Variable Elimination for Disequations in Generalized Linear Constraint Systems

JEAN-LOUIS IMBERT
G.I.A. Faculté des Sciences de Luminy, 163 Avenue de Luminy, case 901,
F-13288 Marseille cedex 9, France

This paper is concerned with the case of generalized linear constraint systems. A
generalized linear constraints system is the conjunction of a sub-system of equations
E, a sub-system of inequations I (≤), and a sub-system of disequations D (≠). We
first of all establish that the variable elimination operation on a generalized linear
constraint system E, I, D has, as its result, a generalized linear constraint system
E′, I′, D′. We then show that E′, I′ does not depend on D, and that the disequations
of D are independent from one another for the variable elimination operation. Next,
we present two algorithms for variable elimination in the disequation sub-system D.
The first algorithm can be easily integrated into an algorithm dealing with variable
elimination in inequation systems by the Fourier elimination method. The variables
are eliminated one after another. The second algorithm assumes that the projection
E′, I′ is known. It is based on new properties that we establish. It globally eliminates
the variables in one single operation. The numerous independencies allow for a high
degree of parallelism which we take advantage of.*

Received January 1993, revised May 1993

1. INTRODUCTION

Variable elimination has a large potential of applications in such domains as robotics, geometry, CLP languages like CHIP [6, 26], CLP(ℜ) [14], Prolog III [3], and systems like Mathematica and Maple. In the middle of this century, Tarski [23], using logical tools, showed that variable elimination in Euclidian Geometry is a decidable problem. Collins [2] has proposed an algorithm for variable elimination from polynomial constraints. However, this algorithm and derived versions published so far, do not allow for the processing of a large number of constraints.

We focus on the limited case of linear constraint systems. This case is chiefly of interest for CLP languages CHIP, CLP(ℜ), Prolog III, and constraint query languages [15], where the elimination of auxiliary variables introduced at the time of a program is desirable. This elimination is almost always suitable for presenting the output. It can also increase the efficiency of the intermediary process.

We classify the constraints in three groups: equations (=), inequations (≤), and disequations (≠). Variable elimination in equation systems is solved in polynomial time by algorithms based on Gaussian elimination. This polynomial time depends on the number of equations and variables. Variable elimination in inequation systems has been extensively investigated [7, 4, 1, 17, 5, 18, 9, 10, 13]. The main problems we face are, on the one hand, the size of the results which can be exponential and, on the other hand, the detection and the suppression of redundancies [24, 16, 21, 12]. As far as disequations are concerned, the only studies, to my knowledge, related to our field are either too general and not very applicable [23, 2, 25], or too partial and only take into account the strict inequality case: (0 ≤ t and 0 ≠ t) [9, pp 27–34 and 97 in French, and the same thing in English with another formulation in 13].

In the general case of disequations, the size of the result can be even worse than that for the inequations, as the following example shows: Let the part of the ℜ³ space be delimited by the five following points (1, 0, 0), (0, 1, 0), (0, 0, 1), (1, 0, 0), (0, −1, 0), except for the points of the form (a, b, 0), where a and b are real values. It is a volume defined by the vertices of a square in a plane with a point above, less the points of the plane containing the square. This volume is represented in an equivalent manner by the constraint system
\[
\{0 ≤ z, x + y + z ≤ 1, −x + y + z ≤ 1, x − y + z ≤ 1, −x − y + z ≤ 1, z ≠ 0\}.
\]

*This paper is an extended version of [11].

THE COMPUTER JOURNAL, Vol. 36, No. 5, 1993
It is well known that variable elimination can be viewed geometrically as a projection operation. Here, for example, to eliminate the variable \( z \) is to project onto the plane \( z = 0 \). In this case we obtain the system
\[
\{ -1 \leq x + y \leq 1, -1 \leq x - y \leq 1, x + y \neq -1, x - y \neq 1, x - y \neq 0 \},
\]
which represents the interior of a square. Notice that the number of disequations has been multiplied.

Our approach tackles the general problem of variable elimination in generalized linear constraint systems. A generalized linear constraint system is composed of a sub-system of equations \( E (\equiv) \), a sub-system of inequations \( I (\leq) \), and a sub-system of disequations \( D (\neq) \). We first of all establish that the variable elimination operation on a generalized system of linear constraints \( E, I, D^1 \) has, as its result, a generalized system of linear constraints \( E', I', D' \). We then show that \( E', I' \) does not depend on \( D \), and that the disequancies of \( D \) are independent from one another for the variable elimination operation. Since there is a large amount of literature on variable elimination in equation and inequation systems, we then focus on disequations. This elimination depends on the subsystems of equations and of inequations. There are two possible ways: The first uses the relations between the constraints of the initial subsystem \( E \). It is close enough to the Fourier variable elimination which is used in most of the algorithms to process inequation systems. The variables are eliminated one after another. We present an algorithm based on this approach. This algorithm becomes not very efficient as soon as the number of variables to be eliminated increases. However, we will use it as a basis to prove the correctness of the second approach.

In contrast with the first approach, which rests on purely syntactic properties, the second approach is based on the semantic properties of the projection. It uses the relations between the constraints of the projected system \( E', I' \) assumed known. The variables to be suppressed are eliminated in one single operation, as a whole. The advantages of this approach are, on the one hand, to process less constraints than in the previous approach \( (E', I') \) can be assumed without redundancies, on the other hand, to output a system simplified as far as possible, and in a canonical form defined in [20], which can be suitable for CLP languages. Moreover, it can be applied independently of the method used for the elimination in inequation systems, (for example the one defined in [18]). The results obtained by this approach allow for a large degree of parallelism that has been taken advantage of in an algorithm based on such an approach.

The rest of the paper is organized as follows. Section 2 defines the main concepts, tools and results necessary for understanding the other sections and the adjoining proofs. A brief outline of the variable elimination methods on equation and inequation systems is given. In Section 3, we establish, on the one hand, that the projection of a generalized system is a generalized system and, on the other hand, that the variable elimination in the sub-system of equations and inequations does not depend on the disequations. We also show that the disequations are independent from one another for the variable elimination operation. Section 4 presents an incremental algorithm and discusses some redundancy problems. Section 5 establishes the results for the global variable elimination, and presents a parallel algorithm based on these results. Finally, we conclude with some remarks on complexity and present some directions for future work.

2. PRELIMINARIES

2.1. Constraints

Generalized linear constraints are, firstly, positive constraints which are equations \( ax = b \) and weak inequations \( ax \leq b \). Here \( a \) denotes an \( n \)-dimensional real vector, \( x \) an \( n \)-dimensional vector of variables, \( b \) a real number, and the juxtaposition \( ax \) denotes the inner product. Lastly, negative constraints which are disjunctions of inequations of the form \( a_i x \neq b_i, i = 1 \ldots m \).

A system is a conjunction of constraints. Using matrix notation and De Morgan's Law, an equation system can be written \( Ex = f \), an inequation system can be written \( Ax \leq b \), and a negative constraint can be written as the negation of an equation system: \( Cx = d \).

Here, \( E, A \) and \( C \) denote real matrices, and \( f, b \) and \( d \) are real vectors. In the sequel, inequation means weak inequation, and a negative constraint is called disequation. An equation can be understood as a line of a system \( Ex = f \), and an inequation can be understood as a line of a system \( Ax \leq b \). A generalized linear constraint system is comprised of an equation system \( Ex = f \), a system of weak inequations \( Ax \leq b \), and a disequation system \( \{ C_1 x = d_1, \ldots, C_m x = d_m \} \).

A variable \( x_k \) occurs in a constraint \( ax = b \) or \( ax \leq b \), if the component \( a_k \) of vector \( a \) is non-zero. A variable occurs in a constraint system if it occurs in at least one of its constraints.

