Incremental Closure for Systems of Two Variables Per Inequality

Jacob M. Howe[*,a], Andy King[*,b], Axel Simon[c]

[a]*Department of Computer Science, City, University of London, EC1V 0HB, UK*
[b]*School of Computing, University of Kent, Canterbury, CT2 7NF, UK*
[c]*Google, Mountain View, California CA94043, USA*

Abstract

Subclasses of linear inequalities where each inequality has at most two variables are popular in abstract interpretation and model checking, because they strike a balance between what can be described and what can be efficiently computed. This paper focuses on the TVPI class of inequalities, for which each coefficient of each two variable inequality is unrestricted. An implied TVPI inequality can be generated from a pair of TVPI inequalities by eliminating a given common variable (echoing resolution on clauses). This operation, called result, can be applied to derive TVPI inequalities which are entailed (implied) by a given TVPI system. The key operation on TVPI is calculating closure: satisfiability can be observed from a closed system and a closed system also simplifies the calculation of other operations. A closed system can be derived by repeatedly applying the result operator. The process of adding a single TVPI inequality to an already closed input TVPI system and then finding the closure of this augmented system is called incremental closure. This too can be calculated by the repeated application of the result operator. This paper studies the calculus defined by result, the structure of result derivations, and how derivations can be combined and controlled. A series of lemmata on derivations are presented that, collectively, provide a pathway for synthesising an algorithm for incremental closure. The complexity of the incremental closure algorithm is analysed and found to be $O((n^2+m^2)\lg(m))$, where n is the number of variables and m the number of inequalities of the input TVPI system.

Key words: abstract interpretation, weakly relational abstract domains, two variables per inequality (TVPI) abstract domain

*Corresponding authors

Email addresses: j.m.howe@city.ac.uk (Jacob M. Howe), a.m.king@kent.ac.uk (Andy King), axelsimon@google.com (Axel Simon)

1. Introduction

The study of how to eliminate a variable from a system of linear inequalities dates back to at least Fourier's 1824 work [15], that demonstrates that the satisfiability of a system of inequalities can be established by successively eliminating variables until exactly one remains. This satisfiability argument is also used by Nelson [34], who considers it in the special case that each inequality has at most two variables, a satisfiability problem further considered by [7, 21]. Further restrictions to the class of linear inequalities have also been considered, with Harvey [19] considering the integer satisfiability of systems of inequalities with two variables per inequality where the coefficients are unit (that is, 0, 1 or -1).

These unit two variables per inequality systems are also considered in the context of program analysis, where they form the underlying system of inequalities of the abstract domain of Octagons [33]. A number of classes of inequalities have been used as the foundation of numeric abstract domains, including the class of inequalities with at most two variables per inequality [40]. Note that Octagons are a strict subclass of these inequalities.

This paper is concerned with systems of inequalities with two variables per inequality (TVPI), and the addition of inequalities to the system. In particular adding inequalities incrementally, that is, one at a time. The work is focused on a normal form for systems of inequalities with two variables per inequality, and how the normal form can be reached using a calculus of operations which combines inequalities using a resolution-like operator, similar to the Fourier-Motzkin elimination step. Key results in this work will show that the calculus need only be applied in a restricted way in order to derive the inequalities required for the normal form.

A system (set) in this normal form is called closed. Given a set of two or fewer variables, the syntactic projection of a system onto these variables is the subsystem (subset of the given system) consisting solely of those inequalities whose variables are drawn entirely from the set. A system is closed when every inequality with two or fewer variables that is entailed (implied) by the system is also entailed by the syntactic projection of the system onto the variables of the inequality being considered. For example, consider the system $\{x - y \leq 0, y - z \leq 1\}$. This is not closed since the $x - z \leq 2$ is entailed by the system, but not by the subsystem of inequalities that contain only the variables x, z (which is empty in this case). However, $\{x - y \leq 0, y - z \leq 1, x - z \leq 1\}$ is closed, because the syntactic projection onto x, z is the subsystem $\{x - z \leq 1\}$ which entails $x - z \leq 2$.

