

Cartesian Factoring of Polyhedra in Linear Relation Analysis

N. Halbwachs and D. Merchat and C. Parent-Vigouroux

Vérimag*, Grenoble - France

{Nicolas.Halbwachs,David.Merchat,Catherine.Parent}@imag.fr

Abstract. Linear Relation Analysis [CH78] suffers from the cost of operations on convex polyhedra, which can be exponential with the number of involved variables. In order to reduce this cost, we propose to detect when a polyhedron is a Cartesian product of polyhedra of lower dimensions, i.e., when groups of variables are unrelated with each other. Classical operations are adapted to work on such factored polyhedra. Our implementation shows encouraging experimental results.

1 Introduction

Linear Relation Analysis [CH78, Hal79] is one of the very first applications of abstract interpretation [CC77], and remains one of the most powerful and effective techniques for analyzing properties of numerical programs. It was applied in various domains like compile-time error detection [DRS01], program parallelization [IJT91], automatic verification [HPR97, HHWT97] and formal proof [BBC⁺00, BBM97].

In its original setting, Linear Relation Analysis was designed to discover linear inequalities (or convex polyhedra) invariantly holding at each control point of a sequential program. These polyhedra are built by forward and/or backward propagation of predicates along the paths of the control graph of the program. These propagations involve the computation of classical operations over convex polyhedra: intersection, least upper bound (convex hull), affine transformation, projection, check for inclusion and emptiness, and widening to enforce termination. Moreover, it is very important that the representation of polyhedra be kept minimal. All these operations are easy, provided the *double-description* of polyhedra [MRTT53] is available: it consists in characterizing a polyhedron both as the set of solutions of a system of linear inequalities and as the convex hull of a finite system of generators (vertices and rays). Knowing both these representations also allows each of them to be minimized, i.e., discarding irrelevant (redundant) inequalities and generators. Several libraries for polyhedra manipulation are available [Wil93]¹ [CL98]² [HPR97][BRZH02]³.

* Vérimag is a joint laboratory of Université Joseph Fourier, CNRS and INPG associated with IMAG.

¹ see also <http://www.ee.byu.edu:8080/~wilde/polyhedra.html>

² see also <http://icps.u-strasbg.fr/polylib/> and <http://www.irisa.fr/polylib>

³ see also <http://www.cs.unipr.it/pp1/>

The problem with the double-description is that the size of each description can grow exponentially with the dimension of the space (number of variables): an n -dimensional hypercube is defined by $2n$ inequalities, but has 2^n vertices; the converse can happen, since the descriptions are completely dual.

As a consequence, the dimension of the space is an important limitation to the application of Linear Relation Analysis to real-life examples. In program verification, a classical way of decreasing the number of variables, consists of applying first a *program slicing* [Tip95]: it consists in analyzing variables dependencies to discard variables which cannot influence the property to be verified. However, precise slicing is not always easy, and the approach works only when the goal of the analysis is precisely known (e.g., a given property to verify) to be used as the source of the slicing.

In this paper, we propose a complementary approach to reduce the number of variables. It consists in detecting that a polyhedron can be factored as a Cartesian product of polyhedra in smaller dimensions. This situation, which occurs very often in real-life examples, means that the set of variables can be partitioned into subsets, such that variables belonging to different subsets are independent, i.e., not related by any inequality in the polyhedron. In order to take advantage of such factorings, they must be detected, and operations should be, as far as possible, performed on factored arguments.

Compared with slicing, our approach will detect variables independence, not in an absolute way, but with respect to the analysis which is performed: some variables may be related by the concrete semantics of the program, but these relations may be ignored by the Linear Relation Analysis, because of approximation. However, notice that in this paper, we consider factoring without additional loss of information. Of course, one could also decide to abstract away some relations in order to obtain a finer factoring and better performances.

