Interprocedurally Analyzing Linear Inequalities

Helmut Seidl, Andrea Flexeder and Michael Petter

Technische Universität München, Boltzmannstrasse 3, 85748 Garching, Germany,
{seidl, flexeder, petter}@cs.tum.edu,
WWW home page: http://www2.cs.tum.edu/~{seidl,flexeder,petter}

Abstract. We present an abstraction of the effect of procedures through convex sets of transition matrices. Conditional branching is handled by postponing the conditional evaluation after the procedure call. In order to obtain an effective analysis convex sets are approximated by polyhedra. For an efficient implementation we approximate polyhedra by means of simplices.

In [CH78], Cousot and Halbwachs present an *intraprocedural* analysis of linear inequalities based on an abstraction of the collecting semantics [CC92] by means of convex polyhedra. They draw upon both the frame and the constraint representation of polyhedra to perform their standard widening [CH78]. More precise widening strategies on convex polyhedra are provided in [BZHR03]. Based on this approach, an *interprocedural* analysis can be obtained by relating input and output states of a procedure call by means of linear inequalities. This leads to transition invariants on program variables before and after the procedure call. In the simple example in figure 1 (from [MOS04]) the relational semantics of procedure `f` at program state 2 can be described by the transition invariants $x = x_{old} \vee x = 2 \cdot x_{old} - 2$. The approximation of these invariants by polyhedra leads to a complete loss of information. Although this approach works in several practical cases (e.g. `McCarthy91` function [MM70]), it is too restrictive for a precise interprocedural analysis.

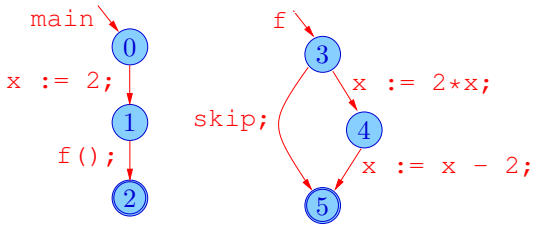

Fig. 1. example CFG

Instead, we propose an *alternative interprocedural* analysis, which is based on convex sets of transition matrices, to capture the effect of procedures. Our analysis considers control flow graphs (CFG) with linear assignments (e.g. `x := y + 5`), procedure calls (e.g. `f()`), linear conditions (e.g. `(x-y>3)`) and loops. For our intraprocedural reachability analysis, the program states are abstracted by convex sets of vectors, describing the values of the program variables. The transformation of program states is described by linear *transition matrices* similar to [MOS04].

The key idea of our approach is to compute finite representations for the effect of procedures [MOS05]. In our approach this effect is represented as a convex set of transition matrices, which can be used to describe procedure calls. In absence of conditional branching, this abstraction can be characterized precisely by means of the least solution of a constraint system. Since the conditionals cannot be represented by linear transformations, they cannot be evaluated on the convex sets of transition matrices. Instead, we introduce auxiliary variables for each condition and postpone their check after the method call. This condition evaluation results in the intersection of a half space, corresponding to the condition, with the convex set of vectors. One alternative for handling conditionals within the reachability analysis comprises the evaluation directly after each procedure call. A second alternative implies a single evaluation at the end of the analysis.

Polyhedra. In order to obtain an effective analysis, we approximate convex sets by means of convex polyhedra. We avoid the expensive continual conversion between the two polyhedral representation forms (frame set, constraint system [CH78]) by resorting to the *frame representation*. An *effective composition* can be achieved by operating on this representation only. In order to infer the linear inequalities for a program point, the conversion to the constraint representation has to be performed only once at the very end of the analysis. The *effective subsumption* test as well as the *convex unification* for the polyhedral frame representation is reduced to linear programming [CH78]. Thus, both operations are rather expensive [SCSM]. This induces the demand for more efficient representations of convex sets.

