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 cedex, France

1. INTRODUCTION

The model of abstract interpretation of programs developed 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):

begin Integer B,J,T;
\[B_1=N;\]
{1}
\[\text{while } B_1 \text{ do }\]
\[J:=1; T:=0;\]
{2}
\[\text{while } J \leq B-1 \text{ do }\]
\[\text{if } K[J] > K[J+1] \text{ then }\]
\[\text{EXCHANGE}[J,J+1]; \text{ (no side effects on } \text{N,B,J,T)}\]
{4}
\[T:=J,\]
{5}
\[fU;\]
{6}
\[J:=J+1;\]
{7}
\[od;\]
{8}
\[\text{if } T=0 \text{ then return } fU;\]
{9}
\[B:=T;\]
{10}
\[od;\]
{11}
end;
{12}

Without user provided inductive assertions nor human interaction we have automatically determined (in 1.562 seconds of C.P.U. time) that the following restraints must hold among the variables of the

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...
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 $P_i(V_1, \ldots, V_n)$ to each arc $i$ of the program. These predicates on the variables $V_1, \ldots, V_n$ 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 arcs(s) to the node.

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

3.1. Linear system of a convex polyhedron.

Let $x_1^1, \ldots, x_n^1$ 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

\[ \sum_{i=1}^{n} (a_{ij}^1 x^1_j - b^1_j) \leq 0 \quad j=1..m \]

(where $a_{ij}^1, b^1_j \in \mathbb{R}$), 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

\[ \sum_{i=1}^{n} (a_{ij}^1 x^1_j) \leq b^1_j \]

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 $\{x \in C \mid x_1, x_2 \in C \}$, $\forall x \in [0,1]$, $\lambda x_1 + (1-\lambda) x_2 \in C$. 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], Charney [1953].

Let $V_1, \ldots, V_p$ be vectors in $\mathbb{R}^n$. A vector of the form $\sum_{i=1}^{p} (\lambda_i V_i)$, where for each $i=1..p$ we have $\lambda_i \in \mathbb{R}$ is called a linear combination of the $V_i$. 
A line of a polyhedron $P$ is a vector $d$ such that both $d$ and $-d$ are rays of $F$: \{$(x+d, y+M, x+MdF)$\}. 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 its vertices 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 $\{s_1, \ldots, s_n\}$ of $P$ and of a positive combination of the extreme rays $\{r_1, \ldots, r_m\}$ of $P$:

\[x = \sum_{i=1}^{n} \lambda_i s_i + \sum_{j=1}^{m} \mu_j r_j\]

where $\lambda_i, \ldots, \lambda_n \in [0, 1]$ and $\mu_j \in \mathbb{R}$.

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 $S = \{s_1, \ldots, s_n\}$, $R = \{r_1, \ldots, r_m\}$, and $D = \{d_1, \ldots, d_n\}$ of vectors of $\mathbb{R}^n$ called the frame of the polyhedron as follows:

\[\{x \in X : x = \sum_{i=1}^{n} \lambda_i s_i + \sum_{j=1}^{m} \mu_j r_j\} \quad \forall \mu_j \geq 0\]

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:

\[\{x_1 \geq 2, x_2 \leq 1, x_1 + 2x_2 \geq 6, x_1 - 2x_2 \geq -6\}\]

is spanned by the following frame (see the diagram):

- vertices: $s_1 = (1, 0, 0), s_2 = (0, 1, 0), s_3 = (1, 1, 0)$
- extreme rays: $r_1 = (0, 0, 1), r_2 = (1, 0, -1)$
- no line.

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 $x^0 = (a_1, \ldots, a_{n-1}) \leq b$ be a linear inequality defining the polyhedron $P$. Then if the intersection of the hyperplane defined by $H = \{x : x^0 + \sum_{i=1}^{n} (a_i x_i) - b^i \}$ with $P$ is not empty, it is a face of $P$.

We say that a point $x$ satisfies the inequality $x^0 + \sum_{i=1}^{n} (a_i x_i) = b^i$ if and only if $\Sigma_{i=1}^{n} (a_i x_i) = b^i$. We say that the ray $r$ satisfies this inequality if and only if $x^0 + \sum_{i=1}^{n} (a_i x_i) = 0$.

Let $\delta$ 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 $\delta = 0$ we mean the edge of the section.) It follows that the number of inequalities satisfied 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 two adjacent extreme rays show at least $n-\delta-2$ inequalities simultaneously.