In the sequel, \( V = \{ x_1, x_2, \ldots, x_n \} \), denotes the set of variables occurring in the generalized constraint system, \( V' \) is a subset of \( V \), \( x \) denotes the vector \( (x_1, x_2, \ldots, x_n) \), \( x' \) denotes the vector of variables of \( V' \) and \( x'' \) denotes the vector of remaining variables of \( V \). We abuse the language by writing \( x = (x', x'') \).

An assignment of the variable vector \( x \) is a real vector \( (v_1, v_2, \ldots, v_n) \), such that, for each \( i \), \( v_i \) corresponds to \( x_i \). A solution of a constraint system is an assignment such that, substituting each \( v_i \) for each \( x_i \) in this system, we obtain a constraint system trivially satisfied. A solution of a constraint means a solution of the system of that single constraint. A constraint system which has at least one solution is said to be satisfiable, solvable or

\[1\] Here, comma is for conjunction of constraint sets. When the context is not clear, we write \((E, I, D)\) in place of \(E, I, D\).
consistent. The solution set of $S$ is denoted by $\text{Sol}(S)$. Two constraint systems are equivalent if they have the same solution set. $S \equiv S'$ denotes that $S$ and $S'$ are equivalent. Two constraint systems are equivalent on $V'$ if, for each solution $s$ of one, there is a solution of the other which coincides with $s$ on $V'$, and vice versa. This can be written $\exists x' : S \equiv \exists x' : S'$.

The elimination of variables of $V - V'$ in the constraint system $S$, consists in finding a constraint system $S'$ which only variables of $V'$ occur in, and such that $S$ and $S'$ are equivalent on $V'$. This can be written $S' \equiv \exists x' : S$. Then, determining whether $S$ is solvable is equivalent to determining whether $S'$ is solvable. $\text{Sol}(S')$ is called the projection of $\text{Sol}(S)$ on $V'$. By abuse of the language we will say projection of $\text{Sol}(S)$ on $V'$. Variable elimination can be viewed either as a decision process to decide if a constraint system is satisfiable, or as a process to simplify a satisfiable system by eliminating undesirable variables.

$S \models R$ means each solution of $S$ is a solution of $R$. $C$ is a redundant constraint of $S$, if $S$ and $S - \{C\}$ are equivalent, that is to say $S - \{C\} \models C$.

### 2.2. Geometric Outline

In the following, we need some definitions and properties related to convex spaces. If $V$ is a subspace of an $n$-dimensional Euclidean space $\mathbb{R}^d$, and if $p$ is a vector of $\mathbb{R}^d$, then the translation $p + V$ is called an affine space. The intersection of all affine spaces which contain a subset $X$ of $\mathbb{R}^d$ is again an affine space and is called the affine hull or affine closure of $X$, and is denoted $\text{Aff}(X)$. If $x = b$ is a linear equation, the set of points $H = \{x : ax = b\}$ is called a hyperplane. Thus, in $\mathbb{R}^3$, hyperplanes are planes, and in case of $\mathbb{R}^2$, straight lines are hyperplanes. A hyperplane is an affine space and it can be shown that every affine space is the intersection of a finite number of hyperplanes. In practice, an affine space is the solution set of a linear equation system $Ax = b$, and its dimension is defined by $\text{dim}(A)$.

A subset $X$ of $\mathbb{R}^d$ is said to be convex if the line segment joining any pair of points in $X$ is included in $X$. In particular, affine sub-spaces of $\mathbb{R}^d$ are convex, as are the half-spaces defined by inequalities $ax \leq b$. Moreover, the intersection of a family of convex sets is again convex. For example, $\text{sol}(Ax \leq b)$ is a convex set (finite intersection of half-spaces $\{x : a_i x \leq b_i\}$) and is called a polyhedral set. The affine hull or affine closure of $X$, denoted $\text{Aff}(X)$, is the intersection of all the affine spaces which include $X$. The dimension of a convex set is defined as that of its affine closure. In [20] the following results are shown:

**Lemma 2.1.** A convex set has a non-empty relative interior in its affine hull.

It can be shown that the affine hull of a polyhedral set defined by an inequation system $S$ is the solution set of the sub-system of its implicit equalities (its inequations $ax \leq b$ such that $S \models ax = b$). By virtue of the definition of convex set dimension, a convex set is said to be full dimensional, if its affine hull is $\mathbb{R}^d$. In particular, the convex hull of a positive linear constraint system is full dimensional if it is an inequation system without implicit equalities.

In the sequel, and without loss of generality, we assume that in a generalized linear system the sub-system of inequations is full dimensional. This is what is really the main constraint in logic programming languages which process linear arithmetic constraints on $\mathbb{Q}$ or/and $\mathbb{R}$. CHIP, CLP($\mathbb{R}$) and Prolog III. These languages systematically detect the implicit equalities.

Suppose now, we have the constraint system $\{0 \leq x, 0 \leq y, x + y \neq 0\}$. It is immediately clear that the only point which satisfies the system $\{0 \leq x, 0 \leq y, x + y \neq 0\}$ is $(0, 0)$. In other words, the negative constraint $\{x + y = 0\}$ lets one wrongly think it prohibits a straight line of points. This is due to the lack of precision of the disequation. In the initial system, the disequation $\{x + y = 0\}$ can be replaced by $\{x = 0, y = 0\}$ to obtain an equivalent system. The new disequation makes the reality more visible than the previous one because it is more precise.

Let $S$ be a satisfiable positive linear constraint system. The disequation $Cz = d$ is said to be relevant in $S$ if the system $S, Cx = d$ is satisfiable. That is to say, $S \nvdash Cz = d$. The disequation $Cz = d$ is said to be precise in $S$ (or precise in unambiguous), if $Cz = d$ is relevant in $S$ and if $\{x | Cx = d\} = \text{Aff}(\text{sol}(S, Cx = d))$. One of the motivations for the introduction of precise negative constraints is that they allow us to obtain a unique canonical representation of a generalized linear constraint system (see Section 2.3).

**Lemma 2.2.** Let $S$ be a consistent positive linear constraint system. Assume that in the generalized system $S, \{C_1 x = d_1, \ldots, C_m x = d_m\}$, all the disequations are precise. Then the disequation $C_j x = d_j$ is redundant if and only if there exists a disequation $C_j x = d_j$ such that

$$
\text{Aff}(\text{sol}(S, C_j x = d_j)) \subset \text{Aff}(\text{sol}(S, C_j x = d_j)).
$$

The following lemma will be useful to prove the independence of positive constraints for variable elimination relative to disequations.

---

LEMMA 2.3. (Density of positive constraints)
Let \( Ex = f, Ax \leq b, \{\bar{C}_1 x = d_1, \ldots, \bar{C}_m x = d_m\} \) be a consistent generalized linear constraint system which defines the solution set \( Q \). Let \( P \) be the polyhedral set defined by the positive constraint sub-system \( Ex = f, Ax \leq b \). Then \( P \) is the topological closure of \( Q \).

2.3. Solved Form

An equation system \( Ex = f \) is in solved form if in each equation of that system there occurs a variable which occurs nowhere else in \( Ex = f \). These variables which occur only once in \( Ex = f \) are said to be basic, bound or eliminable. It is known that a linear equation system is consistent if and only if it can be put into solved form. Let \( Ex = f \) be a system of \( m \) linear equations, in solved form. A basis of \( Ex = f \) is a set of \( m \) bound variables, one for each equation. A basis is not unique. However, it can be shown that two equivalent linear equation systems with the same bases are identically written up to the order of variables and equations, and up to multiplying by a non-zero scalar for each equation. This can be easily overcome with a normalization of coefficients of basic variables, and with an ordering of the basic variables. There is a classical algorithm for deciding whether a linear system \( Ex = f \) is satisfiable and if it is, for computing an equivalent normalized solved form [8]. This algorithm works in terms of a fixed ordering on the variables which occur in the system, and if two sets of equations define the same affine space, this procedure returns an identical solved form. In the following, \texttt{SolvedForm} \((Ex = f)\) denotes this algorithm, and we assume it returns a value \texttt{Inconsistent} if the system is unsolvable.

