for $\sigma_0$.

- For the exemple above, which is one-dimensional, there is only one possibility, $\sigma = 1$, and then one can show that $\sigma_0$ must be null.
The method has been implemented in Java, using PIP and the Polylib.

The algorithm for choosing $\sigma$ is not implemented yet, and the user must supply it if necessary.
Complete the implementation (choice of $\sigma$, detection of special cases)

- Preprocessing of $R$: change of variables, grouping, adding or removing variables ...

- Can one have more than one split (exponential complexity)

- Explore other forms for the function $f$ (max and min) and other orders (lexicographic orders)

- Explore other representations of the transitive closure