Conference Record of the Fifth Annual ACM Symposium on Principles of Programming Languages

AUTOMATIC DISCOVERY OF LINEAR RESTRAINTS AMONG VARIABLES OF A PROGRAM

Patrick Cousot* and Nicolas Halbwachs**

Laboratoire d'Informatique, U.S.M.G., BP. 53
38041 Grenoble cédex, France

1. INTRODUCTION

The model of abstract interpretation of programs developped by Cousot[1976], Cousot[1977] is applied to the static determination of linear equality or inequality relations among variables of programs.

For example, consider the following sorting procedure (Knuth[1973], p.107) :

```
     procedure BUBBLESORT(integer value N;
                              integer array[1:N] K);
     begin integer B,J,T;
{1}    B:=N;
{2}    while B≥1 do
{3}      J:=1; T:=0;
{4}      while J≤(B-1) do
{5}        if K[J]>K[J+1] then
             EXCHANGE(J,J+1); {no side effects on
{6}                                     N,B,J,T}
{7}          T:=J,
{8}        fi;
{9}        J:=J+1;
{10}     od;
{11}     if T=0 then return fi;
{12}     B:=T;
{13}   od;
     end;
```

Without user provided inductive assertions nor human interaction we have automatically determined (in 1.582 seconds of C.P.U. time) that the following restraints must hold among the variables of the above procedure:

{1}	: B=N
{2}	: 1≤B≤N
{3}	: 1≤B≤N, J=1, T=0
{4},{5},{6}	: B≤N, T≥0, T+1≤J, J+1≤B
{7}	: B≤N, J≥1, J+1≤B, J=T
{8}	: B≤N, J+1≤B, J≥1, T≥0, T≤J
{9}	: B≤N, J≤B, J≥2, T≥0, T+1≤J
{10},{11}	: B≤N, J≤B, T≥0, T+1≤J, B≤J+1
{12}	: J≤N, T≥0, T+1≤J, T=B
{13}	: B≤N, B≤1

---

* Attaché de Recherche au C.N.R.S., Laboratoire Associé n°7

** Allocataire de Recherche D.G.R.S.T.

This work was supported by C.N.R.S. under grant ATP-Informatique D3119.

A certain number of classical data flow analysis techniques are included in or generalized by the determination of linear equality relations among program variables. For example constant propagation can be understood as the discovery of very simple linear equality relations among variables (such as X=1, Y=5). However the resolution of the more general problem of determining linear equality relations among variables allows the discovery of symbolic constants (such as X=N, Y=5*N+1). The same way, common subexpressions can be recognized which are not formally identical but are semantically equivalent because of the relationships among variables. Also the loop invariant computations as well as loop induction variables (modified inside the loop by the same loop invariant quantity) can be determined on a basis which is not purely syntactical. The problem of discovering linear equality relations is in fact a particular case of the one of discovering linear inequality relations among the program variables. The main use of these inequality relationships is to determine at compile time whether the value of an expression is within a specified numeric or symbolic subrange of the integers or reals. This includes compile time overflow, integer subrange and array bound checking. In contrast to Suzuki-Ishihata[1977] we do not simply try to verify the legality assertions (such as verifying that array subscripts are within the declared range) but instead we try to discover the assertions (of linear type) that can be deduced from the semantics of the program. The advantage is that we can often discover relations which are never stated explicitely in the program. For example, we can discover that an integer variable lies in a subrange of its declared range or that two references A[I] and A[J] to two elements of the same array refer to different storage locations (since e.g. I≥J+1) or that some piece of code is dead.

The problem of determining equality relationships between a linear combination of the variables of the program and a constant was solved by Karr [1976]. His approach was based on Wegbreit[1975]'s algorithm which requires that the properties to be discovered form a lattice every strictly increasing chain of which is of finite length. This assumption is not valid when considering inequality relationships (because of chains such as (x=1), (1≤x≤2),..., (1≤x≤n),...).

The model of Cousot [1977] is general enough to cope with this problem and we briefly recall it in section 2 as formulated in Cousot [1976]. In section 3 we study formal representations for the particular type of assertions that we consider. In section 4 we describe the linear restraints transformer corresponding to elementary instructions of the language. The algorithm performing the global analysis of programs is presented in section 5 by means of simple examples. Section 6 gives more convincing examples and Section 7 discusses the experimental implementation that has been realized.

2. APPROXIMATE ANALYSIS OF PROGRAM PROPERTIES, Cousot [1976].

For purposes of exposition, a sequential program will be represented by a connected finite flowchart with one entry node and assignment, test, junction and exit nodes. The evaluations of the right-hand side of an assignment and of the boolean expression in a test node are assumed not to affect the values of any variables. Thus all side-effect phenomena must be modeled as assignment statements. The junction nodes contain no computations and represent the merge of program execution paths.

The analysis of a program consists in attaching an assertion $P_i(V_1,\dots,V_n)$ to each arc i of the program. These predicates on the variables $V_1,\dots,V_n$ are not necessarily of the most general form but instead are designed to model a specific aspect of the semantic properties of the program.

The assertion on the entry arc to the program represents what is known about the variables at the start of execution. For each other type of program node a transformation specifies the assertion associated with the output arc(s) of the node in terms of the assertions on the input arc(s) to the node and where relevant the content of the node. Hence we have established a system of equations between the local assertions which can be solved by successive approximations (Cousot [1977]). Starting from no information on each arc, this resolution consists in propagating the entry assertion around the program graph by application of the transformation described for each type of node until stabilization. All possible paths are followed pseudo-parallely with synchronization on junction nodes. It is assumed that each cycle in the program graph contains a specially marked junction node which is called a loop junction node. In order to ensure stabilization the transformation for a loop junction node will be an approximation of the transformation for ordinary junction nodes. More precisely let $P_{i_1 j}, \dots, P_{i_m j}$ be the assertions associated with the input arcs of a junction node at step j. Let Q_j be the assertion associated with the output arc of this node. Then $Q_j = f(P_{i_1 j}, \dots, P_{i_m j})$. For a loop junction node we have $Q_j = Q_{j-1}\ \nabla_j\ f(P_{i_1 j}, \dots, P_{i_m j})$ where the *widening* operation ∇_j performed at step j is such that $\{Q\ \nabla_j P \Rightarrow Q\}$ and $\{Q\ \nabla_j P \Rightarrow P\}$ and for any chain $\{P_0 \Rightarrow P_1 \Rightarrow \dots \Rightarrow P_j \Rightarrow \dots\}$ the chain $Q_0 = P_0$, $Q_1 = Q_0 \nabla_1 P_1, \dots, Q_j = Q_{j-1} \nabla_j P_j, \dots$ is not an infinite strictly increasing chain.

In the examples we shall use the algorithm described loosely above and in more detail in Cousot [1976].

3. FORMAL REPRESENTATIONS OF LINEAR RESTRAINTS AMONG VARIABLES OF A PROGRAM.

3.1. Linear system of a convex polyhedron.

Let $x^1, \dots, x^n$ be the variables of the program. For simplicity we assume that the values of the variables belong to the set $\mathbb{R}$ of reals. The set of solutions to a system of linear equations $\{\Sigma_{i=1}^{n}(a_i^j x^i) = b^j : j=1..m\}$ (where $a_i^j, b^j \in \mathbb{R}$), if such solutions exist, is a *linear variety* of $\mathbb{R}^n$. A linear variety of dimension n-1 is an *hyperplane*. The set of solutions to a linear inequality $\Sigma_{i=1}^{n}(ax^i) \le b$ is a *closed half-space* of $\mathbb{R}^n$. For simplicity, strict inequalities are not considered. By *linear restraint* we mean either a linear equality or a linear inequality. In the formal reasoning we often consider that an equation can be viewed as two opposite inequalities.

A subset C of $\mathbb{R}^n$ is said to be *convex* if and only if $\{\forall x_1, x_2 \in C, \forall \lambda \in [0,1], \lambda x_1 + (1-\lambda) x_2 \in C\}$. For example linear varieties and half-spaces are convex. The intersection of two convex sets is convex, but the union of two convex sets is not necessarily convex.

The set of solutions to a finite system of linear inequalities can be interpreted geometrically as the *closed convex polyhedron* of $\mathbb{R}^n$ defined by the intersection of the closed halfspaces corresponding to each inequality.

3.2. The frame of a convex polyhedron, Weyl [1950], Klee [1959], Charnes [1953].

Let $V_1, \dots, V_p$ be vectors in $\mathbb{R}^n$. A vector of the form $\Sigma_{i=1}^{p}(\nu_i V_i)$ where for each i=1..p we have $\nu_i \in \mathbb{R}$ is called a *linear combination* of the V_i. The set of all linear combinations of the V_i is the *linear variety generated by the* V_i. A *basis* of a linear variety L is a minimal set of vectors generating L.

A vector of the form $\Sigma_{i=1}^{p}(\mu_i V_i)$ where for each i=1..p we have $\mu_i \in \mathbb{R}^+ = \{\mu \in \mathbb{R} : \mu \ge 0\}$ is called a *positive combination* of the V_i. The set of all positive combinations of the V_i is the *conical hull* of the V_i.

A vector of the form $\Sigma_{i=1}^{p}(\lambda_i V_i)$ where for each i=1..p we have $\lambda_i \in \mathbb{R}^+$ with $\Sigma_{i=1}^{p}(\lambda_i) = 1$ is called a *convex combination* of the V_i. The set of all convex combinations of the V_i is the *convex hull* of the V_i.