One of the motivations for introducing systems in solved form is that by assigning arbitrary values to each non-basic variable, a solution of the system is immediately obtained.

Notice that for equation systems \( S \) and \( S' \) given in solved form by means of the \texttt{SolvedForm} procedure, there are simple linear algebraic algorithms for testing inclusion \( \text{Sol}(S) \subseteq \text{Sol}(S') \). Let us remark that \( \text{Sol}(S) \subseteq \text{Sol}(S') \) and \( S \models S' \) have the same meaning.

In [20], Jean-Louis Laszze and Ken MacAlone define a canonical form for generalized linear constraint systems. It assumes a fixed ordering on the variables which occur in the system. In this canonical form, the sub-system of equations is given in solved form by means of the \texttt{SolvedForm} procedure, and none of its basic variables occur in the inequations and disequations. Each disequation \( \bar{C}x = d \) is in precise form, and \( Cx = d \) is in solved form. Lastly, the sub-system of inequations does not contain any implicit equality or redundant constraint.

In this canonical form, two equivalent generalized systems are identically written. In the sequel, an equation system in solved form is assumed to be obtained by means of the \texttt{SolvedForm} procedure, and every generalized linear constraint system is assumed to be in canonical form.

Let \( S \) be a system of equations. To simplify notations we will write \( \text{AffSol}(S) \) in place of \( \text{Aff}(\text{Sol}(S)) \). Moreover, if \( P \) is a point set, \( \text{EqAff}(P) \) denotes the equation system in solved form corresponding to the solution set \( \text{Aff}(P) \), and \( \text{EqAffSol}(S) \) denotes this same system if \( P = \text{Sol}(S) \).

2.4. Variable Elimination in Equation Systems

Let \( V \) be the set of variables which occur in the system and let \( W \) be the subset of variables to be eliminated. Let us choose an ordering on \( V \) with preference on variables to be eliminated. Using the canonical form of Laszze and MacAlone, the equation system \( Ex = f \) is the union of a sub-system \( E'x = f' \) in which only variables of \( V - W \) occur, and a sub-system \( E''x = f'' \) in which each basic variable is in \( W \). It can be shown that the systems

\[
Ex = f, Ax \leq b, \{\bar{C}_1 x = d_1, \ldots, \bar{C}_m x = d_m\}
\]

and

\[
E'x = f', Ax \leq b, \{\bar{C}_1 x = d_1, \ldots, \bar{C}_m x = d_m\}
\]

are equivalent on the set of variables which occur in the system (3).

2.5. Variable Elimination in Inequation Systems

Variable elimination in inequation systems has been extensively investigated [7, 4, 1, 17, 5, 18, 9, 10, 13]. Almost all of them use Fourier's elimination. We will briefly expose this method for understanding the incremental algorithm presented in Section 4.1. However, we strongly encourage the reader to look at the work of Laszze [18], which is a real breakthrough in terms of complexity.

2.5.1. Fourier's elimination

Let \( a_1 x \leq b_1 \) and \( a_2 x \leq b_2 \) be two inequations, and let \( a_{1,1} \) and \( a_{2,1} \) be the coefficients of the variable \( x_1 \) respectively in the first and second inequations. Let us assume that \( a_{1,1} > 0 \) and \( a_{2,1} < 0 \). Then,

\[
-a_{2,1}(a_1 x) + a_{1,1}(a_2 x) \leq -a_{2,1}b_1 + a_{1,1}b_2
\]

is a consequence of the two initial inequations, and \( x_1 \) does not occur in it. Let \( Ax \leq b \) be an inequation system, and \( V \) the variables which occur in it. Let \( A'x \leq b' \) be the system obtained from \( Ax \leq b \), by deleting all inequations which \( x_1 \) occurs in and replacing them by all the inequations of type (4) above we can construct from any pair of deleted inequations. It can be proved that \( A'x \leq b' \) and \( Ax \leq b \) are equivalent on \( V - \{x_1 \} \). This operation must be repeated successively for each variable to be eliminated.
2.5.2. Some results

Assume that $S = \{a_1 x \leq b_1, \ldots, a_n x \leq b_n\}$. It is known [22, pp 87–90], that $S \models cx \leq d$ if and only if $cx \leq d$ is an affine combinatorial with positive coefficients of inequalities of $S$ (i.e. $\exists \alpha \geq 0, i = 0, \ldots, n, c = \sum_{i=1}^{n} \alpha a_i$, and $d = \alpha_0 + \sum_{i=1}^{n} \alpha_i b_i$). A linear combinatorial is an affine combinatorial such that $\alpha_0 = 0$. Hence, an inequation of $S$ is redundant in $S$ if and only if an affine combinatorial with positive coefficients of the other inequations of $S$. It is strongly redundant if for at least one affine combinatorial $\alpha_0 \neq 0$, weakly redundant in other cases.

Lastly, let $S_w$ be a system obtained after eliminating the variables of $W$ in $S$. From Fourier’s elimination, each inequation of $S_w$ is a linear combinatorial with positive coefficients of the inequalities of the inequations of $S$. As a result, if $S$ is full dimensional, then $S_w$ is full dimensional.

3. VARIABLE ELIMINATION IN GENERALIZED LINEAR SYSTEMS

In this section, we establish (Theorem 3.1) that the variable elimination operation on a generalized linear constraint system $E, I, D, H$ has as result a generalized linear constraint system $E', I', D'$, and we show that $E', I'$ does not depend on $D$. Then, from the numerous known results for variable elimination on equation and inequation systems, only the disequation case is left. In Theorem 3.2 we show that the disequations of $D$ are independent from one another in the variable elimination operation. This last result generalizes the theorem of independence of negative constraints in a more general case of variable elimination. It makes it possible to separately process each disequation.

The following results have a bearing on topological properties. It is well known that in $\mathbb{R}^n$, all norms are equivalent. In the proofs we will use the norm: $|x|_\infty = \sup_i |a_i|$, and $B(x, \epsilon)$ will denote the neighbourhood of $x$ defined by $\{y : |x - y|_\infty < \epsilon\}$. Let us remark that $B(x, \epsilon)$ is the topological interior of a full dimensional polyhedron set. By an abuse of notation, we will write $|x|$ in place of $|x|_\infty$.

**Remark 3.1.** Let $S$ be a consistent generalized linear constraint system, as defined in formula (2), and let $V'$ be a part of $V$. We want to eliminate the variables of $V - V'$. Section 2.4 allows us to obtain the system of formula (3), which is equivalent to $S$ on a part $V''$ which includes $V'$. This last system is assumed to be in canonical form. Hence, its equation sub-system is in solved form and its basic variables do not occur in the rest of the system. The inequation sub-system is full dimensional in the affine space corresponding to the variables which are not basic in the equation sub-system. The elimination of variables of $V'' - V'$ no longer depends on the equation sub-system, since from Section 2.3, every solution of the inequation and disequation sub-system gives a solution of the system (3).

**Theorem 3.1. (Independence of Positive Constraints/Disequations)** Let $S, D$ be a generalized linear constraint system, where $S$ is the sub-system of positive constraints, and $D$ a set of disequations. Let $W$ be a set of variables. There is then a generalized linear constraint system $S', D'$ equivalent to $S, D$ on $V - W$, such that $S'$ is equivalent to $S$ on $V - W$.