Finally, a vertex $s$ and a ray $r$ are adjacent iff they lie on an infinite edge which is parallel to $r$. Therefore $s$ must saturate all the inequalities satisfied 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 $S = \{s_1, \ldots, s_n\}, R = \{r_1, \ldots, r_m\}, D = \{d_1, \ldots, d_n\}$ 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 $\lambda_1, \ldots, \lambda_n, \mu_1, \ldots, \mu_m, \nu_1, \ldots, \nu_{n-1}$ such that:

- $\sum_{i=1}^{n} \lambda_i = 1$ for $i = 1, \ldots, g$
- $\sum_{i=1}^{m} \mu_i = 1$ for $i = 1, \ldots, g$
- $\sum_{j=1}^{m} \nu_j = 1$ for $i = 1, \ldots, g$

\[x = \sum_{i=1}^{n} \lambda_i s_i + \sum_{j=1}^{m} \mu_j r_j + \sum_{k=1}^{n-1} \nu_k d_k\]
Hence this polyhedron is characterized by a system of linear restraints in \( \mathbb{R}^{n+p+q} \) upon the variables
\[
x_1 \ (1=1..n), \quad \lambda_1 \ (1=1..q), \quad \mu_j \ (1=1..q), \quad \nu_k \ (1=1..q).
\]
Eliminating the \( \lambda_i, \mu_j, \nu_k \) we get a system of linear restraints in \( \mathbb{R}^n \). This elimination can always be done by the projection operation used in Kuhn[1959].

Let us represent the system of \( m \) inequalities by
\[
Ax \leq B
\]
where \( A \) is a \( m \times n \) 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 \( A^1 \) such \( A^1 \cdot x = 0 \), the restraint \( A^1 \cdot x \leq 1 \)
  is a part of the projected system.
- For each two rows \( A^{11} \) and \( A^{12} \) such that
  \[
  A^{11} \cdot x \leq 0, \quad A^{12} \cdot x \leq 0
  \]
  the restraint \( \{A^{11} \cdot x \leq 1, A^{12} \cdot x \leq 1\} \) is part of the projected system.

The projected system contains only zeros in the columns \( c \), hence it is independent of \( x_c \).

**Example:** Eliminating \( \lambda \) in the system
\[
\begin{align*}
-\lambda & \leq 0 \\
\lambda & \leq 1 \\
x_1 - x_2 & \leq 0 \\
x_1 + x_2 & \leq 2 \\
x_1 - x_2 & \leq 2 \\
x_1 + x_2 + 4\lambda & \leq 2
\end{align*}
\]
we get
\[
\begin{align*}
x_1 + x_2 & \leq 0 \\
x_1 - x_2 & \leq 2 \\
x_1 + x_2 + 4\lambda & \leq 2
\end{align*}
\]
End of example.

The convex hull of the frame \( S, R, D \) is computed by successive approximations \( P_1, P_2, \ldots, P_0 = \delta \) as follows:
- \( P_i \) is the point \( s_i \) so that the corresponding system of restraints is \( \{x = s_i\} \). Then for each
  \( i = 2, \ldots, p \), \( P_i \) is the convex hull of \( P_{i-1} \) and the point \( s_i \).
- Therefore \( \{x \in P_i\} \) if and only if \( \{x \in P_{i-1} \wedge x \in S_i\} \).

Then by a projection according to the column \( \lambda \), \( \lambda \) can be eliminated to get a system of restraints of \( P_i \).

- For each \( j = 1, \ldots, q \), \( P_j \) is obtained by adjoining the ray \( P_{j-1} \to P_j \). If \( P_j \) corresponds
  \( \{x \in P_j \mid x \cdot \lambda = 0\} \), eliminating \( \lambda \), \( \lambda \cdot x \leq 1 \).

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

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 \( C_1 : \{a_1 x \leq b_1\} \) and \( C_2 : \{a_2 x \leq b_2\} \) be two inequalities of the system of restraints which are not equations. Let \( \preceq \) be the quasi-ordering defined by \( C_1 \preceq C_2 \) if \( \forall x \in S : a_2 x \preceq b_2 \) and \( \forall a \in R : a_1 a \preceq a_2 a \).

Then by a projection according to the column \( \lambda \), \( \lambda \) can be eliminated to get a system of restraints of \( P_i \).