A *vertex* of a polyhedron P is a point x of P which is not a convex combination of other points of P: $\{(x = \Sigma_{i=1}^{p} \lambda_i x^i)$ and $(\forall i=1..p, x^i \in P$ and $\lambda_i \ge 0)$ and $\Sigma_{i=1}^{p}(\lambda_i) = 1\}$ implies $\{\forall i=1..p, (\lambda_i = 0$ or $x^i = x)\}$.

A *ray* of a polyhedron P is a vector r such that there exists an half-line parallel to r and entirely included in P : $\{\forall x \in P, \forall \mu \in \mathbb{R}^+, x + \mu r \in P\}$. Two rays r and r' are equal if and only if $\{\exists \mu \in \mathbb{R}^+ : r = \mu r'\}$.

A ray of a polyhedron P is called an *extreme ray* if and only if it is not a positive combination of other rays $r_1, \dots, r_q$ of P: $\{r = \Sigma_{j=1}^{q}(\mu_j r_j)$ and $(\forall j=1..q, \mu_j \in \mathbb{R}^+)\}$ implies $\{\forall j=1..q, (\mu_j = 0)$ or $(r_j = r)\}$. Intuitively, a ray (extreme ray) of a polyhedron is a point (vertex) translated to infinity.

A *line* of a polyhedron P is a vector d such that both d and -d are rays of P: $\{\forall x \in P, \forall \mu \in \mathbb{R}, x+\mu d \in P\}$. A polyhedron which contains at least one line is called a *cylinder*. The linear variety generated by all the lines of a cylinder is the greatest linear variety included in the cylinder. A polyhedron that contains no line has only a finite number of vertices and of extreme rays.

A *bounded polyhedron* has neither lines nor rays. Each point of a bounded polyhedron is a convex combination of the vertices of the polyhedron so that a bounded polyhedron is the convex-hull of its vertices.

Each point x of a polyhedron P which is not a cylinder can be expressed as the sum of a convex combination of the vertices $\{s_1, \ldots, s_\sigma\}$ of P and of a positive combination of the extreme rays $\{r_1, \ldots, r_\rho\}$ of P:
$\exists \lambda_1, \ldots, \lambda_\sigma \in [0,1], \exists \mu_1, \ldots, \mu_\rho \in \mathbb{R}^+ : \Sigma_{i=1}^{\sigma}(\lambda_i)=1$ and $x=\Sigma_{i=1}^{\sigma}(\lambda_i s_i) + \Sigma_{j=}^{\rho}(\mu_j r_j)$.

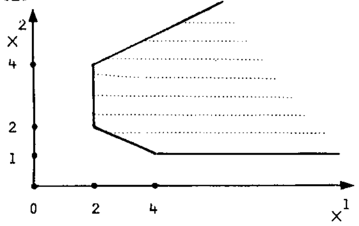
Let L be the greatest linear variety included in a cylinder P. Let L' be a linear variety orthogonal to L. Then the intersection of L' with P is a convex polyhedron which contains no line and which is called a *section* of P. Each point of a cylinder can be expressed as the sum of a convex combination of the vertices of a section of P, a positive combination of the extreme rays of this section and a linear combination of the vectors of a basis of the greatest linear variety included in P.

By misuse of words the vertices and rays of a cylinder will be the vertices and rays of a section of that cylinder.

A closed convex polyhedron P can be characterized by three sets $S = \{s_1, \ldots, s_\sigma\}$, $R = \{r_1, \ldots, r_\rho\}$, $D = \{d_1, \ldots, d_\delta\}$ of vectors of $\mathbb{R}^n$ called the *frame* of the polyhedron as follows:
$\{x \in P\} \Leftrightarrow \{\exists \lambda_1, \ldots, \lambda_\sigma \in [0,1], \exists \mu_1, \ldots, \mu_\rho \in \mathbb{R}^+, \exists \nu_1, \ldots, \nu_\delta \in \mathbb{R} : \Sigma_{i=1}^{\sigma}(\lambda_i)=1$ and $x = \Sigma_{i=1}^{\sigma}(\lambda_i s_i) + \Sigma_{j=1}^{\rho}(\mu_j r_j) + \Sigma_{k=1}^{\delta}(\nu_k d_k)\}$.

We have two equivalent representations of a closed convex polyhedron either as the set of solutions of its system of linear restraints or as the convex hull of its frame.

Example: The polyhedron defined by the following system of restraints: $\{x^1 \geq 2, x^2 \geq 1, x^1+2x^2 \geq 6, x^1-2x^2 \geq -6\}$ is spanned by the following frame (see the diagram)
vertices: $s_1=\binom{4}{1}$, $s_2=\binom{2}{2}$, $s_3=\binom{2}{4}$
extreme rays: $r_1=\binom{1}{0}$, $r_2=\binom{2}{1}$
no line.



End of example.

Before looking at the problem of conversion between these representations we need some definitions.

A *face* of a polyhedron P is a convex polyhedron $F \subseteq P$ such that if a point x lies in F then each segment included in P and containing x is included in F: $\{x \in F, p,q \in P, \lambda \in [0,1]$ and $x=\lambda p+(1-\lambda)q\} \Rightarrow \{p \in F$ and $q \in F\}$.

The *dimension* of a polyhedron P is the dimension of the least linear variety containing P.

A face of dimension k is called a *k-face*. An *edge* is a 1-face. The vertices of a polyhedron containing no line are its 0-faces (thus a cylinder has no 0-face). Let $\Sigma_{i=1}^{n}(a_i x^i) \leq b^i$ be a linear inequality defining the polyhedron P. Then if the intersection of the hyperplane defined by H = $\{x : \Sigma_{i=1}^{n}(a_i x^i)=b^i\}$ with P is not empty, it is a face of P.

We say that a point s *saturates* the inequality $\Sigma_{i=1}^{n}(a_i x^i) \leq b^i$ if and only if $\Sigma_{i=1}^{n}(a_i s^i) = b^i$. We say that the ray r *saturates* this inequality if and only if $\Sigma_{i=1}^{n}(a_i r^i)=0$.

Let δ be the dimension of the greatest linear variety included in the polyhedron P. Two vertices are said to be *adjacent* iff they lie on the same edge (i.e. if $\delta \neq 0$ we mean the edge of the section). It follows that the number of inequalities saturated at the same time by two adjacent vertices is at least $n-\delta-1$.

Two extreme rays are said to be *adjacent* iff they belong to the same 2-face. Therefore they saturate at least $n-\delta-2$ inequalities simultaneously.

Finally, a vertex s and a ray r are *adjacent* iff s lies on an infinite edge which is parallel to r. Therefore s must saturate all the inequalities satured by r.

3.3. Conversions between the representations of a polyhedron by a system of linear restraints and by a frame.

Some operations that we have to perform on closed convex polyhedra are easy when these polyhedra are represented by systems of linear restraints while others are more simple when the frame representation is used. Hence we must be able to make conversions from one representation to the other.

3.3.1. Conversion from the frame to the linear restraints representation.

This conversion consists in finding a system of restraints for the convex-hull of the elements (points, rays, lines) of a frame. This conversion is followed by a simplification of the system of restraints.

3.3.1.1. Convex-hull of a finite frame.

Let $S=\{s_1, \ldots, s_\sigma\}$, $R=\{r_1,\ldots,r_\rho\}$, $D=\{d_1,\ldots,d_\delta\}$ be the frame of a non-empty polyhedron (so that $S \neq \emptyset$). The points x of this polyhedron are characterized by the existence of $\lambda_1, \ldots, \lambda_\sigma, \mu_1, \ldots, \mu_\rho, \nu_1, \ldots, \nu_\delta \in \mathbb{R}$ such that:

- $0 \leq \lambda_i \leq 1$ for $i=1,\ldots,\sigma$
- $\Sigma_{i=1}^{\sigma}(\lambda_i) = 1$
- $0 \leq \mu_j$ for $j=1,\ldots,\rho$
- $x = \Sigma_{i=1}^{\sigma}(\lambda_i s_i) + \Sigma_{j=1}^{\rho}(\mu_j r_j) + \Sigma_{k=1}^{\delta}(\nu_k d_k)$

Hence this polyhedron is characterized by a system of linear restraints in $\mathbb{R}^{n+\sigma+\rho+\delta}$ upon the variables x_l $(l=1..n)$, $\lambda_i(i=1..\sigma)$, μ_j $(j=1..\delta)$, $\nu_k(k=1..\delta)$. Eliminating the λ_i,μ_j,ν_k we get a system of linear restraints in $\mathbb{R}^n$. This elimination can always be done by the *projection* operation used in Kuhn[1956].

Let us represent the system of m inequalities by $AX\leq B$ where A is a mxn real matrix and B a m-vector. In order to eliminate the variable X_c we project according to the column c upon $AX\leq B$ in order to get a simplified projected system defined as follows:

- For each row A^l such $A^l_c=0$, the restraint $A^lX\leq B^l$ is part of the projected system.
- For each two rows A^{l_1} and A^{l_2} such that $A^{l_1}_c.A^{l_2}_c<0$ the restraint $(|A^{l_1}_c|.A^{l_2}+|A^{l_2}_c|.A^{l_1})X\leq |A^{l_1}_c|.B^{l_2}+|A^{l_2}_c|.B^{l_1}$ is part of the projected system.

The projected system contains only zeros in the column c, hence it is independent of X_c.