**Proof.** According to remark 3.1, one can limit oneself to the case where $S$ is an inequation system, and where the system $S, D$ is in canonical form. Let $P$ be the polyhedral set defined by $S$. Let $S_w$ denote the system equivalent to $S$ on $V - W$, and let $P_w$ denote the polyhedral set defined by $S_w$. Here, $\text{Proj}_w$ denotes the projection of $\text{Sol}(S, D)$ on $V - W$. Then, $P_w$ is the topological closure of $\text{Proj}_w$. This comes from the fact that, for each point $a'$ of $P_w$, there is $a''$ such that $a = (a', a'')$ is in $P$ and, according to Lemma 2.3, $P$ is the topological closure of $\text{Sol}(S, D)$. Hence for every $\epsilon > 0$ there is $y = (y', y'')$ of $\text{Sol}(S, D)$ such that $|y - a| < \epsilon$ thus there is $y' \in \text{Proj}_w$ such that $|y' - a'| < \epsilon$.

Now, let $a'$ be a point of $P_w - \text{Proj}_w$, and consider the equation system $x' = a'$. Let $X$ denote the polyhedral set defined by $\{a'\}$. For each $a_j \in (a', a'')$ in $\text{Sol}(S)$, there is a disequation $d_j$ of $D$ such that $a_j$ is an element of $\text{Sol}(d_j)$. Let $X_j$ denote the polyhedral set $X \cap \text{Sol}(d_j)$. Then, $X \subset X_j$, and since a convex set cannot be included in a union of convex sets all with dimensions strictly less than itself [20], there is $j$ such that $X \subset X_j$, that is to say, such that $\text{Aff}(X) \subset \text{Sol}(d_j)$ and then such that $\text{EqAff}(X) \models d_j$.

Assume that $R$ denotes the system of equations $\{ux = v : ux \leq v \in S, \text{EqAff}(X) \models ux = v\}$. Then $a'$ is a solution of any system equivalent to $(R, d_j)$ on $V - W$. The choice of $d_j$ is not unique. Then the same $a'$ can be associated with several pairs $(R, d_j)$. Since the systems $S$ and $D$ are finite, there is a finite number of pairs $(R, d_j)$. Moreover, if $a'$ and $b'$ are two different points of $P_w - \text{Proj}_w$ associated with the same pair $(R, d_j)$, then each point of the line segment joining $a'$ to $b'$ in $P_w - \text{Proj}_w$ is associated with the same pair. As a result, there is a finite number of linear equation systems on variables of $V - W$ such that each point of $P_w - \text{Proj}_w$ is a solution of at least one of these equation systems. Moreover, as a consequence of their construction, the solutions of these systems are in $P_w - \text{Proj}_w$.

Hence, $\text{Proj}_w$ is a generalized linear constraint system $S', D'$. According to lemma 2.3, $\text{Sol}(S')$ is the topological closure of $\text{Sol}(S', D')$, as is $P_w$, and $S'$ is equivalent to $S$ on $V - W$. 

Let $S$ be a generalized linear constraint system. According to the unicity of the canonical form, $\text{Proj}_w(S)$ will denote the generalized linear constraint system which results from the elimination of the variables of $W$. 

---

THEOREM 3.2. (INDEPENDENCE OF DISEQUATIONS IN THE ELIMINATION) Let
\[ S, \{C_1 x = d_1, \ldots, C_m x = d_m\} \]
be a generalized linear constraint system, where \( S \) is the sub-system of positive constraints. Let \( W \) be a set of variables. Then,
\[ \text{Proj}_w(S, \{C_1 x = d_1, \ldots, C_m x = d_m\}) = \bigcup_{j=1}^{j=m} \text{Proj}_w(S, \{C_j x = d_j\}). \]

Proof. According to remark 3.1, it suffices to consider \( S \) as a full dimensional inequation system. Let \( D \) denote the set of disequations \( \{C_1 x = d_1, \ldots, C_m x = d_m\} \). Let \( S_w, D_w \) be the projection of \( S, D \), and let \( S_w, D_j \) be the projection of \( S, \{C_j x = d_j\} \) (the projection of positive constraints are identical according to theorem 3.1). It is obvious that every solution \( x' \) of \( S_w, D_w \) is a solution of each \( S_w, D_j \), and then of \( \bigcup_{j=1}^{j=m} \text{Proj}_w(S, \{C_j x = d_j\}) \). This comes from the fact that there is a solution \( (v', v'') \) of \( S, D \). Reciprocally, let \( x' \) be a solution of \( \bigcup_{j=1}^{j=m} \text{Proj}_w(S, \{C_j x = d_j\}) \), then for each \( j \) of \( \{1, \ldots, m\} \), there is \( v''_j \) such that \( (v'_j, v''_j) \) is a solution of \( S, \{C_j x = d_j\} \). Two cases are then possible: (1) For every \( i \) and \( j \), \( v''_j = v'_j \). Then \( x' \) is a solution of \( S_w, D_w \). (2) There is \( i \) and \( j \) such that \( v''_j \neq v'_j \), then there is \( v''_l \) in the line segment joining \( v''_l \) to \( v'_j \) such that \( (v'_l, v''_l) \) is a solution of \( S, \{C_j x = d_j\} \). Otherwise, either \( (v', v'') \) is not a solution of \( S, \{C_i x = d_i\} \), or \( (v', v'') \) is not a solution of \( S, \{C_i x = d_i\} \). Since \( D \) is finite, there is \( v'' \) such that \( (v', v'') \) is a solution of \( S, D \). Hence, \( x' \) is a solution of \( S_w, D_w \).

An interesting result we will show (lemma 3.5) is that if a disequation of \( D_w \) has as solution an interior point of \( \text{Sol}(S_w) \), then this disequation is in \( D \cap D_w \).

LEMMA 3.3. Let \( P \) be a polyhedral set defined by a full dimensional inequation system. Let \( x \) be an interior point of \( P \), and let \( y \) be an arbitrary point of \( P \). Then, each point of the line segment joining \( x \) to \( y \) is an interior point of \( P \), except perhaps \( y \).

Proof. This comes from the fact that there is an open set \( B(x, \epsilon) \) included in \( P \) and that for each non-zero positive real value \( \alpha \),
\[ B(y + \alpha(x - y), \alpha \epsilon) = \{ y + \alpha(z - y) | z \in B(x, \epsilon) \}. \]

LEMMA 3.4. Let \( P \) be a polyhedral set defined by a full dimensional inequation system. Let \( W \) be a subset of variables of \( V \), and let \( P_w \) be the projection of \( P \) on \( V - W \). Then, every interior point of \( P \) has as projection an interior point of \( P_w \), and reciprocally, every interior point of \( P_w \) is the projection of an interior point of \( P \).

Proof. Let \( S \) be the inequation system which defines \( P \). Let \( P \) and \( P_w \) denote the interior of \( P \) and \( P_w \) respectively. Let \( B_w(x', \epsilon) \) denote a neighbourhood of \( x' \in P_w \). We have to show that \( \exists x''(x', x'') \in P \iff x' \in P_w \). \( \iff \rightarrow \) : Let \( a = (a', a'') \) be a point of \( P \), and let \( B(a, \epsilon) \) be included in \( P \). Then, for each \( b' \) in \( B_w(a', \epsilon) \), \( (b', a'') \) is in \( B(a, \epsilon) \). Thus \( b' \) is in \( P_w \), and so is \( a' \). \( \iff \leftarrow \) : Let \( a' \) be a point of \( P_w \), and let \( B_w(a', \epsilon) \) be included in \( P_w \). According to the definition of projection, there is an \( a'' \) such that \( a = (a', a'') \) is in \( P \). By lemma 2.1, there is an interior point \( b = (b', b'') \) of \( P \) such that \( |b - a| < \epsilon \). In cases where \( b' = a' \), it is finished. Otherwise, \( b' \) is defined by \( b' = b' + 2(a' - b) \). The points \( b' \) and \( b' \) are in \( B_w(a', \epsilon) \). We know there is a point \( c' = (c', c'') \) in \( P \). In accordance with lemma 3.3, the point \( (b + c)/2 \) is an interior point of \( P \). And this point is nothing but \( (a', (b' + c')/2) \).