To obtain a closed system, implied inequalities in each two variable projection need to be made explicit and inserted. To be closed, all the irredundant inequalities in each one or two variable projection must be enumerated. Implied inequalities can be found by eliminating a single common variable from a pair of inequalities drawn from the system, or calculating a resultant of the two inequalities, using the terminology of Nelson [34]. The resultant calculation is here formalised (see Section 3) as the result operator. Control of the generation of these inequalities is not straightforward, as the following example illustrates. Consider the system of inequalities $I_0 = \{x - y \leq 0, 2x - z \leq 0\}$, which is closed. Now consider augmenting the system with $c_0 = -x + z \leq 0$, and note that $I_0' = I_0 \cup \{c_0\}$ is not closed, since $z - y \leq 0$ is entailed by I_0', but not included in it, nor entailed by the syntactic projection of I_0' onto $\{z, y\}$ (which is empty). Using the result operation, implied inequalities can be derived. For example, the result operation can be applied four times to give the following derivation (the notation will be formally introduced in Section 3):

$$\dfrac{\dfrac{\dfrac{\dfrac{\overset{c_0}{-x + z \leq 0} \quad x - y \leq 0}{z - y \leq 0}\,x \quad 2x - z \leq 0}{2x - y \leq 0}\,z \quad -x + z \leq 0}{2z - y \leq 0}\,x \quad 2x - z \leq 0}{4x - y \leq 0}\,z$$

The system is incrementally augmented with derived inequalities as follows.

$$I_1 = I_0' \cup \{z - y \leq 0\} \qquad I_2 = I_1 \cup \{2x - y \leq 0\}$$
$$I_3 = I_2 \cup \{2z - y \leq 0\} \qquad I_4 = I_3 \cup \{4x - y \leq 0\}$$

Observe that the syntactic projection of I_0' onto $\{x, y\}$ is $x - y \leq 0$, which does not entail $2x - y \leq 0$. Thus I_0' is not closed. To address this, the syntactic projection of I_2 onto $\{x, y\}$ is tightened. Again the syntactic projection of I_2 onto $\{x, y\}$ does not entail $4x - y \leq 0$, hence I_2 is not closed. Therefore, again the syntactic projection of I_4 onto $\{x, y\}$ is tightened. In this way, the syntactic projection onto x, y can be tightened ad infinitum, hence a closed system is never achieved. However, observe that

$$\dfrac{2x - z \leq 0 \quad -x + z \leq 0}{x \leq 0}\,z$$

If this were added to I_0' then $2x - y \leq 0$ would be entailed by the syntactic projection onto $\{x, y\}$, $\{x - y \leq 0, x \leq 0\}$, and its addition would not tighten the system. In fact, the system $I_0' \cup \{z - y \leq 0, x \leq 0, z \leq 0\}$ is closed. Note that the addition of new inequalities might well make some inequalities within the system redundant.

This poses some questions as to how to derive a closed system. Which inequalities should be combined using the result operator, and in what order should this be done, so that a terminating procedure giving a closed system is realised? In addition, are all the new inequalities necessary for a closed system or its derivation? Can unnecessary inequalities be removed?

The work in this paper answers these questions by reasoning about the result calculus. The answers then inform the construction of a polynomial algorithm for incrementally closing a system of TVPI inequalities. This results in an algorithm whose construction is principled, and whose results are justified rather than merely aligning with a closed system for inexplicable reasons; the technical challenge is not formulating the algorithm, but understanding and demonstrating that it is correct.

In studying this problem, this paper makes the following contributions:

- formalisation of the result calculus and notions including syntactic projection, closure, filtering (removal of extraneous inequalities) and completion (a combination of closure and filtering);
- the result calculus is treated as an object of study. A series of results are presented on the rewriting of derivations of inequalities into derivations where structural properties hold, demonstrating that only a certain class of derivations need to be considered; these results stem from the desire to make progress [41] on TVPI by advancing reasoning on TVPI systems;
- an algorithm is presented for inserting a new two (or fewer) variable inequality into a non-redundant planar system of inequalities, which results in an updated system with no redundancy. This is lifted to systems of TVPI inequalities and is used to define incremental completion;
- a complexity analysis of operations relating to incremental completion is given. Incremental closure realised as incremental completion resides in $O((n^2 + m^2)\lg(m))$ time where n is the number of variables and m is the number of inequalities in the system being augmented;
- the restriction of these results to the Octagon and Logahedra subclasses of two variables per inequality systems is also considered. The complexity of operations (in particular closure) on Octagons over rationals of this specialisation matches that given in previous work [33] despite not relying on the encoded matrix representation given there. This suggests that alternative representations of Octagons, including compact representations, are conceivable;
- the algorithm is implemented and the growth of incrementally closed systems is evaluated.

The rest of the paper is structured as follows: Section 2 places this paper in the context of related work and Section 3 defines key concepts. Section 4 gives the proof of the correctness of incremental closure (using completion). This is followed by Section 5 that gives a discussion of the construction of the algorithm together with analysis of its complexity, and Section 6 that experimentally evaluates the algorithm. Section 7 concludes.