2 Convex Polyhedra

2.1 The double description

Let \mathcal{N} be either \mathbb{R} or \mathbb{Q} . A closed convex polyhedron — or simply, a polyhedron — P in \mathcal{N}^n can be characterized by a $m \times n$ matrix A and a m -vector B , such that

$$P = \{X \in \mathcal{N}^n \mid AX \leq B\}$$

(A, B) is the *constraint description* of P . A polyhedron P can be also characterized by two finite sets $V = \{v_i\}$, $R = \{r_j\}$ of n -vectors, such that

$$P = \left\{ \sum_{i=1}^{|V|} \lambda_i v_i + \sum_{j=1}^{|R|} \mu_j r_j \mid \lambda_i \geq 0, \mu_j \geq 0, \sum_{i=1}^{|V|} \lambda_i = 1 \right\}$$

(V, R) is the *system of generators* of P ⁴; V is the set of *vertices*, R is the set of *rays*. This description expresses that any point in P is the sum of a *convex*

⁴ The system of generators of the empty polyhedron is (\emptyset, \emptyset) .

combination of vertices and a positive combination of rays. Fig. 1 illustrates this double description.

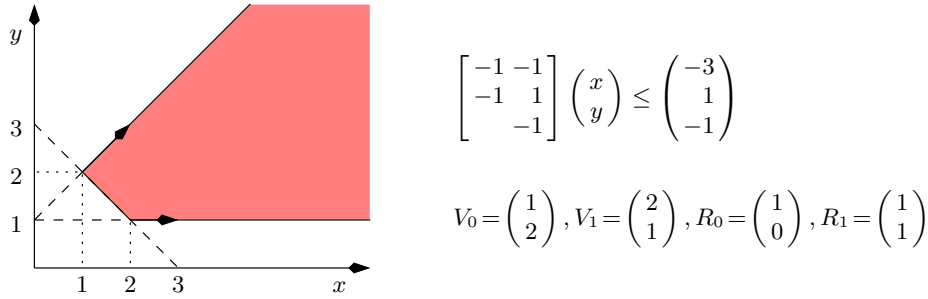


Fig. 1. Double description of a polyhedron.

Chernikova's algorithm [Che68] improved by Leverge [LeV92,Wil93] performs the translation from one description to the other. An important feature of this algorithm is that the resulting description is minimal, in the sense that it does not contain any redundant element: A constraint, a vertex, or a ray is redundant if removing it does not change the polyhedron. In the example of Fig. 1, $x \geq 0$ would be a redundant constraint, $(2, 2)$ would be a redundant vertex, and $(2, 1)$ would be a redundant ray.

Notations: In the rest of the paper, we will generally assimilate a polyhedron with its system of constraints, for instance by noting $P \wedge P'$ the set of solutions of the conjunction of the constraints of P and P' . Notice that we will use this notation even if the constraints of P and P' do not involve the same set of variables, meaning that the conjunction concerns the union of the variables of P and P' .

2.2 Operations on polyhedra

As soon as both descriptions are available, all operations needed for program analysis are available:

Intersection: Let $(A, B), (A', B')$ be the respective systems of constraints of P_1 and P_2 , then $\left(\begin{bmatrix} A \\ A' \end{bmatrix}, \begin{pmatrix} B \\ B' \end{pmatrix} \right)$ (the concatenation, or conjunction, of the systems of constraints) is a system of constraints of $P \cap P'$. Of course, this conjunction can result in redundant constraints.

Convex hull: The convex hull is used as an approximation of union, which is not an internal operation on polyhedra (it does not preserve convexity). The convex hull of two polyhedra is the least convex polyhedron which contains

both of them. Let $(V, R), (V', R')$ be the respective systems of generators of P_1 and P_2 , then $(V \cup V', R \cup R')$ is a system of generators of their convex hull, noted $P \sqcup P'$. Again, this system of generators is generally not minimal.

Affine transformation: An affine transformation in \mathcal{N}^n is given by a pair (C, D) , where C is a $n \times n$ matrix and D is a n -vector. The image of a polyhedron P by an affine transformation (C, D) is

$$CP + D = \{CX + D \mid X \in P\}$$

Then, if (V, R) is a system of generators of P , $(CV + D, CR)$ is a system of generators of $CP + D$. When C is invertible, the transformation can also be done on P 's system of constraints, and the affine transformation preserves the minimality of both descriptions.