- For each \( j = 1, \ldots, q \), \( P_j \) is obtained by adjoining the ray \( P_{j-1} \to P_j \). If \( P_j \) corresponds
  \( \{x \in P_j \mid x \cdot \lambda = 0\} \), eliminating \( \lambda \), \( \lambda \cdot x \leq 1 \).

### 3.4. Conversion from the linear restraints to the
3.4.1. Basic concepts of linear programming

A system of linear inequalities:
(3.4.1.1) \( \sum_{i=1}^{n} (a_{i}^{j}x^{i}) \leq b^{j} : j=1..m \)
can be written in the equivalent form:

\( \sum_{i=1}^{n} (a_{i}^{j}x^{i}) + y^{j} = b^{j}, y^{j} \geq 0 : j=1..m. \)

Hence a system of constraints can be written in standard form:

\[ (3.4.1.2) \quad \begin{align*}
Ax & = b, \quad x \geq 0
\end{align*} \]

where \( A \) is an \((n+m) \times m\) 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 \( m \times m \) submatrix \( A_{I} \) of \( A \). The system can be written in canonical form with respect to the basis \( A_{I} \):

\[ (3.4.1.3) \quad x^{i} = (A^{-1} A_{I}) x^{I} = A^{-1}_{I} B, \quad x^{E} \geq 0 \quad \text{where} \]

\( I = \{1, \ldots, n+m\} - I \). 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 \( (A_{I}^{-1} B)^{E} \geq 0 \). 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 constraints \([3.4.1.2]\) 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 Laener's method.

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

- If \( A_{I} \) is a feasible basis such that \( F \cap I \) then the vector \( (A^{-1}_{I} B)^{F} \) corresponds to a vertex of the convex polyhedron defined by the system of restraints. To find a feasible basis corresponding to a vertex \( a \), we find all extreme rays adjacent to \( a \). 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 \((Ax=B, x \geq 0)\) be the canonical form with respect to any feasible basis \( A_{I} \). Let \( i_{0} \in (F-I) \) be a column satisfying the condition:

\[ (3.4.2.2) \quad \forall i \in (I, n+m), \quad A_{I}^{-1} B \rightarrow \{A_{I}^{-1} B_{i_{0}} \text{ or } k_{e} f\} \]

Let \( d(i_{0}) \) be the vector of \( \mathbb{F}^{i_{0}} \) the \( i \)-th component of which is defined by:

\[ d(i_{0})^{i} = \begin{cases} 1 & \text{if } i = i_{0} \\ -A_{I}^{-1} \text{ where } i_{0} \text{ is the unique index such that } A_{I}^{-1} \neq 1 \\ 0 & \text{if } i \neq i_{0} \end{cases} \]

Then Laener 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 \( AX=B \) be a system of \( m \) linear restraints among the variables \( X \in \mathbb{F}^{n} \).

1 - Build the standard form \( A[0]X=B[0], \quad X \geq 0 \), where \( X \in \mathbb{F}^{n+n} \).

2 - Apply the first step of the simplex method.

- If there is no feasible bases, the polyhedron is empty. Otherwise we get the system \( A[1]X=B[1], \quad X \geq 0 \) in canonical form with respect to the feasible basis \( A_{I} \) with \( (B[1])^{E} \geq 0 \).

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 \( A[2]X=B[2], \quad X \geq 0 \) in canonical form with respect to the basis \( I_{2} \). Two subcases
{AX ≤ b, \sum_{j=1}^{n} d_i^j x_j = 0, \forall k = 1 \ldots 6} 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:

\[
\begin{align*}
-x_1 + x_2 - x_3 & \leq 0 \\
-x_1 & \leq 1 \\
-x_1 + x_2 & \leq 0 \\
-x_2 + x_3 & \leq 3
\end{align*}
\]

[S1]

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

\[
\begin{array}{cccccccc}
x^1 & x^2 & x^3 & x^4 & x^5 & x^6 & x^7 \\
-1 & 1 & -1 & 1 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 0 & 1 & 0 & 0 & -1 \\
-1 & -1 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & -1 & 1 & 0 & 0 & 0 & 1 & 3
\end{array}
\]

[S2]