LEMMA 3.5. Let \( S, D \) be a generalized linear constraint system, where \( S \) is a full dimensional inequation system, and \( D \) is a disequation set. Let \( P \) be the polyhedral set defined by \( S \). Let \( W \) be a set of variables, and let \( P_w \) be the polyhedral set defined by \( \text{Proj}_w(S) \). Then, if \( a' \) is an interior point of \( P_w \) which is not in \( \text{Proj}_w(S, D) \), there is a disequation \( C x = d \) of \( D \cap D_w \) such that \( a' \) is a solution of \( C x = d \).

Proof. According to lemma 3.4, \( S \) has an interior point \( (a', a'') \). Hence, the system defined by \( R \) in the proof of theorem 3.1, is empty, and none of the variables of \( W \) occurs in \( d_j \).

In all that follows, by virtue of remark 3.1, a generalized linear constraint system will be limited to a system \( S, D \), where \( S \) is a full dimensional inequation system \( Ax \leq b \), and \( D \) is a disequation set. \( d \) will denote a disequation, and thus \( d \) represents an equation system (in solved form). Let \( W \) be a subset of variables of \( V \) to be eliminated. Then \( V' \) denotes the complementary of \( W \) in \( V \), \( S_w \) denotes \( \text{Proj}_w(S) \) and \( S_w D_w \) denotes \( \text{Proj}_w(S, D) \).

Moreover, we will often be led to speak of the inequation \( ax \leq b \) transformed into the equation \( ax = b \). We will note as \( i \) the equation obtained in transforming the inequation \( i \). In the same way \( i \) will denote the inequation system \( \{i | i \in I\} \) where \( I \) is an inequation system.

4. INCREMENTAL ELIMINATION IN THE DISEQUATIONS

In the proof of theorem 3.1, we have seen that the disequations of \( D_w \) are obtained from \( D \) and the inequations of \( S \) transformed into equations. In this section, we specify the sets of inequations of \( S \) to be chosen. We assume that the variable elimination algorithms in positive linear constraint systems are known. The algorithm
that we present takes as input, on the one hand, a generalized system in canonical form \( S, D \) only containing inequations and disequations, and, on the other hand, a set of variables \( W \) and it produces as output the set of disequations \( D_w \). This algorithm can be inserted into an algorithm which eliminates variables from a system of inequations by the Fourier elimination method. We will refer to this algorithm calling it incremental algorithm associated with Fourier’s elimination. The end of this section is devoted to the problems raised by redundancies, and constitutes an introduction (not indispensable) to the global elimination presented in Section 5.

### 4.1. Incremental Algorithm Associated with Fourier’s Elimination

The idea exploited here is the following: Let \( P, P_w, Q \) and \( Q_w \) be the sets of solutions respectively for \( S, S_w, (SD) \) and \( (S_wD_w) \). According to lemmas 3.4 and 3.5, if a point of \( P_w - Q_w \) is the projection of an interior point of \( S \), then it satisfies the negation of a disequation of \( D \cup D_w \), i.e. contains no variable of \( W \). The other points of \( P_w - Q_w \) are on the boundary of \( P_w \), and are projections uniquely of boundary points of \( P \). From this fact, if \( u' \) is such a point, there is at least an inequation \( ax' \leq b \) of \( S_w \) such that \( au'' = b \). As we have seen in Section 2.5.2, all inequations of \( S_w \) are linear compositive with positive coefficients of inequations of \( S \) and in the case of the elimination of a single variable, this number is at most two \([1, 10]\). These inequations \( a_x \leq b \) satisfy the equalities \( a(u', u'') = b \) for each point \((u', u'')\) of \( P \). Consequently, in the case of \( W = \{x_0\} \), we associate one disequation with each pair of inequations of \( S \) with non-zero and opposite sign coefficients of \( x_0 \). We again find here the foundation of Fourier’s elimination. The incremental algorithm is given in figure 1.

**Remark 4.1.** Let us assume that each disequation is precise in \( S(x_0) \). By lemma 2.2, a disequation \( x \) of \( D(x_0) \) is redundant if and only if there is a disequation \( c \) of \( D(x_0) \) such that \( EqAffSol(S(x_0), c) = \emptyset \). Since the equation systems are in solved form, this is possible only in the two following cases: either (1) \( EqAffSol(S(x_0), c) \) contains more equations than \( EqAffSol(S(x_0), c') \), or (2) these two systems are identical. This greatly reduces the investigation field.

In the three following examples, \( y \) is the variable to be eliminated.

**Example 4.1.** Let us consider the systems \( S = \{x \leq y, y \leq z\} \) and \( D = \{y = 0\} \), i.e. \( y \neq 0 \). Then, \( S_{[y]} = \{z \leq x\} \) and \( D_{[y]} = \{x = 0, z = 0\} \). The solution set is a half-space with the origin point deleted.

**Example 4.2.** Let us consider the systems \( S = \{x \leq y, y \leq z\} \) and \( D = \{y = x + 1\} \), i.e. \( y \neq x + 1 \).

Then, \( S_{[y]} = \{x \leq z\} \) and the disequation of \( D \) gives the system \( \{x = x + 1, z = x + 1\} \) which is inconsistent. Thus, \( D_{[y]} = \emptyset \).

**Example 4.3.** Let us consider the systems \( S = \{y \leq x + 1, y \leq 5 - x, x - 1 \leq y, 3 - x \leq y\} \) and \( D = \{y \neq 2\} \). Then, \( S_{[y]} = \{y \leq x, x \leq 3\} \) and \( D_{[y]} = \{y \neq 1, x \neq 3\} \).

The first comes from \( \text{Proj}(S)\{y = 2, y = x + 1, 3 - x = y\} \), and the second from \( \text{Proj}(S)\{y = 2, y = 5 - x, x - 1 = y\} \).

The two other inequation pairs give inconsistent equation systems.

**Proof.** (Of the algorithm) According to the definition of variable elimination, we have to show

\[
\exists x_0, (x', x_0) \in \text{Sol}(S,D) \quad \exists x' \in \text{Sol}(S(x_0[D(x_0)])).
\]

By theorem 3.2, it suffices to consider the case \( D = \{d\} \).

\( \models \) : Whereas \( S(x_0) = \text{Proj}(x_0)(S) \), from the classical formulae transformations it suffices to show

\[
(x', x_0) \in \text{Sol}(S) \quad \implies \\
(\forall \overline{d} \in D_{[x_0]}, \exists \overline{d} \in D, (x' \in \text{Sol}(\overline{d}) \implies (x', x_0) \in \text{Sol}(d))).
\]

Assume that \( (x', x_0) \in \text{Sol}(S,D,d) \) and \( \overline{d} \in D_{[x_0]} \) such that \( x' \in \text{Sol}(\overline{d}) \). If \( \overline{d} \in D \), it is trivial. Else, there are two inequations of \( S \) which can be equivalently written \( j(x') \leq x_0 \) and \( x_0 \leq i(x') \), such that \( d' = \text{Proj}(j)(d, \{j(x') = x_0, i(x') = x_0\}) \).

Moreover \( x_0 \) occurs in at least one equation of \( d \), which can be equivalently written \( x_0 = c_0 + c_x x' \). Every solution of \( d' \) is a solution of \( \text{Proj}(j)(d, \{j(x') = x_0, i(x') = x_0\}) \) and is then a solution of \( j(x') = i(x') \). Consequently \( x_0 \) can only take the value \( i(x') \). Then, \( (x', x_0) \in \text{Sol}(d) \).

\( \models \) : Assume that \( x' \in \text{Sol}(S(x_0[D(x_0)]) \).