Existential quantification: We note $\exists x_j, P$ the results of eliminating the j -th variable in P by the *Fourier-Motzkin procedure*: it consists of replacing each constraint $A_i X \leq B$ where $A_{ij} > 0$ by its positive combinations with all constraints $A_k X \leq B$ where $A_{kj} < 0$ to get a null coefficient for X_j (i.e., $(A_{ij} \cdot A_k - A_{kj} \cdot A_i)X \leq (A_{ij} \cdot B_k - A_{kj} \cdot B_i)$). Existential quantification of x_j can be easily done, also, on systems of generators, simply by adding u_j and $-u_j$ as new rays, where u_j is the j th unit vector of \mathcal{N}^n .

Test for inclusion: Let (V, R) be a system of generators of P , and (A, B) be a system of constraints of P' . Then $P \subseteq P'$ if and only if $Av \leq B$, for all $v \in V$, and $Ar \leq 0$, for all $r \in R$.

Test for emptiness: A polyhedron is empty if and only if its set of vertices is empty.

Widening: The widening is used to extrapolate the limit of an increasing sequence of polyhedra. There is some freedom in defining such an operator, we use the one proposed in [Hal79]. Intuitively, the system of constraints of the widening $P \nabla P'$ is the subset of P 's constraints still satisfied by P' . Now, since this definition relies on the system of constraints of P , which is not canonical in general, the actual definition is more complex: a constraint of $P \nabla P'$ is a constraint of P' which is *mutually redundant*⁵ with a constraint of P (meaning that either it is a constraint of P , or it can *replace* a constraint of P without changing it). Here again, both descriptions are used to perform this operation.

3 Factoring of Polyhedra

Let I be a subset of $\{1 \dots n\}$. We note $P \downarrow I$ the projection of the polyhedron P on variables with indices in I (i.e., the result, in $\mathcal{N}^{|I|}$ of the existential quantification of all variables with indices outside I).

Let $(I_1, I_2, \dots, I_\ell)$ be a partition of $\{1 \dots n\}$. We say that a polyhedron P can be *factored according to* $(I_1, I_2, \dots, I_\ell)$ if and only if

⁵ Two constraints are mutually redundant in a system of constraints if you can replace each of them by the other without changing the set of solutions.

$$P = P \downarrow I_1 \times P \downarrow I_2 \times \dots \times P \downarrow I_\ell$$

A matrix A is *block-diagonalizable* according to a partition $(I_1, I_2, \dots, I_\ell)$ if for each of its row A_i there is one $k_i \in \{1.. \ell\}$ such that $\{j \mid A_i^j \neq 0\} \subseteq I_{k_i}$.

Some obvious facts:

- f1. for any polyhedron P , there is a greatest partition $(I_1, I_2, \dots, I_\ell)$ according to which P can be factored (possibly the trivial partition, with $\ell = 1$).
- f2. for any matrix A , there is a greatest partition $(I_1, I_2, \dots, I_\ell)$ according to which A is block-diagonalizable (possibly the trivial partition, with $\ell = 1$).
- f3. if (A, B) is the constraint description of a polyhedron P , and if A is block-diagonalizable according to a partition $(I_1, I_2, \dots, I_\ell)$, then P can be factored according to $(I_1, I_2, \dots, I_\ell)$ (the converse is not true, if (A, B) is not minimal). This gives an easy way to factor a polyhedron, and to get the constraint descriptions of its factors: each constraint (A_i, B_i) becomes a constraint of the factor P_{k_i} .
- f4. For any pair (P, P') of polyhedra (resp., for any pair (A, A') of matrices) there is a greatest common partition (possibly the trivial partition) according to which both polyhedra can be factored (resp., both matrices are block-diagonalizable).
- f5. Conversely, given a description of the factors P_1, \dots, P_ℓ , one can easily obtain the corresponding description of $P = P_1 \times \dots \times P_\ell$:
 - its system of constraints is just the conjunction of those of the factors;
 - its system of generators is obtained by composing together all the ℓ -tuples of vertices (resp., of rays) of the factors. This composition explains the explosion of the size of the systems of generators, since $|V| = \prod_{k=1}^{\ell} |V_k|$ and $|R| = \prod_{k=1}^{\ell} |R_k|$.
 A similar treatment works also to obtain a description of P factored according to any partition rougher than $(I_1, I_2, \dots, I_\ell)$.