2. Context and Background

This section positions the current work in the context of previous work on restricted classes of linear inequalities, how they arise and the problems addressed. This work is motivated by applications in abstract interpretation (Section 2.4) and model checking (Section 2.3).

2.1. Satisfiability

Given a system S of m TVPI inequalities over a set of n variables X, a chosen variable $x \in X$ and an arbitrary constant $c \in \mathbb{Q}$, it is possible to decide whether the augmented system $S \cup \{x = c\}$ is satisfiable over $\mathbb{Q}$ in $O(mn)$ time [1]. Strongly polynomial decision procedures for feasibility have been proposed for TVPI systems [7], including one, founded on [1], which resides in $O(mn^2 \lg(m))$ [21]. The satisfiability problem for integer TVPI is NP-complete [26].

2.2. Linear Programming

Linear programming over TVPI has attracted interest because the dual problem is a generalised minimum-cost flow problem (flow with losses and gains) [20]. A linear program over TVPI can be solved in $O(m^3 n^2 \lg(m) \lg(B))$ time [44] where B is an upper bound on the number of bits required to store the absolute value of the largest (rational) coefficient of the TVPI system. The B term implies the algorithm is not strongly polynomial. Integer TVPI linear programming is NP-complete since satisfiability for integer TVPI is NP-complete [26] (it also follows by encoding the vertex cover problem [17, Section 3.1.3]).

2.3. Model Checking

Pratt [35] observed that solving systems of TVPI inequalities of the restricted form $x_i - x_j \leq c$, where $x_i, x_j \in X$, can be solved in polynomial time. These inequalities, called difference constraints, have gained traction in model checking [11, 28], where they are deployed to bound the time difference [31] between event i and event j. If $X = \{x_0, \ldots, x_{n-1}\}$ then an inequality $x_i - x_j \leq c$ can be represented by storing c at the i, j entry of an $n \times n$ matrix, called a Difference Bound Matrix (DBM). The absence of an upper bound on $x_i - x_j$ is indicated by an entry of ∞. A DBM thus gives a natural representation for differences, and an all-pairs shortest path algorithm [14, 43] can determine satisfiability in $O(n^3)$, whereas a single source shortest path algorithm [3] can determine satisfiability in $O(nm)$ time.

2.4. Abstract Interpretation

The class of TVPI inequalities where the coefficients are either -1, 0 or 1 is called Octagons [33]. DBMs can be adapted to represent Octagons [33] and an all-pairs shortest path algorithm used to determine satisfiability [33], including over integers [2], in $O(n^3)$ time (based on the Floyd-Warshall all pairs shortest path algorithm), or $O(n^2 \lg n + mn)$ (based on Johnson's algorithm [24], exploiting a sparse representation). A single source algorithm has also been applied to this satisfiability problem [27] yielding an $O(nm)$ algorithm, work that has been extended to check satisfiability incrementally in $O(n \lg(n) + m)$ [38].

Two important operations in abstract interpretation are join (least upper bound) and projection (forget). These operators are deployed in concert in abstract interpretation [8] to compute invariants that hold on every path through a program, even when there are infinitely many. For Octagons, join computes the least Octagon which includes two given Octagons; it is used to summarise the properties that hold on different branches of a program. Projection eliminates a variable from an Octagon; it is used to discard information pertaining to a variable of a program which goes out of scope. Closed systems of inequalities provide a way to straightforwardly compute the join and projection operators. The join of two closed systems can be computed pairwise, by considering each two variable pair, and calculating the convex union of the two sets of inequalities found by projecting each of the two input systems onto this pair. Similarly, join and projection for closed TVPI systems can be calculated pairwise for closed systems.