Example: Eliminating λ in the system

$$\begin{bmatrix} & -\lambda \leq 0 \\ & \lambda \leq 1 \\ -x_3 & \leq 0 \\ x_1-x_2 & -2\lambda \leq 0 \\ -x_1+x_2 & +4\lambda \leq 2 \end{bmatrix} \quad \text{we get} \quad \begin{bmatrix} -x_3 \leq 0 \\ -x_1+x_2 \leq 2 \\ x_1-x_2 \leq 2 \end{bmatrix}$$

End of example

The convex-hull of the frame S, R, D is computed by successive approximations $P_1, P_2, \ldots, P_{\sigma+\rho+\delta}$ as follows:

- P_1 is the point s_1 so that the corresponding system of restraints is $(x=s_1)$. Then for each $i=2..\sigma$, P_i is the convex-hull of P_{i-1} and the point s_i. Therefore $\{x\in P_i\}$ if and only if $\{\exists x_1\in P_{i-1}, \exists\lambda\in[0,1] : x=\lambda x_1+(1-\lambda)s_i\}$. If $AX\leq B$ is the system of restraints describing P_{i-1} it can be shown that this is equivalent to $\{\exists\lambda\in\mathbb{R} : 0\leq\lambda\leq 1$ and $AX+\lambda(As_i-B)\leq As_i\}$ Then by a projection according to the column of λ, λ can be eliminated to get a system of restraints of P_i.
- For each $j=\sigma+1,\ldots,\sigma+\rho$, P_j is obtained by adjoining the ray $r_{j-\sigma}$ to P_{j-1}. If P_{j-1} corresponds to $AX\leq X$, this consists in eliminating μ in $(\mu\geq 0$, $AX-\mu Ar_{j-\sigma}\leq B)$.
- For each $k=\sigma+\rho+1,\ldots,\sigma+\rho+\delta$, P_k is built from P_{k-1} by adjonction of the line $d_{k-(\sigma+\rho)}$. If P_{k-1} is described by $AX\leq B$, this consists in eliminating ν in $(AX-\nu Ad_{k-(\sigma+\rho)}\leq B)$. Finally, $P_{\sigma+\rho+\delta}$ is the convex-hull of S, R and D.

Example: The frame of a polyhedron in $\mathbb{R}^3$ is given by

$$S=\left\{\begin{pmatrix}1\\-1\\0\end{pmatrix},\begin{pmatrix}-1\\1\\0\end{pmatrix}\right\},\ R=\left\{\begin{pmatrix}0\\0\\1\end{pmatrix}\right\},\ D=\left\{\begin{pmatrix}1\\1\\0\end{pmatrix}\right\}$$

The first approximation $P_1=\{s_1\}$ is defined by the system of restraints $(x^1=1,\ x^2=-1,\ x^3=0)$. The second approximation P_2 is the convex-hull of P_1 and $\{s_2\}$ that is $(0\leq\lambda\leq 1,\ x^1-2\lambda=-1,\ x^2+2\lambda=1,\ x^3=0)$. Eliminating λ we get $(x^3=0,\ x^1+x^2=0,\ -1\leq x^1\leq 1,\ -1\leq x^2\leq 1)$. P_3 is obtained by adjoining the ray $(0,0,1)$ to P_2 that is $(\mu\geq 0,\ x^1+x^2=0,\ -1\leq x^1\leq 1,\ -1\leq x^2\leq 1,\ x^3-\mu=0)$. Eliminating μ we get $(x^1+x^3=0,\ -1\leq x^1\leq 1,\ -1\leq x^2\leq 1,\ x^3\geq 0)$. The last approximation P_4 is the convex-hull of P_3 and the line $(1,1,0)$ that is $(x^1+x^2-2\nu=0,\ -1\leq x^1-\nu\leq 1,\ -1\leq x^2-\nu\leq 1,\ x^3\geq 0)$. Eliminating ν we get the final solution $(x^3\geq 0,\ -2\leq x^1-x^2\leq 2)$.
End of example.

3.3.1.2. Simplification of a system of linear inequalities.

It is often the case that projections lead to projected systems of restraints containing a great number of irrelevant restraints which can be eliminated without changing the polyhedron represented by the system of restraints. For the sake of efficiency these irrelevant restraints must be excluded. Knowing a frame of the polyhedron, the corresponding system of linear inequalities can be simplified according to the following remarks due to Lanery[1966]:

- An inequality which is never saturated by a vertex of the frame is irrelevant.
- An inequality which is saturated by all vertices and all rays of the frame represents an equality. All the equalities are found in that way.
- Let $C_1 : (a_1x\leq b_1)$ and $C_2 : (a_2x\leq b_2)$ be two inequations of the system of restraints which are not equations. Let $\sqsubseteq$ be the quasi-ordering defined by $\{C_1\sqsubseteq C_2\} \Leftrightarrow \{\{\forall s\in S,\ (a_1s=b_1) \Rightarrow (a_2s=b_2)\}$ and $\{\forall r\in R,\ (a_1r=0) \Rightarrow (a_2r=0)\}\}$. Hence $C_1\sqsubseteq C_2$ means that C_2 is saturated by any vertex or ray saturating C_1. If $C_1\sqsubseteq C_2$ but not $C_2\sqsubseteq C_1$ then C_1 is irrelevant. If $C_1\sqsubseteq C_2$ and $C_2\sqsubseteq C_1$ then one and only one of the inequations is irrelevant so that one of them can be excluded.

Applying these results to a system of restraints we obtain a minimal system of restraints (with no irrelevant inequality) corresponding to the same polyhedron.

3.4. Conversion from the linear restraints to the frame representation of the polyhedron

Based on the pivot method of linear programming, numerous algorithms have been designed to find all vertices of a convex polyhedron (Balinski[1961], Lanery[1966], Manas-Nedoma[1968], Mattheis[1973], Dyer-Proll[1977]). We shall give a brief description of Lanery's method which is general enough to find a frame (vertices, extreme rays, lines) of a convex polyhedron. We first recall some elements of linear programming (Simonnard[1973]).

3.4.1. Basic concepts of linear programming

A system of linear inequalities:

[3.4.1.1] $(\Sigma_{i=1}^{n}(a_i^j x^i) \le b^j \ : \ j=1..m)$

can be written in the equivalent form:

$$(\Sigma_{i=1}^{n}(a_i^j x^i)+y^j = b^j,\ y^j \ge 0 \ : \ j=1..m).$$

Hence a system of restraints can be written in *standard form:*

[3.4.1.2] $AX=B,\ X^E \ge 0$

where A is a (n+m)xm real matrix, B is an m-vector $E=\{n+1, \ldots, n+m\}$, $F=\{1,\ldots,n\}$. The variables $\{x^i : i \in F\}$ are the *initial variables,* the $\{x^i : i \in E\}$ are the *slack variables.*

A *basis* of the system [3.4.1.2] is a non singular mxm submatrix A_I of A. The system can be written in *canonical form* with respect to the basis A_I:

[3.4.1.3] $X^I+(A_I^{-1}A_J)X^J = A_I^{-1}B,\ X^E \ge 0$ where

$J=\{1,\ldots,n+m\}-I$. The variables x^i such that $i \in I$ are said to be *in basis.* The basis A_I is *feasible* if and only if $(A_I^{-1}B)^E \ge 0$. Two feasible bases A_I and A_J are said to be *adjacent* if and only if the cardinal of the set $I \cap J$ is equal to m-1. If two bases are adjacent the classical *pivot* operation transforms the system written in canonical form with respect to A_I into an equivalent system in canonical form with respect to A_J. The *artificial basis method* which is the initialization step of the *simplex method* transforms the system of restraints [3.4.1.2] into an equivalent system in canonical form [3.4.1.3] with respect to a feasible basis of [3.4.1.2] whenever such a basis exists.

3.4.2. Principles of Lanery's method.

The graph of the adjacency relation on the set $\{A_I\}$ of feasible bases containing all initial variables (i.e. such that $F \subset I$) is connected. Hence given such a basis we can by successive pivoting operations and an exhaustive traversal technique find all feasible basis of a given system of restraints.

- If A_I is a feasible basis such that $F \subset I$ then the vector $(A_I^{-1}B)^F$ corresponds to a vertex of the convex polyhedron defined by the system of restraints. To each vertex of a polyhedron containing no line corresponds at least one feasible basis. If two vertices are adjacent then there are two adjacent feasible bases respectively corresponding to them.
- Let $(AX=B,\ X^E \ge 0)$ be the canonical form of a system of restraints with respect to the feasible basis A_I containing all initial variables. Then Lanery shows that if a column $i_0 \in (E-I)$ satisfies the condition:

[3.4.2.1] $\{\forall j \in [1,m],\ \{A_{i_0}^j > 0\} \Rightarrow \{\forall k \in (E \cap I),\ A_k^j = 0\}\}$

then the vector $r \in \mathbb{R}^n$ defined by $r^i = -A_{i_0}^{j_0}$ where $i \in [1,n]$ and j_0 is the unique index such that $A_i^{j_0} = 1$ is an extreme ray of the polyhedron. Applying this result to each column i_0 verifying [3.4.2.1] in each feasible basis corresponding to a vertex s, we find all extreme rays adjacent to s. Since each extreme ray of a polyhedron that contains no line is adjacent to a vertex we can find all extreme rays of such a polyhedron.
- For polyhedra containing a line there is no feasible bases containing all initial variables.