Example: Fig. 2 shows a factored polyhedron in 2 dimensions. In its minimal system of constraints:

$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} -1 \\ 2 \\ 0 \end{pmatrix}$$

the matrix is block-diagonal. Now, if the redundant constraint $2x \geq y$ is added, the matrix is non longer block-diagonalizable, and the factoring of the polyhedron is hidden.

4 Easy Operations

Most of the operations mentioned in Section 2.2 can be easily applied componentwise to factored polyhedra. The operands need first to be factored in the

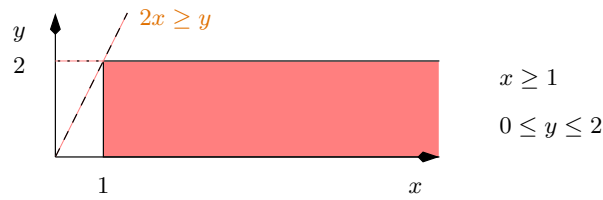


Fig. 2. A factored polyhedron.

same way (using f_4 and f_5 above). Moreover, in many cases, the result may be better factored than the operands, which can be done using f_2 .

Intersection. If P and P' are factored according to the same partition $(I_1, I_2, \dots, I_\ell)$, then so is $P \cap P' = P_1 \cap P'_1 \times P_2 \cap P'_2 \times \dots \times P_\ell \cap P'_\ell$. It may be the case that $P \cap P'$ can be further factored (Fig. 3).

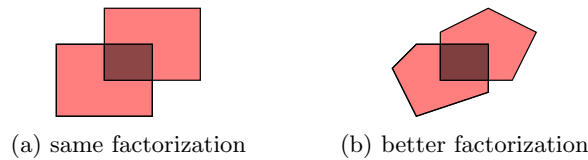


Fig. 3. Intersection of factored polyhedra.

Affine transformation. Let $X \mapsto CX + D$ be an affine transformation. If C is block-diagonalizable according to $(I_1, I_2, \dots, I_\ell)$, and P is factored according to the same partition, then so is $CP + D = C_{I_1}P_1 + D_{I_1} \times \dots \times C_{I_\ell}P_\ell + D_{I_\ell}$. If C is not invertible, it can be the case that $CP + D$ can be further factored.

Widening. If P and P' are factored according to the same partition $(I_1, I_2, \dots, I_\ell)$, then so is $P \nabla P' = P_1 \nabla P'_1 \times P_2 \nabla P'_2 \times \dots \times P_\ell \nabla P'_\ell$. It may be the case (in fact, it happens very often) that $P \nabla P'$ can be further factored.

Emptiness and inclusion. Let $P = P_1 \times P_2 \times \dots \times P_\ell$. Then P is empty if and only if there exists $k \in \{1.. \ell\}$ such that P_k is empty. If P and P' are factored according to the same partition $(I_1, I_2, \dots, I_\ell)$, then $P \subseteq P'$ if and only if, for all $k \in \{1.. \ell\}$, $P_k \subseteq P'_k$.

5 The Convex Hull

The computation of the convex hull is more difficult. The convex hull of two factored polyhedra can be either less factored (Fig. 4.a) or as factored (Fig. 4.b), or even more factored (Fig. 4.c) than the operands.