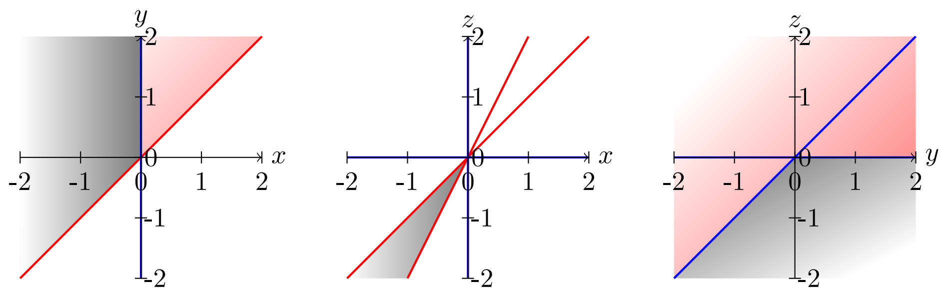
To illustrate, consider again $I_0' = \{x-y \leq 0, 2x-z \leq 0, -x+z \leq 0\}$ (coloured red in the first row of Figure 1) and recall that $I_0'' = I_0' \cup \{z-y \leq 0, x \leq 0, z \leq 0\}$ is closed (coloured grey in the first row of Figure 1). Now consider in addition, $J_0' = \{x+y \leq 0, 2x-y \leq 0, z \leq -1, y+z \leq 0, z-y \leq -1\}$ (coloured red in the second row of Figure 1) and observe that $J_0'' = J_0' \cup \{x \leq 0\}$ is closed (coloured grey in the second row of Figure 1). Observe that the third row of Figure 1 describes the TVPI system that encloses those of I_0'' and J_0''. Moreover, this system can be derived pairwise by computing the planar join (convex union) of each two variable projection. That is, each projection on the final row of Figure 1 is the convex union of the two projections on the preceeding rows. Notice that without computing closure the pairwise convex union calculations will not give the join of the two input systems; for example, without closure, the convex union of the two $\{y, z\}$ spaces is the whole $\{y, z\}$ space.

Invariants can be inferred using general polyhedra [9], but there is much interest in identifying subclasses of linear inequalities which balance the expressiveness of the invariants against computational tractability. As well as Octagons and TVPI [40], other examples of these weakly relation domains include: differences (or zones) [32], pentagons [30], zonotopes [18], Logahedra [22], octahedra [6], subpolyhedra [29], and linear template constraints [37].

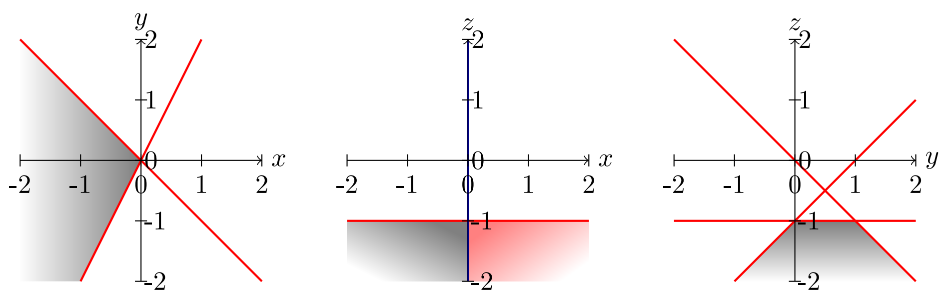
A frequently occurring use-case in abstract interpretation is adding a single constraint to system of constraints and then checking satisfiability, a problem which is addressed herein for TVPI. This use-case arises when a TVPI description for the program state at one line is adjusted to obtain a TVPI description for the next, possibly adding a series of TVPI inequalities to a given TVPI system [33]. Incremental closure also arises when modelling machine arithmetic with polyhedra [39] where integer wrapping is simulated by repeatedly partitioning a space into two (by adding a single unary inequality), closing and then performing translation. Integer wrapping is applied whenever a guard is encountered and since each encounter invokes incremental closure repeatedly, the faithful modelling of machine arithmetic is predicated on the existence of an efficient incremental closure algorithm.

2.5. Polyhedral Compilation

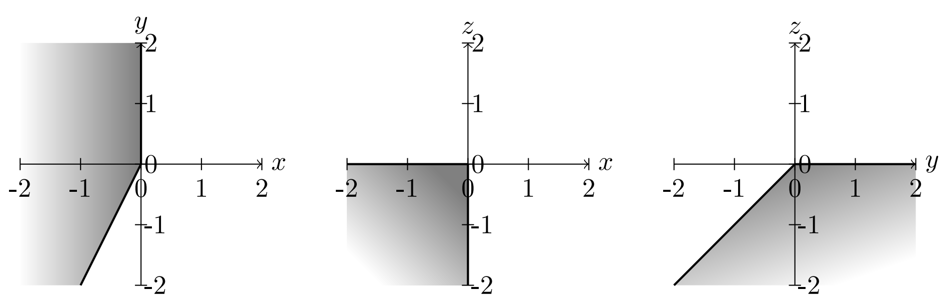
Polyhedral compilation [36] offers a formal way to understand loop optimisation for bounded iteration, where array indices are restricted to affine functions on the loop variables (possibly augmented with symbolic constants). A dependency graph is created with statements for nodes and edges which indicate a



I'_0 (red) tightened to its closure I''_0 (grey)



J'_0 (red) tightened to its closure J''_0 (grey)



Least upper bound of I''_0 and J''_0

Figure 1: Closure and least upper bound

dependence between one statement that writes to an array element and another that reads from an array. Each edge is decorated with a polyhedron, which indicates how the array indices differ or overlap when the read of one statement is reached after the write of another statement [12]. A transformation phase [10] then follows, for example to minimise the overall latency [13], which amounts to finding a partial order (a schedule) whose edges include those of the dependency graph, which can be realised with code that satisfies given architectural constraints. Although it appears intractable to compute a best schedule [25], good schedules can be derived efficiently by approximating the polyhedra with TVPI inequalities, so as to provide a way of scaling polyhedral compilation [42].