A notable representation is introduced by *Sankaranarayanan* et al. in [SSM04], which tries to prevent widening as done in [CH78]. This is achieved by generic inequality templates and solving systems of inequalities on the coefficients of the templates. However, there is no feasible way to perform this proceeding. In [Min01b] *octagons*, an efficient subclass of polyhedra, are introduced. Within this approach only two program variables are considered, with restrictions on the coefficients on the program variables, permitting only inequalities of the form $\pm x \pm y \leq c$. *Simon* et al. abandon all restrictions on the coefficients for the two considered program variables [SKH02]. Another approach [Min01a] only permits inequalities of the form $x - y \leq c$, respectively $\pm x \leq c$, which represent polyhedra as *difference-bound matrices*. In contrast, *Clarisó* et al. propose in [CC04] to approximate *polyhedra* with *octahedra*. In contrast to the former approaches, they allow any number of program variables where the coefficients of the inequalities are only restricted to $\pm 1, 0$. All these approaches consist in specially restricted constraint systems, neglecting that their frame representation is possibly exponential in the number of constraints.

Simplices. By no means this does hold for *simplices*, a subclass of polyhedra, because for simplices the frame representation has approximately the same size as the constraint representation. Simplices are convex polyhedra, which are restricted in the number of frame elements to maximal n *linear independent elements* and a base point. Based on this approximation of polyhedra, we achieve reducing the subsumption test to solve a system of n linear equations. Thus, our subsumption test is quite *efficient*, cubic in n. The composition can therefore be effectivly performed as specified for polyhedra. Additionally, we may have to widen to the enclosing simplex containing the result. The

only challenge is given when the conditions are evaluated, as the result of the intersection is not forced to result in a simplex. This is the reason why the condition check is postponed to the end of the analysis. Another possibility is the use of simplices only for efficiently representing the procedure effect. So the conditions can still be evaluated after each procedure call resorting to polyhedra.

An integer analysis over integer variables can be performed in $\mathbb{Z}$ up to the condition check, which is executed at the end of the analysis. The use of an ILP solver is recommended to obtain an integer solution, otherwise the intersections can be computed on rationals.

First practical experiments indicate the resulting analysis is rather efficient and provides reasonably precise results. Regarding the example in figure 1, the linear inequality which is identified by our approach to be valid at program point 2 results in $x = 2$. This is the exact relation holding at this program point.

References

BZHR03. Roberto Bagnara, Enea Zaffanella, Patricia M. Hill, and Elisa Ricci. Precise widening operators for convex polyhedra. In *10th International Static Analysis Symposium (SAS) 2003*, pages 337–354, 2003.

CC92. P. Cousot and R. Cousot. Abstract interpretation frameworks. *Journal of Logic and Computation*, 2(4):511–547, 1992.

CC04. Robert Clarisó and Jordi Cortadella. The Octahedron abstract domain. In *11th International Static Analysis Symposium (SAS) 2004*, pages 312–327, 2004.

CH78. Patrick Cousot and Nicholas Halbwachs. Automatic discovery of linear restraints among variables of a program. In *5th Annual ACM Symposium on Principles of Programming Languages (POPL) 1978*, pages 84–97, 1978.

Min01a. Antoine Miné. A new numerical abstract domain based on difference-bound matrices. In *2nd Symposium on Programs as Data Objects (PADO II) 2001*, pages 155–172, 2001.

Min01b. Antoine Miné. The Octagon abstract domain. In *Analysis, Slicing, and Transformation (AST) 2001*, pages 310–319, 2001.

MM70. Z. Manna and J. McCarthy. Properties of programs and partial function logic. 1970.

MOS04. Markus Müller-Olm and Helmut Seidl. Program analysis through linear algebra. In *31th annual ACM Symposium on Principles of Programming Languages (POPL) 2004*, pages 330–341, 2004.

MOS05. Markus Müller-Olm and Helmut Seidl. A generic framework for interprocedural analysis of numerical properties. In *12th Static Analysis Symposium (SAS)*, pages 235–250, 2005.

SCSM. Sriram Sankaranarayanan, Michael Colon, Henry Sipma, and Zohar Manna. Efficient strongly relational polyhedral analysis. In *7th International Conference, Verification, Model Checking and Abstract Interpretation (VMCAI) 2006*.

SKH02. Axel Simon, Andy King, and Jacob M. Howe. Two Variables per Linear Inequality as an Abstract Domain. In *Logic Based Program Development and Transformation (LOPSTR) 2002*, pages 71–89, 2002.

SSM04. Sriram Sankaranarayanan, Henry Sipma, and Zohar Manna. Constraint based linear relations analysis. In *11th International Static Analysis Symposium (SAS) 2004*, pages 53–68, 2004.