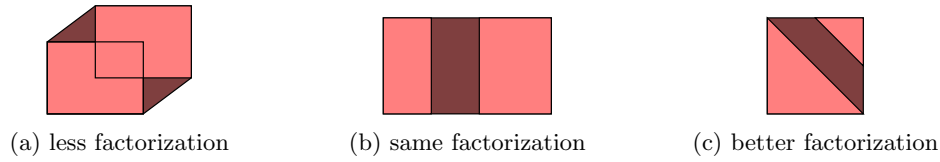


Fig. 4. Convex hull of factored polyhedra.

The goal is to get the factored result, when possible, in a decomposed way, and without penalizing the computation when the result is not factored. This can be achieved thanks to the following proposition:

Proposition 1. Let $P = P_1 \times P_2$ and $P' = P'_1 \times P'_2$ be two polyhedra factored according to the same partition. Let λ be a fresh variable and let us consider the polyhedra Q_1 and Q_2 defined by:

$$Q_1 = (P_1 \wedge \{\lambda = 0\}) \sqcup (P'_1 \wedge \{\lambda = 1\}) \quad Q_2 = (P_2 \wedge \{\lambda = 0\}) \sqcup (P'_2 \wedge \{\lambda = 1\})$$

Then,

- if λ is lower-bounded by a non constant expression in Q_1 — i.e., there is a constraint $E(X_1) \leq \lambda$ in the system of constraints of Q_1 , where $E(X_1)$ is a non constant linear expression of X_1 — and upper-bounded by a non constant expression in Q_2 , or conversely, the convex hull $P \sqcup P'$ is non longer factored, and can be computed as $\exists \lambda, Q_1 \wedge Q_2$;
- otherwise $P \sqcup P' = (\exists \lambda, Q_1) \times (\exists \lambda, Q_2)$.

Proof: By definition of the convex hull,

$$\begin{aligned} X \in P \sqcup P' &\Leftrightarrow \exists Y \in P, Y' \in P', \lambda \in [0, 1], \text{ such that } X = \lambda Y + (1 - \lambda)Y' \\ &\Leftrightarrow X = (X_1, X_2) \wedge \exists \lambda \in [0, 1] \text{ such that} \\ &\quad \exists Y_1 \in P_1, Y'_1 \in P'_1, X_1 = \lambda Y_1 + (1 - \lambda)Y'_1 \wedge \\ &\quad \exists Y_2 \in P_2, Y'_2 \in P'_2, X_2 = \lambda Y_2 + (1 - \lambda)Y'_2 \\ &\Leftrightarrow X = (X_1, X_2) \wedge \exists \lambda \in [0, 1] \text{ such that} \\ &\quad (X_1, \lambda) \in Q_1 \wedge (X_2, \lambda) \in Q_2 \end{aligned}$$

Now, the existential quantification of λ in the last system of constraints can only produce dependencies between previously independent variables in two cases:

- if there is some constraint $E(X_1) \leq \lambda$ in Q_1 , and some constraint $\lambda \leq F(X_2)$ in Q_2 — which will produce $E(X_1) \leq F(X_2)$;
- or, conversely, if there is some constraint $\lambda \leq E(X_1)$ in Q_1 , and some constraint $F(X_2) \leq \lambda$ in Q_2 — which will produce $F(X_2) \leq E(X_1)$;

where $E(X_1)$ and $F(X_2)$ are non constant expressions. Otherwise, λ can be quantified separately in Q_1 and Q_2 .

□

This result generalizes to multiple factorings:

Proposition. Let $P = P_1 \times \dots \times P_\ell$ and $P' = P'_1 \times \dots \times P'_\ell$ be two polyhedra factored according to the same partition. Let λ be a fresh variable and let us consider the polyhedra $(Q_k)_{k=1..l}$ defined by:

$$Q_k = (P_k \wedge \{\lambda = 0\}) \sqcup (P'_k \wedge \{\lambda = 1\})$$

Then, the partition of $P \sqcup P'$ is obtained from (I_1, \dots, I_ℓ) by merging I_k and $I_{k'}$ whenever either λ is lower-bounded by a non constant expression in Q_k and upper-bounded by a non constant expression in $Q_{k'}$, or conversely. Let (J_1, \dots, J_h) be the resulting partition, each J_m being a union of some I_k s. Then

$$P \sqcup Q = R_1 \times R_2 \times \dots \times R_h \text{ where } R_m = \exists \lambda, \prod_{I_k \subseteq J_m} Q_k$$