Hence let $(AX=B,\ X^E \ge 0)$ be the canonical form with respect to any feasible basis A_I. Let $i_0 \in (F-I)$ be a column satisfying the condition:

[3.4.2.2] $\{\forall j \in [1,m],\ \forall k \in I,\ \{A_{i_0}^j \ne 0\} \Rightarrow \{A_k^j = 0 \text{ or } k \in F\}$

Let $d(i_0)$ be the vector of $\mathbb{R}^n$ the i-th component of which is defined by:

$d(i_0)^i =$ *if* $i=i_0$ *then* 1
elsif $i \in I$ *then* $-A_{i_0}^{j_0}$ {where j_0 is the unique index such that $A_i^{j_0} = 1$}
else 0 *fi;*

Then Lanery shows that $d(i_0)$ corresponds to a line of the polyhedron.

Also if the basis A_I is such that each $i_0 \in (F-I)$ verifies the property [3.4.2.2], then the set $\{d(i_0) : i_0 \in (F-I)\}$ is a basis of the greatest linear variety contained in the polyhedron.

3.4.3. Algorithm for finding the frame of a convex polyhedron.

Let $AX \le B$ be a system of m linear restraints among the variables $X \in \mathbb{R}^n$.

1 - Build the standard form $A[0]X=B[0]$, $X^E \ge 0$, where $X \in \mathbb{R}^{m+n}$.

2 - Apply the first step of the simplex method. If there is no feasible bases, the polyhedron is empty. Otherwise we get the system $A[1]X=B[1]$, $X^E \ge 0$ in canonical form with respect to the feasible basis A_{I_1} with $(B[1])^E \ge 0$.

3 - While there exists an initial variable staying out of the basis and satisfying [3.4.2.2] perform a pivoting which puts this variable in the basis by removing a slack variable from the basis.

4 - We get a system $A[2]X \le B[2]$, $X^E \ge 0$ in canonical form with respect to the basis I_2. Two subcases must be considered:

4.1. If all initial variables are in the basis ($F \subset I_2$), then the polyhedron contains no line and a vertex has been found. Then traverse exhaustively all feasible bases of the system and note at each step the vertex and the ray that are found. An efficient algorithm for this travel is given by Dyer-Proll[1977].

4.2. The initial variables $x^{i_1}, \ldots, x^{i_\delta}$ possibly remaining out of the basis verify [3.4.2.2]. So $\{d(i_1), \ldots, d(i_\delta)\}$ is a basis of the greatest linear variety contained in the polyhedron. The following system of retraints

$\{AX \le B,\ \Sigma_{j=1}^{n} d(i_k)^j x^j = 0,\ \forall k=1..\delta\}$ defines a new polyhedron P' which is a section of the initial polyhedron. Applying the algorithm to this new system of restraints case 4.1 is applicable since P' contains no lines so that the algorithm terminates.

Example: Let us compute a frame of the polyhedron P corresponding to the following system of restraints:

$$
[S1]\quad \begin{cases} -x_1+x_2-x_3 \le 0 \\ -x_1 \le -1 \\ -x_1-x_2+x_3 \le 0 \\ -x_2+x_3 \le 3 \end{cases}
$$

1-Using the matrix notations for denoting systems of equations the standard form of [S1] is:

	x^1	x^2	x^3	x^4	x^5	x^6	x^7	
	-1	1	-1	1	0	0	0	0
[S2]	-1	0	0	0	1	0	0	-1
	-1	-1	1	0	0	1	0	0
	0	-1	1	0	0	0	1	3

The system [S2] is in canonical form with respect to the infeasible basis $A_{\{4,5,6,7\}}$.

2-The artificial basis method supplies the system [S3]in canonical form with respect to the basis $A_{\{4,1,6,7\}}$ which is feasible:

	x^1	x^2	x^3	x^4	x^5	x^6	x^7	
	0	1	-1	1	-1	0	0	1
[S3]	1	0	0	0	-1	0	0	1
	0	-1	1	0	-1	1	0	1
	0	-1	0	0	0	0	1	3

3-Then we put into the basis as many initial variables as we can, we get the basis $A_{\{2,1,6,7\}}$:

	x^1	x^2	x^3	x^4	x^5	x^6	x^7	
	0	1	-1	1	-1	0	0	1
[S4]	1	0	0	0	-1	0	0	1
	0	0	0	1	-2	1	0	2
	0	0	0	1	-1	0	1	4

4-The initial variable x^3 remains out of the basis and cannot be put into the basis whithout getting x^2 out. The third column verifies [3.4.2.2]. Hence the vector d=(0,1,1) is the only line of the polyhedron P. Let us build the system of restraints of a section P' of P.

$$
[S5]\quad \begin{cases} -x_1+x_2-x_3 \le 0 \\ -x_1 \le -1 \\ -x_1-x_2+x_3 \le 0 \\ -x_2+x_3 \le 3 \\ x_2+x_3 = 0 \end{cases}
$$

Applying (1) and (2) to [S5] we get the canonical form with respect to the feasible basis $A_{\{2,3,1,6,7\}}$ which contains all initial variables. Thus we have found a first vertex of P' and we start the traversal of the graph of vertices:

	x^1	x^2	x^3	x^4	x^5	x^6	x^7	
	0	1	0	1/2	-1/2	0	0	1/2
	0	0	1	-1/2	1/2	0	0	-1/2
[S6]	1	0	0	0	-1	0	0	1
	0	0	0	1	-2	1	0	2
	0	0	0	1	-1	0	1	4

Basis: $A_{\{2,3,1,6,7\}}$
Vertex: s_1=(1,1/2,-1/2)
Column 5 satisfies [3.4.2.1]
Ray: r_1=(1,1/2,-1/2)
Adjacent feasible basis: $A_{\{2,3,1,4,7\}}$

	x^1	x^2	x^3	x^4	x^5	x^6	x^7	
	0	1	0	0	1/2	-1/2	0	-1/2
	0	0	1	0	-1/2	1/2	0	1/2
[S7]	1	0	0	0	-1	0	0	1
	0	0	0	1	-2	1	0	2
	0	0	0	0	1	-1	1	2

Basis:$A_{\{2,3,1,4,7\}}$
Vertex: s_2=(1,-1/2,1/2)
No column satisfies [3.4.2.1]
Adjacent feasible bases: $A_{\{2,3,1,6,7\}}$already found and $A_{\{2,3,1,4,5\}}$

	x^1	x^2	x^3	x^4	x^5	x^6	x^7	
	0	1	0	0	0	0	-1/2	-3/2
	0	0	1	0	0	0	-1/2	3/2
[S8]	1	0	0	0	0	-1	1	3
	0	0	0	1	0	-1	2	6
	0	0	0	0	1	-1	1	2

Basis: $A_{\{2,3,1,4,5\}}$
Vertex: s_3=(3,-3/2,3/2)

Column 6 satisfies [3.4.2.1]
Ray: r_2=(1,0,0)
Adjacent feasible basis: $A_{\{2,3,1,4,7\}}$ already found

All feasible bases that contain initial variables have been found, the algorithm terminates with the following frame for P:
[S = {(1,1/2,-1/2),(1,-1/2,1/2),(3,-3/2,3/2)},
R = {(1,1/2,-1/2),(1,0,0)},
D = {(0,1,1)}]
End of example.

3.4.4. Simplification of a frame

Lanery[1966] proposes a method for eliminating all irrelevant members of a frame when a system of restraints of the polyhedron is known. This method is the dual of the one given at paragraph 3.3.1.2. for simplifying a system of inequalities. Lanery's method is based on the following results:
- A vertex or a ray saturating no inequality is irrelevant.
- A ray saturating all restraints corresponds to a line.
- If e_1 and e_2 are two vertices or two rays (which are notlines) in the frame the quasi-ordering $\sqsubseteq$ is defined by:
  $\{e_1 \sqsubseteq e_2\}$ <=> {every inequality satured by e_1 is satured by e_2} then:
  $\{e_1 \sqsubseteq e_2\}$ and $\{e_2 \not\sqsubseteq e_1\}$imply that e_1 is irrelevant
  $\{e_1 \sqsubseteq e_2\}$ and $\{e_2 \sqsubseteq e_1\}$ imply that one but only one of the two elements may be eliminated.

3.5. On the use of two representations for assertions.

The search for the frame of a polyhedron as described above may become very expensive although many important optimizations are possible. The number of vertices of a polyhedron generated in $\mathbb{R}^n$ by m inequalities is known to be bounded only by functions increasing very quickly with m and n, Saaty[1955], Klee[1964]. So the use of this method should be avoided except when applied to polyhedra which are thought to have very few vertices. However the use of two representations is necessary for the following reasons:

- Some operations like the convex-hull of two polyhedra can only be performed on the frame representation, others like widening require the restraints representation.
- In order to define the inclusion relation between two polyhedra it is very useful to know both representations. Indeed $P_1 \subseteq P_2$ if and only if each element of the frame of P_1 satifies the restraint of P_2.
- It appears that it is difficult to simplify one representation without knowing the other one and neither can be used efficiently without simplications. So we shall build at the same time and in a consistent way the systems of restraints and the frames of the polyhedra. Both representations will be simultaneously used to represent the assertions. Experience shows that this redundant representation is much less expensive than the frequent use of conversions.