For \( D = D_{[x_0]} \), it is immediate. In other cases \( x_0 \) occurs in at least one equation of \( d \). Then there is only one solution \( (x', x_0) \) of \( d \). Every inequation of \( S \) which \( x_0 \) occurs in, can be equivalently written either \( j(x') \leq x_0 \) or \( x_0 \leq i(x') \).

Whereas \( x' \) is a solution of \( S(x_0) \), two cases can happen:

1. There are two inequations of \( S \), denoted by \( j(x') \leq x_0 \) and \( x_0 \leq i(x') \) such that:

\[
j(x') = \inf j(x') \leq i(x') = \inf i(x').
\]

If \( j(x') = i(x') \), \( x_0 \) can only take one value, and \( (x', x_0) \) is a solution of \( D \) (otherwise we have a contradiction with \( x' \in \text{Sol}(S(x_0,D(x_0))) \)). Else, \( j(x') < i(x') \), and since \( D \) is finite, there is always a value \( x_0 \) between \( j(x') \) and \( i(x') \) such that \( (x', x_0) \) satisfies \( D \).

2. \( S \) only contains inequations of type \( j(x') \leq x_0 \) (resp. \( x_0 \leq i(x') \)), then there are infinitely many values \( x_0 \) which satisfy \( j(x') \leq x_0 \) (resp. \( x_0 \leq i(x') \)). Then \( (x', x_0) \in \text{Sol}(D) \).

### 4.2. Redundancies and Incremental Algorithm

In observing example 4.3, we notice that the two pairs of inequations which have not given relevant disequations, produced some trivially redundant inequations in \( S_{[y]} \).

Now, if a system of inequations is of full dimension, the
Input: $S, D$, where $S$ is a full dimensional inequation system, and $D$ is a set of inequations.
$x_0$ a variable to be eliminated.

Output: $S_{(x_0)}, D_{(x_0)}$ a system equivalent to $S, D$ on $V - \{x_0\}$.

begin
1. $D_{(x_0)}$ is empty.
2. Compute $S_{(x_0)} = \text{Proj}_{(x_0)}(S)$.
3. For each $\bar{d} \in D$,
   If $x_0$ does not occur in $\bar{d}$, $\text{PutInto}(\bar{d}, D_{(x_0)}, S_{(x_0)})$.
   Else, For each pair $(i_1, i_2)$ of inequations of $S$ with non-zero and opposite sign coefficients of $x_0$,
   3.1 $e = \text{Proj}_{(x_0)}(\bar{d}, \{i_1, i_2\})$,
   3.2 $\text{PutInto}(e, D_{(x_0)}, S_{(x_0)})$.
4. Return $S_{(x_0)}, D_{(x_0)}$.
end

$\text{PutInto}(e, D_{(x_0)}, S_{(x_0)})$.

begin
If $(\bar{e}$ is relevant in $S_{(x_0)}$, and $D_{(x_0)} \neq \bar{e})$
1. Put $\bar{e}$ into $D_{(x_0)}$,
2. Suppress every new redundant inequation from $D_{(x_0)}$.
end

\textbf{FIGURE 1.} Incremental algorithm associated with Fourier's elimination

trivial inequations cannot be implicit equalities, and are thus strongly redundant. The following lemma shows that if the inequation produced by a pair is strongly redundant, then this pair only gives irrelevant inequations.

\textbf{LEMMA 4.1.} Let $S$ be a full dimensional inequation system. Let $i_1$ and $i_2$ be two inequations of $S$ with non-zero and opposite sign coefficients of $x_0$. Assume that the resulting inequation by Fourier's elimination is strongly redundant in $S_{(x_0)}$. Then, for every inequation $\bar{d}$ the system $(d, \{i_1, i_2\})$ is inconsistent.

\textbf{Proof.} Let $a_1 x \leq b_1$ and $a_2 x \leq b_2$ be two inequations. By the definition of strong redundancy (subsection 2.5.2), every solution of $S$ cannot simultaneously satisfy the equations $a_1 x = b_1$ and $a_2 x = b_2$. The resulting inequation is thus irrelevant.

The problem of weak redundancies is more delicate. As in example 4.4 below, a loss of information can happen when the linear combinatory with positive coefficients involves only inequations of $S \cap S_{(x_0)}$.

\textbf{EXAMPLE 4.4.} Assume that $S = \{0 \leq x, 0 \leq y, -x - y \leq z \leq x + y, x - 1 \leq z\}$ and $D = \{z \neq 0\}$. $z$ is the variable to be eliminated. There are only two pairs of inequations of $S$. The first, $\{x - y \leq z, z \leq x + y\}$, gives the weak redundant inequation $0 \leq x + y$ of $S_{(x)} = \{0 \leq x, 0 \leq y\}$, and the inequation $\{x + y \leq 0\}$ for which the precise form is $\{x = 0, y = 0\}$. The second pair, $\{x - 1 \leq z, z \leq x + y\}$, gives the strongly redundant inequation $0 \leq y + 1$ of $S_{(x)}$, and the inequation $y \leq -1$ irredundant in $S_{(x)}$. If we do not take into account the inequation $0 \leq x + y$ there is a loss of information.

The following lemma constitutes an introduction to global variable elimination.

\textbf{LEMMA 4.2.} Let $S$ be a full dimensional inequation system. Let $i_1$ and $i_2$ be two inequations of $S$ with non-zero and opposite sign coefficients of $x_0$. Assume that the resulting inequation $i$ by Fourier's elimination is weakly redundant in $S_{(x_0)}$. Let $\bar{d}$ be a disjunction of $D$, and let $\bar{d}'$ be the disjunction which results from $i_1, i_2$ and $d$. Then, either (1) one of the inequations of $S_{(x_0)}$ which $i$ linearly depends\textsuperscript{ll} on is not in $S$, then $\bar{d}'$ is redundant in $D_{(x_0)}$, or (2) $i$ linearly depends only on inequations $(j_1, \ldots, j_s)$ of $S \cap S_{(x_0)}$. Then the disjunction $\text{Proj}_{(x_0)}(\text{EqAffSol}(d, \{j_1, \ldots, j_s\}, S))$ is equivalent to $\bar{d}'$.

\textbf{Proof.} (1) Let $j$ be one of the inequations of $S_{(x_0)}$ which $i$ linearly depends on. Assume that $j$ results from

\textsuperscript{ll}Here, linearly depends means that we only take into account the inequations of $S$ involved in the combinatory with non-zero positive coefficients.
the pair $j_1, j_2$ of inequations of $S$. Let $d_j$ be the disjunction which results from $j_1, j_2$ and $d$. Then, every solution $\nu'$ of $S_{\{x_2\}}$ which makes $i$ an equation, makes $j_1, j_2, i, j_1$ and $i_2$ five equations. Consequently, if $\nu'$ is a solution of $d'$, it is a solution of $d_j$. Then $d'$ is redundant in $S_{\{x_2\}}$.

(2) Now, assume that $i$ linearly depends only on inequations \{ $j_1, \ldots, j_n$ \} of $S \cap S_{\{x_0\}}$. Let us rewrite $i_1$ and $i_2$ as $x_0 \leq e_1 x'_1 + f_1$ and $x_0 \geq e_2 x'_2 + f_2$. Then, $\{ x_0 = e_1 x'_1 + f_1, x_0 = e_2 x'_2 + f_2 \}$, $S \models \{ j_1, \ldots, j_n \}$. $S$. And then, $\text{Proj}_{\{x_0\}}(\text{EqAffSol}(d, \{ j_1, \ldots, j_n \}, S))$ is equivalent to $d'$ in $S_{\{x_0\}}$.

Now, the problem is to detect each part of $S \cap S_{\{x_0\}}$ such that there is a pair of inequations of $S$ which the resulting inequation linearly depends on. We will see this problem in the more general case of the elimination of more than one variable.