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 in canonical form with respect to the basis \( A_{(4,5,6,7)} \) which is feasible:

\[
\begin{array}{cccccccc}
x^1 & x^2 & x^3 & x^4 & x^5 & x^6 & x^7 \\
0 & 1 & -1 & 1 & -1 & 0 & 0 & 1 \\
1 & n & n & n & n & n & n & 0 & 1
\end{array}
\]

\[
\begin{array}{cccccccc}
x^1 & x^2 & x^3 & x^4 & x^5 & x^6 & x^7 \\
0 & 1 & 0 & 0 & 1/2 & -1/2 & 0 & 0 \\
0 & 0 & 0 & 1 & 1/2 & 0 & 0 & -1/2 \\
0 & 0 & 0 & 0 & 1/2 & -1/2 & 0 & 0 \\
1 & 0 & 0 & 0 & 1/2 & -1/2 & 0 & 0 \\
0 & 0 & 0 & 1 & -2 & 1 & 0 & 2 \\
0 & 0 & 0 & 0 & 1 & -1 & 1 & 2
\end{array}
\]

[S6]

Basis: \( A_{(2,3,4,6,7)} \)

Column 5 satisfies [3.4.2.1]

Rey: \( r_1 = (1,1/2,-1/2) \)

Adjacent feasible basis: \( A_{(2,3,4,5,7)} \)

\[
\begin{array}{cccccccc}
x^1 & x^2 & x^3 & x^4 & x^5 & x^6 & x^7 \\
0 & 1 & 0 & 0 & 1/2 & -1/2 & 0 & 0 \\
0 & 0 & 1 & 0 & -1/2 & 1/2 & 0 & 0 \\
1 & 0 & 0 & 0 & -1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & -2 & 1 & 0 & 2 \\
0 & 0 & 0 & 0 & 1 & -1 & 1 & 2
\end{array}
\]

[S7]

Basis: \( A_{(2,3,4,7)} \)

Vertex: \( s_1 = (1,1/2,-1/2) \)

No column satisfies [3.4.2.1]

Adjacent feasible bases: \( A_{(2,3,4,6,7)} \) already found

and \( A_{(2,3,4,5,7)} \)

\[
\begin{array}{cccccccc}
x^1 & x^2 & x^3 & x^4 & x^5 & x^6 & x^7 \\
0 & 1 & 0 & 0 & 0 & 0 & -1/2 & -3/2 \\
0 & 0 & 1 & 0 & 1/2 & 0 & 0 & 3/2 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 3 \\
0 & 0 & 0 & 1 & 0 & -1 & 2 & 6 \\
0 & 0 & 0 & 0 & 1 & -1 & 1 & 2
\end{array}
\]

[S6]

Basis: \( A_{(2,3,4,5)} \)
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 \( P_2 \subseteq P_2 \) if and only if each element of the frame of \( P_1 \) satifies the restraints 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 simplifications. 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 constraints on the input variables may be known. In this case they define the assertion associated with the input arc. Otherwise the variables are.

4.2. Assignment.

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

4.2.1. Assignment of a non linear expression.

If the expression \( E(X) \) is not linear it is assumed that any value of \( X \) can be assigned to \( X^1 \).

Hence after the assignment \( X^1_i:=E(X) \) nothing is known about the value of \( X^1 \). Therefore \( X^1 \) is eliminated from the system of restraints \( AXB \) by a projection along the column \( l_1 \) (§ 3.3.1.1). For the frame representation this consists in adding the line \( d \) (such that \( d_j=1 \) and \( d_j=0 \) for every \( j=1..n \) different from \( j_0 \) ) 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:

\[
\begin{align*}
x_1-x_2 & \geq 1 \\
x_2 & \geq 1 \\
x_1x_2 & \geq 5
\end{align*}
\]

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

![Diagram](attachment:image)

The output assertion \( P' \) after the assignment \( x_2=x_1x_2 \) is \( \{s_2 \geq 0, x_1 \geq 2\} \), the frame of which is \( S'=\{(2,3),(4,1)\} \), \( R'=\{(1,1),(1,0)\} \). Simplifying we get \( \{x_1 \geq 2\} \), \( S'=\{(2,3)\} \), \( R'=\{(1,0)\} \).
The approximation is obviously very coarse since the substitution of $x_1x_2$ for $x_2$ in the input system of constraints would lead to:

\[
\begin{align*}
    x_2 &\leq x_1(x_2)^2 \\
    x_2 &\geq x_1 \\
    x_2 &\geq 5x_1-(x_1)^2
\end{align*}
\]