4. TRANSFORMATION OF LINEAR ASSERTIONS BY ELEMENTARY LANGUAGE CONSTRUCTS.

Given the flowchart representation of programs we now give for each type of program node a transformation specifying the assertion associated with the output arc(s) of the node in terms of the assertions associated with the input arc(s) of the node and where relevant the content of this node. The conditions of Cousot[1977] guaranteeing the correctness of these transformations apply to the specific case analyzed here.

4.1. Program entry node.

In some cases (parameters of procedures) the restraints on the input variables may be known. In this case they constitute the assertion associated with the input arc. Otherwise the variables are assumed not to be initialized so that they satisfy no restraints. Hence the vector of their values may be any point of $\mathbb{R}^n$ where n is the number of variables involved in the program analysis. The corresponding polyhedron is given by a system with no restraints and equivalently by the frame consisting in:

- The vertex which is the origin of $\mathbb{R}^n$
- No ray
- The lines $d_1,\ldots d_n$ such that for every i=1..n, $d^i_i=1$ and $d^j_i=0$, for j=1..n and j≠i.

4.2. Assignment.

Let {A,B,S,R,D} be the representation of the input assertion P corresponding to the polyhedron $AX \leq B$ the frame of which is $S=\{s_i : i=1..\sigma\}$, $R=\{r_j : j=1..\rho\}$, $D=\{d_k : k=1..\delta\}$. Then the representation of the output assertion P' after the assignment $X^{l_0}:=E(X)$ is given by

assign $(P, X^{l_0}:=E(X)) = \{A',B',S',R',D'\}$.

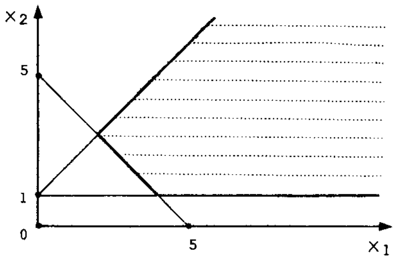
4.2.1. Assignment of a non linear expression.

If the expression E(X) is not linear it is assumed that any value of $\mathbb{R}$ can be assigned to X^{l_0}. Hence after the assignment $X^{l_0}:=E(X)$ nothing is known about the value of X^{l_0}. Therefore X^{l_0} is eliminated from the system of restraints $AX \leq B$ by a projection along the column l_0 (§ 3.3.1.1). For the frame representation this consists in adding the line d (such that $d_{l_0}=1$ and $d_j=0$ for every j=1..n different from l_0) to the set of lines D. Then in general the representation {A',B',S',R',D'} of P' is not minimal and must be simplified (§3.3.1.2 and §3.4.4).

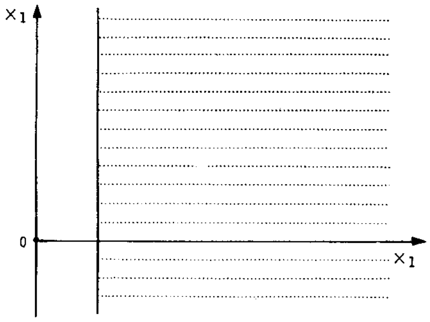
Example: Let n=2 and P be the input assertion represented by the system of restraints:

$$\begin{cases} x_1 - x_2 \geq -1 \\ x_2 \geq 1 \\ x_1 + x_2 \geq 5 \end{cases}$$

The frame representation is S={(2,3),(4,1)}, R={(1,1),(1,0)}, D=∅. The geometrical interpretation is:



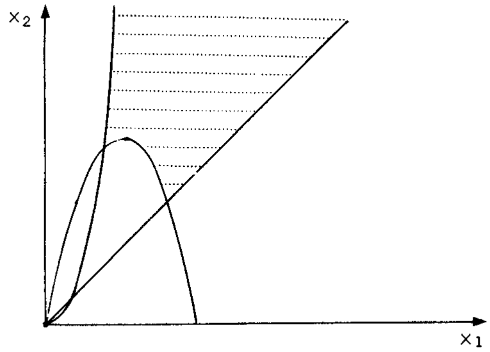
The output assertion P' after the assignment $x_2:=x_1*x_2$ is $\{x_1 \geq 0,\ x_1 \geq 2\}$, the frame of which is S'={(2,3),(4,1)}, R'={(1,1),(1,0)}, D'={(0,1)}. Simplifying we get $\{x_1 \geq 2\}$, S'={(2,3)}, R'={(1,0)}, D'={(0,1)} that is:



The approximation is obviously very coarse since the substitution of $x_1 * x_2$ for x_2 in the input system of constraints would lead to:

$$\begin{cases} x_2 \leq x_1+(x_1)^2 \\ x_2 \geq x_1 \\ x_2 \geq 5x_1-(x_1)^2 \end{cases}$$

However the corresponding domain is not a polyhedron and this situation is hardly manageable:



Note however that the exact domain is covered by the approximate domain (Cousot[1977]). Also, a more precise analysis is feasible. For example $\{x_1 \geq 0$, and $x_2 \geq 0\}$ implies $\{x_1 * x_2 \geq 0\}$ or the assignment x := y**2 implies that x is greater than or equal to zero.
End of example.

4.2.2. Assignment of a linear expression

The assignment is of the form $X^{l_0} := \Sigma_{i=1}^{n}(a_i X^i)+b$ where a is a n-row vector of integers or reals and b is an integer or a real. The transformation consists in an alteration of the basis of the space $\mathbb{R}^n$. The output assertion P' is defined by the frame {S',R',D'} computed as follows:

- $S'=\{s'_1,\dots,s'_\sigma\}$ where s'_i is defined by $s'^{l_0}_i = as_i+b$ and $s'^{l}_i = s^{l}_i$ where $l=1..\sigma$ with $l \neq l_0$.
- $R'=\{r'_1,\dots,r'_\rho\}$ where r'_j is the vector defined by $r'^{l_0}_j = ar_j$ and $r'^{l}_j = r^{l}_j$ for $l=1..n$ and $l \neq l_0$.
- $D'=\{d'_1,\dots,d'_\delta\}$ where d'_k is the vector defined by $d'^{l_0}_k = ad_k$ and $d'^{l}_k = d^{l}_k$ for $l=1..n$ and $l \neq l_0$.

The system of restraints corresponding to P' can be obtained as the convex-hull of the frame {S',R',D'}. However, following Karr[1976] this system of restraints can be computed directly.

4.2.2.1. Invertible assignments

The assignment is of the form $X^{l_0} := \Sigma_{i=1}^{n}(a_i X^i)+b$ $a^{l_0} \neq 0$. The fact that $a^{l_0} \neq 0$ allows us to carry over our knowledge of the previous value of X^{l_0} to the new value of X^{l_0}. To see this, denote the values of the variables by X before and by X' after the invertible assignment statement. Then for $l=1..n$ and $l \neq l_0$ we have $X'^{l}=X^{l}$ whereas $X'^{l_0}=aX+b$. Therefore $X = MX'+K$ as defined by

- $X^{l} = X'^{l}$ for $l \in ([1,n]-\{l_0\})$
- $X^{l_0} = (X'^{l_0} - \Sigma_{l \neq l_0}(a_l X'^{l_0}) - b)/a_{l_0}$

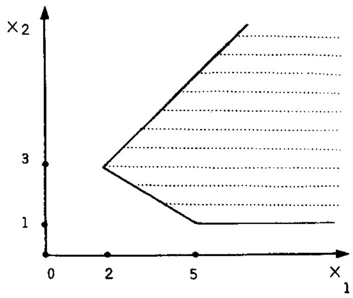
Also $\{AX \leq B\}$ is equivalent to $\{(MA)X' \leq (B-AK)\}$ which leads to the output system of restraints satisfied by X'.

4.2.2.2. Non-invertible assignments

If $a^{l_0}=0$ then we cannot solve X' in terms of X so that some information is lost by the assignment. Hence X^{l_0} is eliminated from the input restraints by a projection operation. The case is similar to the one of the assignement of a non-linear expression except that the restraint $X^{l_0} = aX+b$ is adjoined to the resulting system.
Example:

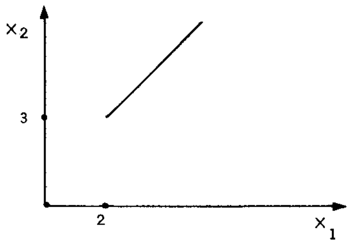
Let P be the input assertion defined by

$$\begin{cases} x_2 \geq 1 \\ x_1+x_2 \geq 5 \\ x_1-x_2 \geq -1 \end{cases}$$



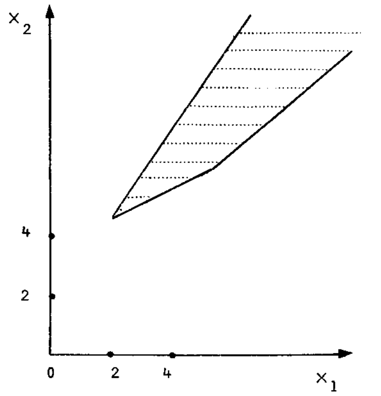
The assignment $x_2 := x_1+1$ is not invertible. The elimination of x_2 in the input system of restraints leads to $\{x_1 \geq 2\}$ so that the output system of restraints is:

$$\begin{cases} x_1 \geq 2 \\ x_2-x_1 = 1 \end{cases}$$



The assignement $x_2 : x_1+x_2/2+1$ is invertible so that $x_1=x'_1$ and $x_2 = 2x'_2-2x'_1-2$. Substituting in the input system of restraints we get the output system:

$$\begin{cases} 2x'_2 -2x'_1 \geq 3 \\ 2x'_2 - x'_1 \geq 7 \\ -2x'_2 +3x'_1 \geq -3 \end{cases}$$



End of example.