5. GLOBAL VARIABLE ELIMINATION

This section establishes the results upon which the global elimination of variables rest. This elimination suppresses, in a single operation, the totality of undesirable variables. An algorithm follows from these results.

5.1. Basic Results

The incremental algorithm associated with Fourier's elimination that we have just presented, starts from inequations of $S$ for determining the inequations of $D_w$. With lemma 4.2, we have begun to take the problem in the other sense, by looking at $S_w$. In this section, the point of departure will always be a subset $I$ of $S_w$. Then, we will try to determine if the boundary part of $S$, consistent with $I^*$ and the equation system $d$ associated with a disjunction, allow for the elimination of all the variables of $W$ in $d$. If this is the case, it will give a new disjunction of $D_w$. Notice that $\text{EqAffSol}(I^*, S)$ is equivalent to a system $R^*$, where $R$ is a subset of $S$ containing at least all the inequations that produce $I$, thus all the inequations of $S$ producing an inequation implied by $I$.

Let $E$ and $F$ be two equation systems and let $S$ be an inequation system. We define the following equation systems:

$$\delta(E, S, F) = \text{Proj}_w(\text{EqAffSol}(E, S, F)),$$

and

$$\Delta(E, S, F) = \begin{cases} \delta(E, S, F) & \text{if } \text{EqAffSol}(E, S, F) = \text{EqAffSol}(E, S, \delta(E, S, F)) \\ \emptyset & \text{otherwise.} \end{cases}$$

Let $S$ be a constraint system. We will need an equation system denoted by $\text{Inconsistent}(S)$ (or $\text{Inconsistent}$ if there are no ambiguities), which is the conjunction of every equation which can be constructed using variables occurring in $S$. We will consider that $\text{Inconsistent}$ is the canonical form of all inconsistent systems. Thereby, if $E, S, F$ is inconsistent, then $\Delta(E, S, F) = \text{Inconsistent}$.

Let us remark that the following relation is always satisfied:

$$\text{EqAffSol}(E, S, F) \models \text{EqAffSol}(E, S, \delta(E, S, F)).$$

(5)

The following result will be very useful to us:

**Lemma 5.1.** Let $S$ be a full dimensional inequation system, and let $E$, $E'$ and $F$ be three equation systems. Then, $(E \models E') \iff (\Delta(E, S, F) \models \Delta(E', S, F)).$

**Proof.** If $\Delta(E', S, F)$ is empty, then it is obvious. Otherwise, $E \models E'$ implies $\text{EqAffSol}(E, S, F) \models \text{EqAffSol}(E', S, F)$. Then $\delta(E, S, F) \models \delta(E', S, F)$, and consequently $\text{EqAffSol}(E, S, \delta(E, S, F)) \models \text{EqAffSol}(E', S, \delta(E', S, F))$. Since $\Delta(E', S, F)$ non-empty implies $\text{EqAffSol}(E, S, \delta(E', S, F)) \models F$, we have $\text{EqAffSol}(E, S, \delta(E, S, F)) \models \text{EqAffSol}(E, S, F)$, then the equivalence by (5) and then the identity since these systems are in solved form. Consequently, $\Delta(E, S, F) \models \Delta(E', S, F)$. 

In the utilization that we made of $\Delta$, $F$ will represent the negation of a disjunction $\bar{d}$ of $D$, and $E$ will be an equation system $I^*$, where $I$ is a subset of $S_w$. The function $\Delta$ is intended to be seen if there is a loss of information. Let us consider the set of all the equation systems in solved form which can be constructed on the set of variables $V$. Let $G$ be the set of all these equation systems in solved form, to which is added the system $\text{Inconsistent}$. And let $S$ be an inequation system. Let us consider the application $\Phi$ defined by $G \xrightarrow{\Phi} G$

$$g \rightarrow \Phi(g) = \text{EqAffSol}(E, S, \text{Proj}_w(g)).$$

Then, $\Delta(E, S, F)$ is non-empty and only if $g = \text{EqAffSol}(E, S, F) = \text{fixed point of the application } \Phi$ (i.e. $\Phi(g) = g$). Moreover, if $\Delta(E, S, F)$ is not empty, then $\Delta(E, S, F) = \text{Proj}_w(g)$.

**Theorem 5.2.** (Global Elimination) Let $S$ be a full dimensional inequation system, and let $D$ be a set of disjunctions. Then, the two systems $(S_w, D_w)$ and $(S_w, \{ \Delta(I^*, S, d) : I \subseteq S_w, \ d \in D, \ \Delta(I^*, S, d) \neq \emptyset \})$ are equivalent.

**Proof.** According to theorem 5.2, it is only necessary to verify for $D = \{ \bar{d} \}$. Whereas $\Delta(I^*, S, d)$ non-empty implies $\text{EqAffSol}(I^*, S, d)$ is a fixed point of $\Phi$, we have $(I^*, S, \Delta(I^*, S, d)) \models d$. Moreover, $\delta(I^*, S, d) \models I^*$ and thus $\Delta(I^*, S, d) \models I^*$. Consequently, for each solution $v'$ of $S_w$, $\Delta(I^*, S, d)$, and for all $v''$ such that $(v', v'')$ is a solution of $S$, $(v', v'')$ is a solution of $d$. This proves that $(S_w, D_w) = \Delta(I^*, S, d)$.

Assume that $D_w$ only contains disjunctions relevant in $S_w$ and is irredundant. According to the proof of the incremental algorithm associated with Fourier's
elimination, and using a recurrent procedure, \( D_w \) can be wholly constructed from equation systems \((R^w, d)\), where \( R \) is a subset of \( S \). Moreover, \( R \) makes it possible to construct an inequation \( i \) such that \( S_w \models i \) (Fourier’s algorithm 2.5.1). According to Section 2.5.2, \( i \) is a non-zero positive linear combinatory of the inequations of \( R \). As a result, \( R^w \models i^w \), and \((i^w, S) \models R^w \). Moreover, \( S_w \models i \) means there is a subset \( I \) of \( S_w \) such that \( i \) is a non-zero positive linear combinatory of the inequations of \( I \) (according to the redundancy definition, it is an affine combinatory. If it were not linear, then \((i^w, S_w)\) and thus by lemma 4.1, \((i^w, S) \) would not be consistent). Hence, \( I^w \models i^w \), and then \((i^w, S) \models I^w \). As a result, \((R^w, S)\) and \((I^w, S)\) are equivalent, and according to lemma 5.1, \( \Delta(R^w, S, d) = \Delta(I^w, S, d) \). Since by construction of \( R \) in the proof of the incremental algorithm, \( (R^w, \text{Proj}_w(R^w, d)) \models (R^w, d) \), \( \Delta(R^w, S, d) \) is not empty and thus \( \Delta(I^w, S, d) \) is not empty.

The three following results allow for some simplifications with respect to the disequations to be retained. Theorem 5.3 makes it unnecessary to verify the relevance of the disequation \( \Delta(I^w, S, d) \), since it is detected at the time of its computation.

**Theorem 5.3. (Relevance of \( \Delta \))** Let \( S \) be a full dimensional inequation system and, let \( E \) and \( F \) be two equation systems. Then, \( E, S, F \) is consistent if and only if \( S_w, \Delta(E, S, F) \) is consistent.

Proof. If \( E, S, F \) is Inconsistent, then \( \Delta(E, S, F) \) is Inconsistent, and then \( S_w, \Delta(E, S, F) \) is Inconsistent. If \( E, S, F \) is consistent, then either \( \Delta(E, S, F) \) is empty, and then \( S_w \) is consistent as \( S \) is, or \( \Delta(E, S, F) \) is non-empty, and every solution \( (v', v'') \) of \( E, S, F \) is a solution of \( S \) and of \( \text{EqAff}_w(S, E, F) \), and then \( v' \) is a solution of \( \Delta(E, S, F) \) and of \( S_w \).