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

Note however that the exact domain is covered by the approximation domain ([Cousot1977]). Also, a more precise analysis is feasible. For example, $x_1 = 0$ and $x_2 > 0$ implies $x_1x_2 = 0$ 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 of the form $X_{1}^b := \sum_{i=1}^{n}(a_iX^i)+b$ 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:

- $S' = \{s_1', \ldots, s_0'\}$ where $s_1'$ is defined by $s_1'^{10} = a_{i1}x_{i1}$ and $s_1'^{1} = s_1'$, where $l=1..n$ and $i=10$.
- $R' = \{r_1', \ldots, r_{1}^{10}\}$ where $r_{1}^{10}$ is the vector defined by $r_{1}^{10} = r_1$ and $r_{1}^{1} = r_1$ for $l=1..n$ and $i=10$.
- $D' = \{d_1', \ldots, d_0'\}$ where $d_1'$ is the vector defined by $d_1'^{10} = ad_k$ and $d_1'^{1} = d_k$ for $l=1..n$ and $i=10$.

The system of constraints corresponding to $P'$ can be obtained as the convex hull of the frame $(S',R',D')$. However, following Karr[1976] this system of constraints can be computed directly.

4.2.2.1. Invertible assignments

The assignment of the form $X_{1}^b := \sum_{i=1}^{n}(a_iX^i)+b + a_{10}x_{10}$. The fact that $a_{10} \neq 0$ allows us to carry over our knowledge of the previous value of $X_{1}^b$ to the new value of $X_{1}^b$. 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 $i=10$, we have $X'^l = X^l$ whereas $X'^{10} = aX + b$. Therefore $X' = MX + k$ as defined by

\[-X^l = X^l \text{ for } 1 \in \{1, n\} \setminus \{l\}\]

- $X'^{10} = (X^{10} - \sum_{l=1}^{n}(a_{l}X^{l}) - b)/a_{10}

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

4.2.2.2. Non-invertible assignments

If $a_{10} = 0$ then we cannot solve $X'$ in terms of $X$ so that some information is lost by the assignment. Hence $X_{1}^b$ is eliminated from the input constraints by a projection operation. The case is similar to the one of the assignment of a non-linear expression except that the constraint $X_{1}^b = aX + b$ is adjoined to the resulting system.

Example:

Let $P$ be the input assertion defined by

\[
\begin{align*}
    x_1 &\geq 1 \\
    x_1 \cdot x_2 &\geq 5 \\
    x_1 \cdot x_2 &\geq -1
\end{align*}
\]

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

\[
\begin{align*}
    x_1 &\geq 2 \\
    x_2 - x_1 &\geq 1
\end{align*}
\]

The assignment $x_2 := x_1 \cdot x_2 / 2 + 1$ is invertible so that $x_1 \cdot x_1$ and $x_2 = 2x_1 \cdot x_2$. Substituting in the input system of constraints we get the output system:

\[
\begin{align*}
    2x_1^2 - 2x_1 &\geq 3 \\
    2x_1^2 &- X_1 &\geq 7 \\
    -2x_1^2 - 3x_1 &\geq -3
\end{align*}
\]

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 \( P_0 = (A_t, B_t, S_t, R_t, D_t) \) 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 \( \neg C \).

The condition \( C \) is said to be linear if and only if it is of the form \( a \leq b \) or \( a \geq b \) 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.

### 4.3.3. Linear inequality tests

If \( C \) is of the form \( a \leq b \) then \( P_t = (P \text{ and } a \leq b) \) whereas \( P_f = (P \text{ and } a \geq b) \). (Note that the inequality is not strict since \( P_f \) must be a closed polyhedron and that we can write \( a \leq b + 1 \) 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 \( H = \{ x \in \mathbb{R}^n : a \cdot x = b \} \).

The set \( S_t \) of vertices of \( P_t \) is \( \{ q \in S : q \cdot S \leq b \} \cup S' \).