4.3. Test Nodes

Let P(A,B,S,R,D) be the assertion associated with the input arc to a decision node testing some boolean condition C. Let $P_t(A_t,B_t,S_t,R_t,D_t)$ and $P_f(A_f,B_f,S_f,R_f,D_f)$ be the assertions associated respectively with the true and false exits of the test. Obviously P_t=P *and* C and P_f=P *and not*(C).

The condition C is said to be linear if and only if it is of the form $aX \le b$ or $aX=b$ where a is an n-row-vector of integers or reals, X is the n-column-vector of program variables and b is an integer or a real.

4.3.1. Non-linear tests.

If C is not a linear condition we "ignore" the test by putting $P_t = P_f = P$. This is certainly valid, but it may not be as much information as could be gathered. As for non linear assignments specific studies can be made of how best to handle test conditions which are not linear (for example $\{Log(x) \ge 0\}$ implies $\{x \ge 1\}$).

4.3.2. Linear equality tests.

When C is a linear condition let H be the hyperplane $\{X \in \mathbb{R}^n : aX=b\}$.

If C is of the form aX=b then either P is included in H in which case P_t=P and $P_f=\emptyset$ or P is not included in H in which case $P_t=P \cap H$ and P_f=P. Note that $\{AX \le B$ and $aX \ne b\}$ is approximated by $\{AX \le B\}$ since the domain $\{X \in \mathbb{R}^n : AX \le B$ and $aX \ne b\}$ is not, in general, a closed convex polyhedron.

The frame of $P \cap H$ is found thanks to the following results:

1 - Each vertex s of $P \cap H$ lies on an edge of P according to one of the following alternatives:
   - 1.1- s is a vertex of P that belongs to H,
   - 1.2- there are two adjacent vertices s_1 and s_2 of P such that $s=\lambda s_1+(1-\lambda)s_2$ where $\lambda=(b-as_2)/(as_1-as_2)$ belongs to [0,1].
   - 1.3- there are a vertex s_1 and a ray r_1 of P such that s_1 is adjacent to r_1 in P and $s=s_1+\mu r_1$ where $\mu=(b-as_1)/ar_1$ is positive.
   - 1.4- there are a vertex s_1 and a line d_1 of P such that $s=s_1+\nu d_1$ where $\nu=(b-as_1)/ad_1$.

2 - Each extreme ray r of $P \cap H$ is either
   - 2.1- an extreme ray of P that belongs to H
   - 2.2- or, a positive combination of two adjacent extreme rays r_1 and r_2 of P: $r=r_1+\mu r_2$ where $\mu= ar_1/ar_2$ must be positive.
   - 2.3- or the sum of an extreme ray r_1 of P and a vector linearly dependant of a line d_1 of P, that is $r=r_1-(ar_1/ad_1)d_1$.

3 - Finally a vector d is a line of $P \cap H$ if and only if there are two lines d_1 and d_2 in P such that $d=d_1-(ad_1/ad_2)d_2$.

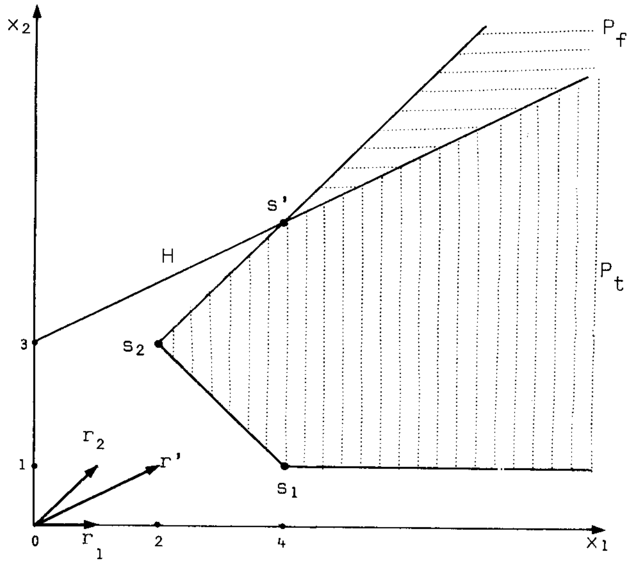
4.3.3. Linear inequality tests

If C is of the form $aX \le b$ then P_t=(P *and* $aX \le b$) whereas P_f = (P *and* $aX \ge b$).(Note that the inequality is not strict since P_f must be a closed polyhedron and that we can write $aX \ge b+1$ for integers).

As above, the determination of the frames of P_t and P_f makes use of a frame (S',R',D') of the intersection $P \cap H$ of P with the hyperplane $H=\{X \in \mathbb{R}^n : aX=b\}$.

The set S_t of vertices of P_t is $\{s \in S : as<b\} \cup S'$. If S_t is empty then P_t is the empty polyhedron whereas P_f equals P. Otherwise S_t is not empty and The set R_t of extreme rays of P_t is $\{r \in R : ar<0\} \cup \{d \in D : ad<0\} \cup \{-d \in D : ad>0\} \cup R'$. The set of lines D_t of P_t is D'. A symmetric reasoning is used to determine a frame of P_f. The resulting frames (S_t,R_t,D_t) and (S_f,R_f,D_f) are not necessarily minimal and must be simplified (§ 3.4.4).

Example: Let P be an input assertion defined by $\{(x_2 \ge 1,\ x_1+x_2 \ge 5,\ x_1-x_2 \ge -1),\ S=\{s_1=(4,1),\ s_2=(2,3)\},\ R=\{r_1=(1,0),\ r_2=(1,1)\},\ D=\emptyset\}$. Let P_t and P_f be the output assertions associated respectively with the true and false exits of a test on the condition $x_1+6 \ge 2x_2$ (that is $aX \le b$ where a=(-1,2), b=6 and X is the 2-column-vector (x_1,x_2)). Let H be the hyperplane defined by the equation $x_1+6=2x_2$. The geometrical interpretation is the following:



In order to determine a frame of $P \cap H$, it is necessary to know the adjacent elements of the frame of P . (s_1,s_2), (s_1,r_1), (s_2,r_2), (r_1,r_2) are adjacent. The vertex s' of H corresponds to the case 4.3.2.1.3, that is $s' = s_2+((b-as_2)/ar_2)r_2 = (4,5)$.

The extreme ray r' of $P \cap H$ corresponds to case 4.3.2.2.2 that is $r' = r_1-(ar_1/ar_2)r_2 = (2,1)$.

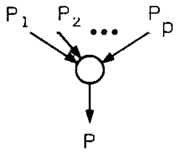
A frame of P_t is given by $S_t=\{s_1,s_2,s'\}$, $R_t=\{r_1,r'\}$, $D_t=\emptyset$. A frame of P_f is given by $S_f=\{s'\}$,

$R_f=\{r_2,r'\}$, $D_f=\emptyset$.

End of example.

4.4. Simple junction nodes.

Simple junction nodes correspond for example to the merge of the "then" and "else" paths in a conditional statement. More generally let $P_1, P_2, \ldots, P_p$ be the assertions associated with the input arcs to the simple junction node and P the output assertion:



Then the output assertion $OR_{i=1}^{p}(P_i)$ does not necessarily correspond to a convex polyhedron so that it must be approximated by the convex-hull P of $P_1, \ldots, P_p$ which is the least polyhedron containing $P_1, \ldots, P_p$.

Since the convex-hull operation is associative we can assume without lost of generality that there are two input arcs (p=2). Given $P_1(A_1,B_1,S_1,R_1,D_1)$ and $P_2(A_2,B_2,S_2,R_2,D_2)$ the frame of P = *convex-hull*(P_1,P_2) is $S=S_1 \cup S_2$, $R=R_1 \cup R_2$ and $D=D_1 \cup D_2$. The system of restraints $AX \leq B$ describing P is obtained by the convex-hull of the frame (S,R,D) which can be computed by successive approximations (§ 3.3.1.1). As soon as the system of restraints $\{AX \leq B\}$ of P is known the frame (S,R,D) of P can be simplified (3.4.4).

Notice that *convex-hull*(P_1,P_2) cannot be computed directly from the systems of restraints $A_1X \leq B_1$ and $A_2X \leq B_2$. In order to avoid a costly conversion (§ 3.4) the redundant frame representation has been kept along with the restraints representation. For the output assertion P the conversion from frame (S,R,D) to restraints $AX \leq B$ representation is less expensive (§ 3.3). Since the system of restraints of P_1 (or P_2) is known. This conversion is optimized as follows:

convex-hull$(\{S_1 \cup S_2,\ R_1 \cup R_2,\ D_1 \cup D_2\})$

= *convex hull*(*convex-hull*$(\{S_1,R_1,D_1\})$, $\{S_2,R_2,D_2\}$)
= *convex-hull*$(A_1X \leq B_1,\ \{S_2,R_2,D_2\})$

so that starting from $A_1X \leq B_1$ we can successively incorporate in the convex-hull the elements of the frame of P_2.