Now we will show that the disequations so obtained are precise. As a result, the new generalized linear constraint system will be easily maintained in canonical form, and the detection of the remaining redundant disequations will be easier.

**Theorem 5.4. (Precision)** Let \( S \) be a full dimensional inequation system, let \( \bar{d} \) be a disequation, and let \( I \subset S_w \), such that \( I^w, S, d \) is consistent. Let us assume that \( \Delta(I^w, S, d) \) is non-empty. Then \( \Delta(I^w, S, d) \) is a disequation precise in \( S_w \).

Proof. Since the system \( I^w, S, d \) is consistent, according to theorem 5.3, \( \Delta(I^w, S, d) \) is relevant. And since \( S_w, \Delta(I^w, S, d) \models S_w, \Delta(I^w, S, d), \) according to the definition of \( \Delta \), for all \( i \in S_w \),

\[
(S_w, \Delta(I^w, S, d) \models i^w) \implies (\Delta(I^w, S, d) \models i^w).
\]

According to the definition of the Inconsistent system, the set \( S_\Delta \) defined by \( \{\Delta(I^w, S, d) \mid I \subset S_w\} \) has a lattice structure for \( \models \). Let us assume that \( S \) and \( d \) are known. The mapping which associates \( \Delta(I^w, S, d) \) to each subset \( I \) of \( S_w \) is monotonic for the inclusion in \( S_w \) and for \( \models \) in \( S_\Delta \) (lemma 5.1). An acceptable minimal subset is a part \( I \) of \( S_w \) such that \( \Delta(I^w, S, d) \) is non-empty and consistent, and such that \( I \) does not contain any other part with this same property. When \( I \) is acceptable minimal, the system \( \Delta(I^w, S, d) \) is said to be acceptable minimal in \( S_\Delta \). The non-empty elements of \( S_\Delta \) which are not acceptable minimal, give irrelevant or redundant disequations. As a result:

**Theorem 5.5. (Optimal Global Elimination)** Let \( S \) be a full dimensional inequation system, and let \( D \) be a set of disequations. Then, \( S_w, D_w \) is equivalent to

\[
S_w, \{\Delta(I^w, S, d) \mid I \subset S_w, \bar{d} \in D, I \text{ minimal acceptable}\}.
\]

Proof. This is straightforward from the comment before this theorem, theorem 5.2 and lemma 2.2.

This theorem shows the great advantage of an inequation system \( S_w \) without redundancy, which is thus the smallest. Moreover, in practice, \( \Delta(I^w, S, d) \) quickly becomes inconsistent when \( I \) increases.

Remarks:

- If no variable of \( W \) occurs in \( d \), then \( \Delta(\emptyset, S, d) \) is not empty. It is the lone minimal element of \( S_\Delta \).
- Let \( (i, j) \) be a pair of inequations of \( S \) with non-zero opposite sign coefficients of \( x_0 \). Let \( k \) be the inequation of \( S_{\{x_0\}} \) produced by \( i \) and \( j \). Let us assume that \( x_0 \) occurs in \( d \). Then \( \Delta(k^w, S, d) \) is an acceptable minimal element.
- Let us assume that \( S_{\{x_0\}} \) is computed from \( S \) using an algorithm of the type described in [10] based on Fourier’s elimination. In the incremental algorithm it is possible to gather the computation of an inequation \( k \) of \( S_{\{x_0\}} \) and the computation of \( \Delta(k^w, S, d) \). Hence, the computation of numerous irrelevant or redundant disequations can be avoided. However, according to lemma 4.2, theorem 5.5 and the two previous remarks we will have to add the computation of \( \Delta(I^w, S, d) \) for all acceptable minimal parts \( I^w \) of \( S \cap S_{\{x_0\}} \).

**Example 5.1.** Return to Example 4.4. Let \( S = \{0 \leq x, 0 \leq y, -x - y \leq z \leq x + y, x - 1 \leq z\} \) and \( D = \{z \neq 0\} \). Then, \( S_{\{z\}} = \{0 \leq x, 0 \leq y\} \). With \( d = \{z = 0\} \), the systems \( \Delta(\emptyset, S, d) \), \( \Delta(\{x = 0\}, S, d) \) and \( \Delta(\{y = 0\}, S, d) \) are empty and, \( \Delta(\{0 = x, 0 = y\}, S, d) = \{x = 0, y = 0\} \). Hence, \( D_{\{z\}} = \{x = 0, y = 0\} \).

**Example 5.2.** Let \( S = \{0 \leq x, -x - y \leq z \leq x + y\} \) and \( D = \{z \neq 0\} \). Then, \( S_{\{z\}} = \{0 \leq x, 0 \leq x + y\} \). With \( d = \{z = 0\} \), the systems \( \Delta(\emptyset, S, d) \) and \( \Delta(\{x = 0\}, S, d) \) are empty, and, \( \Delta(\{0 = x + y\}, S, d) = \{x = 0, y = 0\} \). Then \( S_{\{z\}} \) is not acceptable minimal. Hence, \( D_{\{z\}} = \{0 \neq x + y\} \).
5.2. Global Variable Elimination Parallel Algorithm

According to previous results, we can independently determine for each disequation \( d \) and each part \( I \) of \( S_W \), whether \( \text{EqAffSol}(I^=, S, d) \) is consistent and is a fixed point of \( \Phi \). If it is, we obtain a disequation of \( D_W \) (theorem 5.2). Then it remains to sieve the disequations so obtained for eliminating redundant constraints. According to lemma 2.2, section 2.3, and theorem 5.4, there exists efficient sequential or parallel algorithms to achieve the sieving. The algorithm presented in Figure 2 rests on the results of theorems 5.2, 5.3 and 5.4, which allow for a high degree of parallelism. In the pseudocode of this algorithm, we use the construct `parallel do - parallel end`, to describe the parallel execution of a set of routines.

6. CONCLUSION

Using principally the studies of J.-L. Laszlo and K. MacAloon on the canonical form of a generalized linear constraint system [20], we have established that for the variable elimination operation, on the one hand, the equation and inequation (\( \leq \)) type constraints are independent of disequation type constraints (\( \neq \)) and, on the other hand, the disequations are independent from one another. Considering the elimination done for positive constraints, we have shown that variable elimination in the set of disequations is dependent on the system of final inequations. This elimination is done in an independent way for each pair formed of a disequation and of a part of the final inequation system. In order to take into account redundancies and the relevance of the disequations obtained, we have made the properties of minimality appear on the parts of the inequation systems in relation to each disequation. The numerous independencies thus revealed allow for a high degree of parallelism. This is largely exploited in the global variable elimination algorithm.

The complexity of global elimination depends largely on the size of the final inequation system. In some degree, this complexity is not as bad as one might have thought, mainly because in the general case, there are few hyperplanes among those which delimit a polyhedral set that pass by the same point. To a less important degree, this complexity depends on the size of the initial inequation system. Equally, it depends linearly on the number of disequations. In no case does it depend on the size of an intermediary system. This is a major advantage with regard to the incremental elimination and, more general methods as in Tarski [23] and Collins [2]. However, it would be desirable to precisely study the complexity of these algorithms and to make a larger experimental evaluation.

A study is now in progress, on the detection of redundant disequations. A mapping between subsets of initial inequations and disequations makes it possible to bring these redundancies to light using only comparisons. This fact is used in a sequential global elimination algorithm (paper forthcoming).

ACKNOWLEDGEMENTS

I would like to thank Alain Colmerauer and Pascal Van Hentenryck for interesting discussions and comments. I would also like to thank Pamela Morton for her help in the English translation of this paper. This research was supported by the ACCLAIM Esprit Research Project (PE7195).
REFERENCES