6 Experimental Results

6.1 Operations on factored hypercubes

We first compared the performances of the operations in the best case, where operands are completely factored hypercubes (i.e., in fact, intervals). For intersection and widening⁶, the results are, of course, excellent:

Intersection	Classical operations				Factored operations				
Dimension	10	11	12	13	10	20	30	40	50
Operation time	3"	13"	60"	4'36	0"02	0"03	0"03	0"04	0"07

Widening	Classical operations					Factored operations				
Dimension	8	10	12	14	16	10	20	30	40	50
Operation time	0"05	0"22	1"68	20"63	6'58"	0"01	0"02	0"04	0"06	0"07

Concerning the convex-hull, we made two series of experiments:

- the first one concerns the best case, where the result is still fully factored. The following table only gives the times of the classical convex hull, since the factored one gives non measurable times (less than 0.01 sec. in dimension 50): in dimension n , it performs n convex hulls of polyhedra with 2 variables (the current variable and the variable λ of Prop. 1).

Dimension	10	11	12	13	14	15	16
Classical	0"06	0"28	2"01	12"07	54"99	3'44"	15'27"

- the second series concerns the worst case, where the result is not factored at all:

⁶ Operands are such that half of the constraints are kept by the widening.

Dimensions	10	11	12	13	14	15	16
Classical	0"09	0"48	3"81	21"89	1'35"	6'29"	25'54"
Factored	0"17	0"41	2"93	5"76	1'20"	5'50"	22'53"

Notice that, even in this case, the factored operation is still better than the classical one. In addition with the same operations than in the previous case, it has to perform an existential quantification of λ , which appears to dominate largely in the overall cost.

6.2 Experiments in program analysis

Real experiments in program analysis are more difficult to conduct, for the following reason: our verification tool, NBAC [JHR99] performs much more than a simple analysis, since it deals with successive forward and backward analyses together with slicing and dynamic partitioning. As a consequence, the cost of polyhedra operations is not always prominent. Our experiments concerned parameterized problems, where the number of numeric variables can be easily increased without increasing the rest of the treatment. Let us show a few results:

Problem	car			
Dimension	4	5	6	7
Time without factoring	1"63	4"74	7"79	55'31"
Time with factoring	3"84	10"48	16"09	2'55"

Problem	Ntoggle			
Dimension	4	5	6	7
Time without factoring	1"39	2"01	3"40	44'51"
Time with factoring	3"21	4"60	7"71	16"32

Of course, these results should be taken with care, and further experiments must be conducted. However, all our experiments show that the factored version outperforms the classical one from a quite small dimension (less than 10, in general). We will start experiments in a completely different context, by linking the analyzer of [DRS03] with our library with factored operations.

7 Conclusion and Future Works

We presented an adaptation of classical operations on polyhedra to work on Cartesian products of polyhedra. This adaptation is straightforward, except in the case of the convex hull. The gain in performance is obvious — and spectacular — when the arguments of the operations are factored in the same way. So the interest of the approach relies on the fact that, during an analysis, polyhedra are indeed factored in similar ways. It appears to be often the case, at least in the end of an analysis, after the application of the widening, which tends to suppress dependences between variables.

This work can be pursued in several directions. First, the fact that polyhedra are factored depends on the choice of variables. A simple variable change (e.g., a rotation) can make possible a factoring. The question is whether the detection of this fact can be done efficiently enough to make it worthwhile. An other improvement concerns the beginning of the analysis: as mentioned before, factoring is more likely to occur at the end, after one or several widenings. Now, at the beginning of an analysis, variables are tightly dependent; in fact, they are likely to satisfy linear equations. This suggests to look for an other way of saving variables, by early detection of these equations [Kar76] and elimination of variables. Of course, since each equation allows only one variable to be removed, this technique is less likely to produce spectacular improvements than polyhedra factoring.

