Science of Computer Programming ••• (••••) •••–•••

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

Inferring linear invariants with parallelotopes

Gianluca Amato, Marco Rubino, Francesca Scozzari *

Università di Chieti-Pescara, Italy

ARTICLE INFO

Article history:
Received 20 May 2016
Received in revised form 27 May 2017
Accepted 31 May 2017
Available online xxxx

Keywords:
Static analysis
Abstract interpretation
Numerical abstract domain
Linear invariant
Parallelotopes

ABSTRACT

We propose a new numerical abstract domain for inferring linear invariants based on parallelotopes. The domain may encode any linear constraint, as the polyhedra abstract domain, while maintaining the efficiency of weakly relational abstract domains, such as intervals and octagons. We provide the full set of abstract operators, define a reduced product with intervals and present an experimental comparison with polyhedra and octagons. According to these experiments, the reduced product we propose is much more precise than both polyhedra and octagons in inferring interval constraints.

1. Introduction

In the field of static analysis by abstract interpretation, much effort has been devoted in designing domains for inferring numerical properties such as "the value of the variable x in program point p is between 0 and 10". In this paper we are mainly interested in linear invariants, which are typically expressed as linear inequalities such as $2x + 3y \leq 42$. The polyhedra domain by Cousot and Halbwachs [1] is able to express any such invariant, but its computational complexity makes it difficult to use it in practice. Many other numerical domains have been proposed to gain efficiency. One of the simplest one is the Cousot and Cousot's interval domain [2], a non-relational abstract domain which can represent only constraints involving a single variable such as $0 \leq x \leq 10$. Analyses using the interval domain are not very precise, since it cannot represent relationships between variables. Many weakly relational domains, more expressive than intervals but less expressive than polyhedra, have been proposed in the last years, such as octagons [3], octahedra [4], logahedra [5] and TVPI [6]. Weakly relational domains have proved to be quite efficient, but the invariants that can be inferred using these domains are seriously limited by syntactic restrictions, since expressivity is bartered for efficiency. For instance, the octagons abstract domain can only deal with inequalities involving two variables whose coefficients are $\{-1, 0, 1\}$, such as $x - y \leq 5$, while TVPI can only express inequalities with two variables, such as $5x + 2y \leq 8$.

In order to handle more expressive constraints, Sankaranarayanan et al. have proposed a different approach called template polyhedra [7]. For each program, they fix a constraint matrix A and consider all the polyhedra of the form $\{\boldsymbol{x} \in \mathbb{R}^n \mid A\boldsymbol{x} \leq \boldsymbol{b}\}$, where $\boldsymbol{x}$ is the vector of program variables. The *a priori* choice of the matrix differentiates template polyhedra from other domains, where the matrix is fixed for all the programs (such as intervals or octagons) or varies freely (such as polyhedra). Template polyhedra generalize most weakly relational domains, with the difference that its abstract operators are defined by means of linear programming. Along the same direction there are the proposals of generalized template polyhedra [8], which combine template polyhedra and bilinear forms, and template parallelotopes [9,10], which

* Corresponding author.
E-mail addresses: gianluca.amato@unich.it (G. Amato), marco.rubino@unich.it (M. Rubino), francesca.scozzari@unich.it (F. Scozzari).

http://dx.doi.org/10.1016/j.scico.2017.05.011
0167-6423/

are a special case of template polyhedra. A parallelotope is a polyhedron defined by at most n linearly independent constraints, where n is the number of variables. Any parallelotope can be described as the set of points $\{\boldsymbol{x} \in \mathbb{R}^n \mid \boldsymbol{b}_1 \leq A\boldsymbol{x} \leq \boldsymbol{b}_2\}$, where $\boldsymbol{x}$ is the vector of program variables and the constraint matrix A is square and invertible. In the special case of template parallelotopes, it is possible to derive efficient abstract operators, without resorting to linear programming tools. In general, when using templates, the result of the analyses greatly varies depending on the choice of the matrix A, and finding such matrices is a hard and still open problem. We are aware of only two proposals for generating templates, one in Sankaranarayanan et al. original paper [7] based on a syntactic inspection of the program, and another one based on the statistical analysis of partial execution traces [9,10].

Here we propose a different approach which does not use templates and does not restrict the syntactic shape of constraints, still maintaining efficiency. We propose a domain whose abstract objects are parallelotopes, namely we only limit the number of constraints to n (the number of variables) and we require the constraints to be linearly independent.

While our abstract objects are parallelotopes as in [9,10], there is the fundamental difference that the constraint matrix is not fixed *a priori* but may freely change during the analysis, as for the polyhedra domain. The domain of parallelotopes so defined can encode any linear constraints, does not require templates and can be equipped with very efficient abstract operations, whose complexity is comparable to that of octagons. The key to scalability is the restriction to n linearly independent constraints. This allows the use of algorithms adapted from linear equation solving which are simpler and more efficient than those used in the theory of polyhedra. For instance, minimizing a linear form on a parallelotope is cubic, since it essentially amounts to solving a linear equation system, while the corresponding operation on polyhedra and even on template polyhedra requires the use of the simplex algorithm.

Since many concrete operations are not closed with respect to parallelotopes, in most cases there is not a unique best parallelotope which approximates the result of the concrete operation. The careful design of suitable abstract operators is then a key point in the parallelotope domain. In some cases, as for the abstract union operator, we propose different operators, and in general we resort to appropriate heuristics in order to ensure good precision in the overall analysis.