Example:

$P_1=\{(x_1 \geq 0, x_2 \geq 0, x_1+x_2 \leq 1),\ S_1=\{(0,0),(1,0),(0,1)\},$
$R_1=D_1=\emptyset\}$

$P_2=\{(x_1 \geq 1, x_2=2),\ S_2=\{(1,2)\},\ R_2=\{(1,0)\},\ D_2=\emptyset\}$

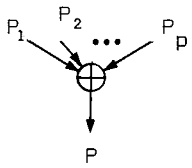
The convex-hull of P_1 (given by $A_1X \leq B$) and the vertex s=(1,2) of P_2 is obtained (§3.3.1.1) by elimination of λ in $\{0 \leq \lambda \leq 1,\ A_1X+\lambda(A_1s-B_1) \leq A_1s\}$. Eliminating λ in $\{0 \leq \lambda \leq 1,\ x_1+\lambda \geq 1,\ x_2+2\lambda \geq 2,\ x_1+x_2+2\lambda \leq 3\}$ we get the approximation $A'X \leq B'$ given by $\{x_2-x_1 \leq 1,\ 0 \leq x_1 \leq 1,\ x_2 \geq 0\}$.

Incorporating the ray r=(1,0) of P_2 in $A'X \leq B'$ by elimination of μ in $\{\mu \geq 0,\ A'x-\mu A'r \leq B'\}$ that is $\{\mu \geq 0,\ x_2-x_1+\mu \leq 1,\ 0 \leq x_1-\mu \leq 1,\ x_2 \geq 0\}$ we get the convex-hull of P_1 and P_2 given by $\{0 \leq x_2 \leq 2,\ x_1 \geq 0,\ x_2-x_1 \leq 1\}$.

End of example.

4.5. Loop junction nodes.

Each cycle in the program graph contains a loop junction node:



The corresponding transformation
P = $P \nabla$ *convex-hull*$(P_1,P_2,\ldots,P_p)$ is defined by the widening operation ∇. Let $Q_1(A_1,B_1,S_1,R_1,D_1)$ and $Q_2(A_2,B_2,S_2,R_2,D_2)$ be two convex polyhedra. Then $Q_1 \nabla Q_2$ is the convex polyhedron consisting in the linear restraints of Q_1 verified by every element of the frame (S_2,R_2,D_2) of Q_2.

Example:

$Q_1 = \{(-x_1+2x_2 \leq -2, x_1+2x_2 \leq 6, x_2 \geq 0),$
$S=\{(2,0),(6,0),(4,1)\},\ R=\emptyset,\ D=\emptyset\}$

$Q_2 = (-x_1+2x_2 \leq -2, x_1+2x_2 \leq 10, x_2 \geq 0),$
$S=\{(2,0),(10,0),(6,2)\},\ R=\emptyset,\ D=\emptyset\}$

$Q_1 \nabla Q_2 = (-x_1+2x_2 \leq -2,\ x_2 \geq 0)$

End of example.

In order to compute the frame of $Q_1 \nabla Q_2$ it is necessary to convert from the linear restraint representation . Lanery's method (§ 3.4) is known to be expensive. However when applied after a widening operation the cost remains reasonable because of the following arguments:

- In the widening $Q_1 \nabla Q_2$ of Q_1 by Q_2 a certain number of vertices of Q_1 have been replaced by extreme rays. Hence the widening operation eliminates vertices, so that $Q_1 \nabla Q_2$ can be assumed to have a small number of vertices. Now it is the discovery of vertices which is the most costly in Lanery's method.
- A frame of Q_1 is known and Q_1 is included in $Q_1 \nabla Q_2$. Therefore the initialization of the simplex method needs not to be used. Moreover since any restraint of $Q_1 \nabla Q_2$ is a restraint of Q_1 it is highly probable that the frames of Q_1 and $Q_1 \nabla Q_2$ have numerous common elements.

The correctness criterion of Cousot[1977] recalled at paragraph 2 is satisfied since $Q_1 \subseteq Q_1 \nabla Q_2$, $Q_2 \subseteq Q_1 \nabla Q_2$ and for every chain $C_0 \subseteq C_1 \subseteq \ldots \subseteq C_n \subseteq \ldots$. The chain $S_0=C_0$, $S_1=S_0 \nabla C_1$, ..., $S_n=S_{n-1} \nabla C_n$, ... is not an infinite strictly increasing chain since at each step n the number of restraints describing S_n is finite and less than or equal to the number of restraints describing S_{n-1}.

The definition of the widening operation must be a balance between compelling the convergence of the global analysis of the program (by throwing away the restraints that do not quickly stabilize in the program cycles) and discovering as much information as possible about the program. Hence it is wise not to perform widening operations at loop junction nodes before gathering the information along the program cycles containing that loop junction node. Also the definition of the widening which we have given is a tentative one. The experimentations that have been carried out seems to corroborate our choice but further studies are necessary to give a definite conclusion.

5. GLOBAL ANALYSIS OF PROGRAMS.

We illustrate the global analysis of programs on the following ad-hoc skeletal program which is simple enough to allow hand computations:

```
{P0}
        I:=2; J:=0;
{P1}
     L:
{P2}
        if ... then
{P3}
           I:=I+4;
{P4}
        else
{P5}
           J:=J+1; I:=I+2;
{P6}
        fi;
{P7}
        go to L;
```

The test involving some non-linear condition is not taken into account. Each assertion P^0_i, i=0..7 is initially the empty polyhedron Ø and the input assertion is propagated through the program graph:

$P^1_0 = \mathbb{R}^2$, S={(0,0)}, R=Ø, D={(1,0),(0,1)}

P^1_1 = *assign*(*assign*(P^1_0, I:=2), J:=0)
  = {I=2,J=0}, S={(2,0)}, R=Ø, D=Ø

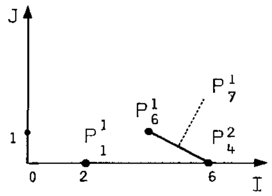
P^1_2 = *convex-hull*(P^1_1,P^0_7) = *convex-hull*(P^1_1,Ø) = P^1_1

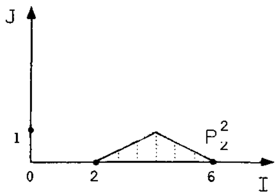
P^1_3 = P^1_5 = P^1_2

P^1_4 = *assign* (P^1_3,I:=I+4)
  = {I=6,J=0}, S={(6,0)}, R=Ø, D=Ø

P^1_6 = *assign*(*assign*(P^1_5,J:=J+1),I:=I+2)
  = {I=4,J=1}, S={(4,1)}, R=Ø, D=Ø

P^1_7 = *convex-hull*(P^1_4,P^1_6)
  = {I+2J=6,4≤I≤6}, S={(6,0),(4,1)}, R=Ø, D=Ø





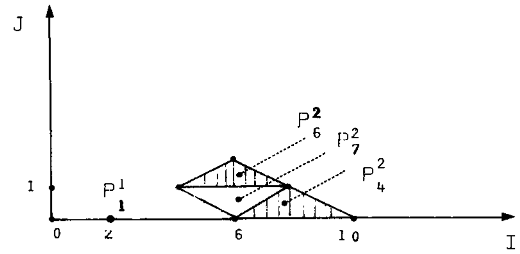
P^2_2 = *convex-hull*(P^1_1,P^1_7)
  = (2J+2≤I,I+2J≤6,0≤J), S={(2,0),(6,0),(4,1)}, R=Ø, D=Ø

P^2_3 = P^2_5 = P^2_2

P^2_4 = *assign*(P^2_3,I:=I+4)
  = (2J+6≤I,I+2J≤10,0≤J), S={(6,0),(10,0),(8,1)}, R=Ø, D=Ø

P^2_6 = *assign*(*assign*(P^2_5,J:=J+1),I:=I+2)
  (2J+2≤I,I+2J≤10,1≤J), S={(6,2),(8,1),(4,1)}, R=Ø, D=Ø

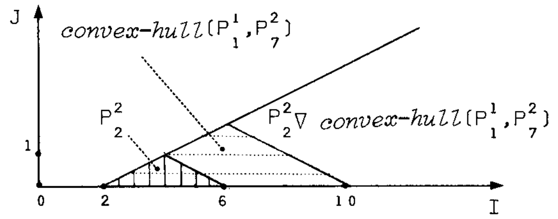
P^2_7 = *convex-hull*(P^2_4,P^2_6)
  = (2J+2≤I,6≤I+2J≤10,0≤J), S={(6,0),(10,0),(6,2),(4,1)}, R=D=Ø



When the loop body has been analyzed a widening operation takes place at the loop junction node L:

P^3_2 = P^2_2 ∇ *convex-hull*(P^1_1,P^2_7)

We have P^2_2 = (2J+2≤I, I+2J≤6, 0≤J) and *convex-hull*(P^1_1,P^2_7) = (2J+2≤I, I+2J≤10, 0≤J) with S={(2,0), (10,0), (6,0)}, R=D=Ø, so that I+2J≤6 which is the only constraint of P^2_2 not verified by every element of the frame of *convex-hull*(P^1_1,P^2_7) is eliminated by the widening operation.