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 \( \{ r \in R : a \cdot r \leq 0 \} \cup \{ d : a \cdot d = 0 \} \cup \{ -d : a \cdot d > 0 \} \cup R' \). The set of
\[ R_p = \{r_2, r'\}, D_p = \emptyset. \]

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 \( P_1, P_2, \ldots \)
\( \ldots \)
\( P_p \) be the assertions associated with the input arcs to the simple junction node and \( P \) the output assertion:

\[
\begin{array}{c}
\begin{array}{c}
\text{P}_1 \\
\text{P}_2 \\
\ldots \\
\text{P}_p
\end{array}
\end{array}

\begin{array}{c}
\text{P}
\end{array}
\]

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

Since the convex-hull operation is associative we can assume without loss 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 \( P = \text{convex-hull}(P_1, P_2) \) is \( S_1 S_2 S_2 R_1 R_2 \) and \( D_1 D_2 D_2 \). The system of restraints \( \text{AXS.B} \) 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 \( \text{AXS.B} \) of \( P \) is known the frame \( [S, R, D] \) of \( P \) can be simplified (§ 3.4.4).

Notice that \( \text{convex-hull}(P_1, P_2) \) cannot be computed directly from the systems of restraints \( A_1, B_1, S_1, R_1, D_1 \) and \( A_2, B_2, S_2, R_2, D_2 \). 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 \( \text{AXS.B} \) 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:

\[
\begin{align*}
\text{convex-hull} & \{S_1 S_2 R_1 R_2 \} \\
\text{convex-hull} & \{A_1, B_1, A_2, B_2\} \\
\text{convex-hull} & \{A_1, B_1, A_2, B_2\}
\end{align*}
\]

so that starting from \( A_1, B_1 \), we can successively incorporate in the convex-hull the elements of the frame of \( P_2 \).

Example:
\[
\begin{align*}
P_1 = & \{(x_2 \geq 0, x_2 \geq 0, x_1 \geq 0) \}
S_1 = & \{(0,0),(1,0),(0,1)\}, \\
R_1 = & \{0\} \\
P_2 = & \{(x_2 \geq 1, x_2 \geq 2) \}
S_2 = & \{(1,2)\}, \\
R_2 = & \{(1,0)\}, \\
D_2 = & \{0\}
\end{align*}
\]

The convex-hull of \( P_1 \) (given by \( A_1, B_1 \)) and the vertex \( s = (1,2) \) of \( P_2 \) is obtained (§ 3.3.1.1) by elimination of \( \lambda \) in \( \{S_1 S_1, A_1 X_1 \lambda (A_1 a - B_1 S_2) = A_1\} \).

Eliminating \( \lambda \) in \( \{S_1 S_1, x_1 \lambda x_2, x_1 x_2 x_3 \leq 3\} \) we get the approximation \( A' \times S'B' \) given by:
\[
\{x_2 \leq x_1, D_1 x_1 \leq 1, x_2 \geq 0\}.
\]

Incorporating the ray \( r = (1,0) \) of \( P_2 \) in \( A' \times S'B' \) by elimination of \( \mu \) in \( \{\mu a, A' \times \mu A' \times S'B'\} \) that is \( \{\mu a, x_2 \mu x_1 \mu, D_1 x_1 \mu x_2 \} \) we get the convex-hull of \( P_1 \) and \( P_2 \) given by \( \{D_1 x_2 \leq 2, x_1 \geq 0, x_2 \leq x_1 \leq 1\} \). End of example.

4.5. Loop junction nodes.

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

\[
\begin{array}{c}
\begin{array}{c}
\text{P}_1 \\
\text{P}_2 \\
\ldots \\
\text{P}_p
\end{array}
\end{array}

\begin{array}{c}
\text{P}
\end{array}
\]

The corresponding transformation \( P = P \text{convex-hull}(P_1, P_2, \ldots, P) \) is defined by the widening operation \( V \). 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 \( Q_1 V Q_2 \) 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:
\[
\begin{align*}
Q_1 = & \{(-x_1 + 2x_2 \leq -2, x_1 + 2x_2 \leq 6, x_2 \geq 0)\}, \\
S_1 = & \{(2,0), (6,0), (4,1)\}, R = \emptyset, D = \emptyset \\
Q_2 = & \{(-x_1 + 2x_2 \leq -2, x_1 + 2x_2 \leq 10, x_2 \geq 0)\}, \\
S_2 = & \{(2,0), (10,0), (6,2)\}, R = \emptyset, D = \emptyset \\
Q_1 V Q_2 = & \{(-x_1 + 2x_2 \leq -2, x_2 \geq 0)\}
\end{align*}
\]

End of example.