We also propose a combination of the domain of parallelotopes and intervals, and we experimentally show that it compares well with respect to polyhedra and octagons. Finally, we show that, if we are interested in inferring precise interval constraints, then the combination of parallelotopes and intervals is far more precise than both polyhedra and octagons.

Preliminary results on the domain of parallelotopes appeared in [11]. We provide here a new comprehensive presentation of the domain, containing several novelties such as new abstract operators on parallelotopes, the reduced product of parallelotopes and intervals, experimental evaluations, and proofs (which are in the Appendix).

2. Notation

2.1. Linear algebra

We denote by $\bar{\mathbb{R}}$ the set of real numbers extended with $+\infty$ and $-\infty$. Addition and multiplication are partially extended, when possible, to $\bar{\mathbb{R}}$ in the obvious way. We avoid the use of indeterminate forms, with the exception of 0 times $\pm\infty$ which we define to be 0. We use boldface for elements $\boldsymbol{v}$ of $\bar{\mathbb{R}}^n$. Any vector $\boldsymbol{v} \in \bar{\mathbb{R}}^n$ is intended as a column vector, and $\boldsymbol{v}^T$ is the corresponding row vector. We denote by $\boldsymbol{u} \cdot \boldsymbol{v}$ the dot product of $\boldsymbol{u}$ and $\boldsymbol{v}$ and we use it indifferently for both row and column vectors. Given $\boldsymbol{u}, \boldsymbol{v} \in \bar{\mathbb{R}}^n$, and a relation $\bowtie \in \{<, >, \leq, \geq, =\}$, we write $\boldsymbol{u} \bowtie \boldsymbol{v}$ if and only if $u_i \bowtie v_i$ for each $i \in \{1, \ldots, n\}$. Given $\boldsymbol{u} \in \mathbb{R}^n$ and $i \in \{1, \ldots, n\}$, we write $\boldsymbol{u}[i \mapsto x]$ to denote a vector $\boldsymbol{v}$ such that $v_i = x$ and $v_j = u_j$ for $j \neq i$.

If $A = (a_{ij})$ is a matrix, we denote by A^T its *transpose*. If A is invertible, A^{-1} denotes its inverse, and $\mathrm{GL}(n)$ is the group of $n \times n$ invertible matrices. The identity matrix in $\mathrm{GL}(n)$ is denoted by I_n and the standard basis of $\mathbb{R}^n$ is denoted by $\{\boldsymbol{e}^1, \ldots, \boldsymbol{e}^n\}$. Clearly, any $1 \times n$-matrix can be viewed as a vector: in particular, we denote by $\boldsymbol{a}_{i*}$ the row vector given by the i-th row of any matrix A, and by $\boldsymbol{a}_{*i}$ the column vector given by the i-th column of A.

More generally, if J is a set of indexes, then A_{J*} is the submatrix of A composed of the rows in J and A_{*J} the submatrix composed of the columns in J. We use A_{-J*} for the submatrix of A composed of the rows of A whose indexes are not in J.

In the rest of the paper, when talking about the computational complexity of matrix and vector operations, we always assume to use a dense representation, which is the one adopted in our implementation.

2.2. Convex sets

A set $C \subseteq \mathbb{R}^n$ is *convex* when, given $\boldsymbol{x}, \boldsymbol{y} \in C$, for each $\lambda \in [0, 1]$ we have that $\lambda\boldsymbol{x} + (1 - \lambda)\boldsymbol{y} \in C$. A *ray* in C is a non-null vector $\boldsymbol{v} \in \mathbb{R}^n$ such that for each $\boldsymbol{x} \in C$ and $\alpha \geq 0$, $\boldsymbol{x} + \alpha\boldsymbol{v} \in C$. A convex set C is *bounded* iff it has no rays. A *line* in C is a vector $\boldsymbol{v} \in \mathbb{R}^n$ such that both $\boldsymbol{v}$ and $-\boldsymbol{v}$ are rays. The lines in C form a vector space, called the *linearity space* of C, denote by $\mathrm{lin}(C)$. Given a vector space V, we denote by $V^\perp$ its orthogonal complement.

A set of vectors $S = \{v_1, \ldots, v_m\}$ is *affinely dependent* if there are $\lambda_1, \ldots, \lambda_m \in \mathbb{R}$ such that they are not all zero, $\lambda_1 + \cdots + \lambda_m = 0$ and $\lambda_1\boldsymbol{v}_1 + \cdots + \lambda_m\boldsymbol{v}_m = 0$. The *affine hull* of S is the set $\mathrm{aff.hull}(S) = \{\lambda_1\boldsymbol{v}_1 + \cdots + \lambda_m\boldsymbol{v}_m \mid \lambda_1 + \cdots + \lambda_m = 1\}$.

A *flat* F is a set of points of the form $\boldsymbol{a} + H$ where H is a vector space. Given a flat, H is uniquely determined. Therefore, we call the dimension of F the dimension of the corresponding vector space H. By convention, the dimension of the empty set is -1. The *dimension* of a set S is the dimension of the smallest flat containing S, and it is denoted by $\dim(S)$.

Please cite this article in press as: G. Amato et al., Inferring linear invariants with parallelotopes, Sci. Comput. Program. (2017), http://dx.doi.org/10.1016/j.scico.2017.05.011