References

- [BBC⁺00] N. Bjorner, A. Browne, M. Colon, B. Finkbeiner, Z. Manna, H. Sipma, and T. Uribe. Verifying temporal properties of reactive systems: A STEP tutorial. *Formal Methods in System Design*, 16:227–270, 2000.
- [BBM97] N. Bjorner, I. Anca Browne, and Z. Manna. Automatic generation of invariants and intermediate assertions. *Theoretical Computer Science*, 173(1):49–87, February 1997.
- [BRZH02] R. Bagnara, E. Ricci, E. Zaffanella, and P. M. Hill. Possibly not closed convex polyhedra and the parma polyhedra library. In M. V. Hermenegildo and G. Puebla, editors, *9th International Symposium on Static Analysis, SAS'02*, Madrid, Spain, September 2002. LNCS 2477.
- [CC77] P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fix-points. In *4th ACM Symposium on Principles of Programming Languages, POPL'77*, Los Angeles, January 1977.
- [CH78] P. Cousot and N. Halbwachs. Automatic discovery of linear restraints among variables of a program. In *5th ACM Symposium on Principles of Programming Languages, POPL'78*, Tucson (Arizona), January 1978.
- [Che68] N. V. Chernikova. Algorithm for discovering the set of all solutions of a linear programming problem. *U.S.S.R. Computational Mathematics and Mathematical Physics*, 8(6):282–293, 1968.
- [CL98] Ph. Clauss and V. Loechner. Parametric analysis of polyhedral iteration spaces. *Journal of VLSI Signal Processing*, 19(2), July 1998.
- [DRS01] N. Dor, M. Rodeh, and M. Sagiv. Cleaness checking of string manipulations in C programs via integer analysis. In P. Cousot, editor, *SAS'01*, Paris, July 2001. LNCS 2126.
- [DRS03] N. Dor, M. Rodeh, and M. Sagiv. CCSV: towards a realistic tool for statically detecting all buffer overflows in C. to appear in PLDI03, 2003.
- [Hal79] N. Halbwachs. Détermination automatique de relations linéaires vérifiées par les variables d'un programme. Thèse de troisième cycle, University of Grenoble, March 1979.
- [HHWT97] T. A. Henzinger, P.-H. Ho, and H. Wong-Toi. Hytech: A model checker for hybrid systems. *Software Tools for Technology Transfer*, 1:110–122, 1997.

- [HPR97] N. Halbwachs, Y.E. Proy, and P. Roumanoff. Verification of real-time systems using linear relation analysis. *Formal Methods in System Design*, 11(2):157–185, August 1997.
- [IJT91] F. Irigoin, P. Jouvelot, and R. Triolet. Semantical interprocedural parallelization: An overview of the PIPS project. In *ACM Int. Conf. on Supercomputing, ICS'91, Köln*, 1991.
- [JHR99] B. Jeannet, N. Halbwachs, and P. Raymond. Dynamic partitioning in analyses of numerical properties. In A. Cortesi and G. Filé, editors, *Static Analysis Symposium, SAS'99*, Venice (Italy), September 1999. LNCS 1694, Springer Verlag.
- [Kar76] M. Karr. Affine relationships among variables of a program. *Acta Informatica*, 6:133–151, 1976.
- [LeV92] H. LeVerge. A note on Chernikova's algorithm. RR. 635, IRISA, February 1992.
- [MRTT53] T. S. Motzkin, H. Raiffa, G. L. Thompson, and R. M. Thrall. The double description method. In H. W. Kuhn and A. W. Tucker, editors, *Contribution to the Theory of Games - Volume II*. Annals of Mathematic Studies, nr 28, Princeton University Press, 1953.
- [Tip95] F. Tip. A survey of program slicing techniques. *Journal of Programming Languages*, 3(3):121–189, September 1995.
- [Wil93] D. K. Wilde. A library for doing polyhedral operations. RR. 785, IRISA, December 1993.