In order to compute the frame of \( Q_1 V Q_2 \) 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 \( Q_1 V Q_2 \) 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 \( Q_1 V Q_2 \) 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 \( Q_1 V Q_2 \). Therefore the initialization of the simplex method needs not to be used. Moreover since any restraint of \( Q_1 V Q_2 \) is a restraint of \( Q_1 \) it is highly probable that the frames of \( Q_1 \) and \( Q_1 V Q_2 \) have numerous common elements.

The correctness criterion of Cousot [1977] recalled at paragraph 2 is satisfied since \( Q_1 V Q_2 \), \( Q_2 V Q_1 \) and for every chain \( C_0 \leq C_1 \leq \ldots \leq C_n \), the chain \( S_n = C_0, C_1 = S_n \ldots, C_n = S_n \ldots V C_n \), is not an infinite strictly increasing chain since at each step the number of restraints describing \( S_n \) is finite and less than or equal to the number of restraints describing \( S_n \ldots \).
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:

```
(P_0)
  I:=2; J:=0;

(P_1)
L:

(P_2)
  if J > 0

(P_3)
    I:=I+4;

(P_4)
  else

(P_5)
    J:=J+1; I:=I+2;

(P_6)
fi;

(P_7)
  go to L;
```

The test involving some non-linear condition is not taken into account. Each assertion \( P_i \), \( i=0,..,7 \) is initially the empty polyhedron \( \emptyset \) and the input assertion is propagated through the program graph:

\[
P_0 = \emptyset, \quad S = \{(0,0), R = \emptyset, D = \emptyset\}
\]

\[
P_1 = \text{assign}(\text{assign}(P_0, I := 2), J := 0)
\]

\[
P_2 = \text{convex-hull}(P_1, P_0)
\]

\[
P_3 = P_0
\]

\[
P_4 = \text{assign}(P_3, I := I + 4)
\]

\[
P_5 = \text{assign}(P_3, J := J + 1, I := I + 2)
\]

\[
P_6 = \text{assign}(P_5, I := I + 4)
\]

\[
P_7 = \text{assign}(P_6, J := J + 1, I := I + 2)
\]

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

\[
P_3 = P_2 \lor \text{convex-hull}(P_1, P_2)
\]

We have \( P_2 = \{(2I + 2S, I + 2S \leq 0, 0 \leq J) \} \) and the convex hull \( \text{convex-hull}(P_1, P_2) = \{(2I + 2S, I + 2S \leq 0, 0 \leq J) \} \) with \( S = \{(2,0), (1,0), (2,1)\} \), \( R = \emptyset \), \( D = \emptyset \), so that 2I + 2S which is the only constraint of \( P_2 \) not verified by every element of the frame of \( \text{convex-hull}(P_1, P_2) \) is eliminated by the widening operation.
Then \( \text{convex-hull}(P_1, P_2) \) is included in \( P_2 \) so that the program analysis has converged.

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

\[
\begin{align*}
0 & : 	ext{no information} \\
1 & : i = 2, j = 0 \\
2, 3, 5 & : 2j \leq i, j \geq 0 \\
4 & : 2j > 6i, j \geq 0 \\
6 & : 2j > 6i, j \geq 1 \\
7 & : 2j > 6i, 6i > 2j, j \geq 0
\end{align*}
\]

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 div 2)+1; R:=N;
{3}  if (L>=2) then
{4}   L:=L-1; \{K:=T[L]\}
{5}  else
{6}   fi;
{7}  while (R>=2) do
{8}   I:=L; J:=2*I;
{9}   while (J<=R) do
{10}  if (J=R) then
{11}   if (T[I]<T[J+1]) then J:=J+1 fi; fi;
{15}  end { J+2s21>R, 2J+2s4I+N+1, R+3s2N, 1sL, RsN, 2J+6s2R+1s2I+12N, 2I+1s2I+1, LsI, 2L+2s1s3N, 23+2L+1s1s3I+2R+3N
{17}  \{15\}, L\geq 2
{19}  \{15\}, L\leq 2
{21}  \{Rs=1, 2LsN+1, R+4s2N, 2L+2s3s3N, L\geq 1, RsN

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 without 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 inequalities 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 \( \frac{p}{q} \) where
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


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]


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 polyèdre convexe, Thèse de 3ème cycle, Cen, France, [1968]

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


M.SIMONVARD, Programmation linéaire, Dunod, Paris, [1970]


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