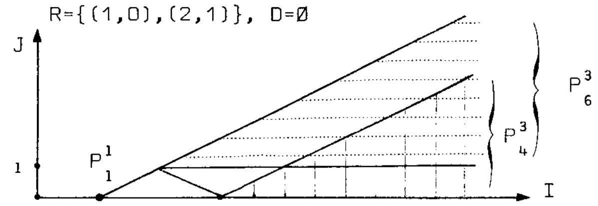
P^3_2 = {2J+2≤I,0≤J}, S={(2,0)}, R={(1,0),(2,1)}, D=Ø

P^3_3 = P^3_5 = P^3_2

P^3_4 = *assign*(P^3_3,I:=I+4)
  = {2J+6≤I,0≤J}, S={(6,0)}, R={(1,0),(2,1)}, D=Ø

P^3_6 = *assign*(*assign*(P^3_5,J:=J+1),I:=I+2)
  = {2J+2≤I,1≤J}, S={(4,1)}, R={(1,0),(2,1)}, D=Ø

P^3_7 = *convex-hull*(P^3_4,P^3_6)
  = {2J+2≤I,6≤I+2J,0≤J}, S={(6,0),(4,1)}, R={(1,0),(2,1)}, D=Ø



Then *convex-hull*(P^1_1, P^3_7) is included in P^3_2 so that the program analysis has converged.

The final result shows up linear restraints among the variables of the program that never appear explicitely in the program text and often escape the notice of anyone studying this simple example:

```
{0}            : no information
{1}            : I=2, J=0
{2},{3},{5}    : 2J+2≤I, J≥0
{4}            : 2J+6≤I, J≥0
{6}            : 2J+2≤I, J≥1
{7}            : 2J+2≤I, 6≤I+2J, J≥0
```

6. EXAMPLE.

On the next example (HEAPSORT, Knuth[1973,p.148]) it is not possible to trace the details of the analysis so that we directly provide the results produced by our experimental implementation:

```
      procedure HEAPSORT(integer value N;
                                  real array[1..N] T);
{1}    begin integer L,R,I,J; real K;
{2}     L:=(N div2)+1; R:=N;
{3}     if (L≥2) then
{4}      L:=L-1; {K:=T[L];}
{5}     else
{6}      {K:=T[R]; T[R]:= T[1];} R:=R-1;
{7}     fi;
{8}     while (R≥2) do
{9}      I:=L; J:=2*I;
{10}     while (J≤R) do
{11}      if (J≤R-1) then
{12}       if {T[J]<T[J+1]} then J:=J+1 fi;fi:
          if {K≥T[J]} then
{13}                    exit {of the inner loop} fi;
{14}      {T[I]:=T[J];} I:=J; J:=2*J;
{15}     od;
{16}     {T[I]:=K;}
{17}     if L≥2 then
{18}      L:=L-1; {K:=T[L];}
{19}     else
{20}      {K:=T[R]; T[R]:=T[1];} R:=R-1;
{21}     fi;
{22}     {T[1]:=K;}
{23}    od;
       end;
```

The procedure is analyzed with the input specification N≥2 (see Knuth[1973], p.146). This analysis does not take account of the statements involving operations on arrays, these statements such as {K:=T[L];}have been bracketed in the text of the procedure. The result of the analysis (taking about 20 seconds of C.P.U. time) is the following:

```
{1}        : N≥2
{2}        : N≥2, N≤2L≤N+1 R=N
{3}        : N≥2, N≤2L≤N+1, R=N, L≥2
{5}        : N≥2, N≤2L≤N+1, R=N, L≤2
{8}        : R≥2, 2L≤N+1, R+3≤N, 2L+2R+1≤3N, L≥1,
             R≤N
{10}       : R≥2, 2L≤N+1, R+3≤2N, 1≤L, R≤N, 2I=J,
             L≤I, 2I+6L+R+18≤12N, J≤R, 2L+2R+1≤3N,
             4I+2L+1≤2R+3N
{11}       : R≥2, 2L≤N+1, R+3≤2N, 1≤L, R≤N, 2I=J,
             L≤I, 2I+6L+R+18≤12N, J≤R-1,
             2L+2R+1≤3N, 4I+2L+1≤2R+3N
{12},{13}  : R+3≤2N, L≥1, R≤N, J≤2I+1, 2I≤J, L≤I,
             J≤R, 2L+2R+1≤3N
{15}       : J+2≤2I+R, 2J+2L≤4I+N+1, R+3≤2N, 1≤L,
             R≤N, 7J+6L+R+18≤12I+12N, 2I≤J≤2I+1,
             L≤I, 2L+2R+1≤3N, 8J+2L+1≤12I+2R+3N
{17}       : {15}, L≥2
{19}       : {15}, L≤2
{21}       : R≥1, 2L≤N+1, R+4≤2N, 2L+2R+3≤3N, L≥1,
             R≤N
```

Once the above invariant assertions have been discovered it is very easy using projections (3.3.1.1) to check statically that all array accesses are correct. Notice that some relationships among the variables of the procedure are not obvious and cannot be discovered by hand whithout deep understanding of the program.

7. NOTES ON THE EXPERIMENTAL IMPLEMENTATION.

We have produced an experimental implementation written in PASCAL on the CII-IRIS 80 computer. The length of the program is about 2500 lines.

The systems of equations and inequations have been represented for simplicity by real matrices and this sometimes results in a loss of precision. It seems very difficult to write the program so that this loss of precision is acceptable that is so that the relationships which have been found correspond to a domain including any value that each real variable can take during any execution of the analyzed program. However the main applications we have in mind (such as array bound checking) deal with integer variables. In this case the coefficients of the linear restraints are rationals, which can be represented as fractions (p/q where p and q are integers). Then the operations which are performed on these coefficients (+,-,x,/) are more costly but introduce no loss of precision.

We have noticed that programs involving numerical constants are better handled when these numerical constants are replaced by the declaration of a symbolic constant. It is often the case that the convergence of the analysis is faster althought the systems of restraints are bigger.

It seems to be very difficult to evaluate the cost of the analysis of a program. The cost of the analysis does not only depend on the length (number of lines) of the program but mainly on the complexity of the program graph (number of loops, degree of loop combination, etc...). It seems that the cost of an analysis is almost linear in the length of the program but exponential in the number of variables involved in the analysis. From this point of view nested static scopes (such as ALGOL 68 blocks which authorizes very local declarations or better EUCLID with its import mechanism) is useful since for example the variables of an outer block can often be analyzed independently of the variables of the inner blocks whereas the variable of the inner block can often be analyzed using only few global variables. In general it can be taken advantage of the scope rules of the usual languages.

We have deliberately taken the point of view not to take account of the legality restraints which must hold on the variables. For example if a program variable is used as an array index which do not take account of the fact that we should try to show that the value of this variable must be within the array bounds. Taking account of such facts we could propagate this information backward to the loop junction nodes so that we would have a guideline for the widening operation. This would enable us to combine the discovery and verification approaches.

Acknowledgements. We thank Radhia COUSOT for helpful discussions on linear programming and Mrs H.DIAZ for typing the manuscript.

8. REFERENCES

M.L.BALINSKI, *An algorithm for finding all vertices of convex polyhedral sets*, J. Soc. Indust. Appl. Mathem., 9, [March 1961]

A.CHARNES, W.W.COOPER and A.HENDERSON, *An Introduction to Linear Programming*, J.Wiley, New-York, [1953]

P.COUSOT and R.COUSOT, *Static determination of dynamic properties of programs*, 2nd Int. Symposium on Programming, B.Robinet(Ed.), Dunod, Paris, [1976]

P.COUSOT and R.COUSOT, *Abstracts interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints*, Conf. Record of the 4th ACM Symposium on Principles of Programming Languages, [Jan.1977]

N.E.DYER and L.G.PROLL, *An algorithm for determining all extreme points of a convex polytope*, Mathematical Programming, 12, [1977]

M.KARR, *Affine relationships among variables of a program*, Acta Informatica, 6, [1976]

V.KLEE, *Some characterizations of convex polyhedra*, Acta Mathematica, 102, [1959]

V.KLEE, *On the number of vertices of a convex polytope*, Canadian J. of Mathematics, 16,[1964]

D.E.KNUTH, The art of computer programming, vol.3, *Sorting and Searching*, Addison-Wensley Pub.Co., Reading, Mass, [1973]

H.W.KUHN, *Solvability and consistency for linear equations and inequalities*, Amer. Math. Monthly, 63, [1956]

E.LANERY, *Recherche d'un système générateur minimal d'un polyhedre convexe*, Thèse de 3ème cycle, Caen, France, [1966]

M.MANAS and J.NEDOMA, *Finding all vertices of a convex polyhedron*, Numerische Mathematik, 12, [1968]

T.H.MATTHEIS, *An algorithm for determining irrelevant constraints and all vertices in systems of linear inequalities*, Operations Research, 21, [1973]

T.L.SAATY, *The number of vertices of a polyhedron*, Amer. Math. Monthly, 62, [1955]

M.SIMONNARD, *Programmation Linéaire*, Dunod, Paris, [1973]

N.SUZUKI and K.ISHIHATA, *Implementation of an array bound checker*, Conf. Record of the 4th ACM Symposium on Principles of programming languages, [Jan.1977]

B.WEGBREIT, *Property extraction in well founded property sets*, IEEE Trans. on Soft. Eng., vol. SE-1, n°3, [Sept.1975]

H.WEYL, *The elementary theory of convex polyhedra*, Annals of Math. Study, 24, [1950]

Conference Record
of the
FIFTH ANNUAL ACM SYMPOSIUM ON PRINCIPLES OF PROGRAMMING LANGUAGES

Papers Presented at the Symposium
Tucson, Arizona
January 23-25, 1978

Sponsored by the
ASSOCIATION FOR COMPUTING MACHINERY
SPECIAL INTEREST GROUP ON AUTOMATA AND COMPUTABILITY THEORY
SPECIAL INTEREST GROUP ON PROGRAMMING LANGUAGES