3. Preliminaries

The study commences with the class of two variables per inequality constraints [34, 40] that is defined over a given set of variables X:

Definition 1. $\mathsf{TVPI}_X = \{ax + by \leq e \mid x, y \in X \wedge a, b, e \in \mathbb{Q}\}$

Suppose $X = \{u, x, y, z\}$. Note that TVPI_X includes unary constraints, such as $2x \leq 3$, by setting, say, $b = 0$. It also contains constant constraints such as $0 \leq 0$ and $0 \leq -1$ which abbreviate to true and false respectively. This class of inequalities possesses the property that TVPI_X is closed under variable elimination. That is, eliminating variables from some $S \subseteq \mathsf{TVPI}_X$ results in a system of inequalities $S' \subseteq \mathsf{TVPI}_X$. For example, the variable y can be eliminated from the system $S = \{x-2y \leq 5, 3y+z \leq 7, 5y-u \leq 0\}$ by combining pairs of inequality with opposing signs for y. This yields the projection $S' = \{3x+2z \leq 29, -2u+5x \leq 25\}$, which is indeed a system of TVPI_X inequalities.

Observe that $c \in \mathsf{TVPI}_X$ can be represented in several ways. For example the inequality $2x+4y \leq 2$ might also be represented by $x+2y \leq 1$. This leads to the concept of semantic equivalence. Denote by $c \equiv c'$ that one inequality is merely a multiple of the other. This will be used when naming inequalities, for example, $c \equiv x-2y \leq 3$ or $c_1 \equiv a_1x+b_1y \leq e_1$. More generally, equivalence is formulated in terms of the entailment relation. Given two systems $S, S' \subseteq \mathsf{TVPI}_X$, S entails S', denoted $S \models S'$, if any assignment that satisfies S also satisfies S' (where an assignment is assumed to be a mapping from X to $\mathbb{Q}$). For instance, $S \models S'$ where $S = \{x - 2y \leq 7, y \leq -2\}$ and $S' = \{x \leq 4\}$ since every assignment to x and y that satisfies S also satisfies S'. The converse is not true since the assignment $\{x \mapsto 4, y \mapsto 3\}$ satisfies S' but not S. Equivalence is defined as $S \equiv S'$ iff $S \models S'$ and $S' \models S$. The entailment of a single inequality c by system S will be denoted $S \models c$. For notational convenience this paper implicitly assumes that an inequality is a representative of an equivalence class and might be multiplied through by any positive constant.

The set of variables that occur in c is denoted $\mathsf{vars}(c)$ and is defined:

Definition 2.

$$\mathsf{vars}(ax+by \le e) = \begin{cases} \emptyset & \text{if } \ x = y \wedge a = -b \\ \emptyset & \text{else if } \ a = b = 0 \\ \{y\} & \text{else if } \ a = 0 \\ \{x\} & \text{else if } \ b = 0 \\ \{x, y\} & \text{otherwise} \end{cases}$$

The set $\mathsf{vars}(c)$ contains either 2, 1 or 0 variables defining whether c is binary, unary or constant. This is then lifted to sets of inequalities, so as $\mathsf{vars}(S \cup \{c\}) = \mathsf{vars}(S) \cup \mathsf{vars}(c)$. For example, in the example at the start of this section, $\mathsf{vars}(S) = X$.

Other classes of inequalities with two variables per inequality can be defined and are of interest. In particular, Octagons [33] and their generalisation, Logahedra [22]. These classes of inequalities are given below. Octagons have variations where the constant term is drawn from $\mathbb{Z}$; in addition, bounded Logahedra are defined where the set of coefficients is generated with a maximum exponent (so that Octagons are 0-Logahedra).

Definition 3. $\mathsf{Oct}_X = \{ax + by \le e \mid x, y \in X \wedge a, b \in \{-1, 0, 1\} \wedge e \in \mathbb{Q}\}$

Definition 4. $\mathsf{Log}_X = \{ax + by \le e \mid x, y \in X \wedge a, b \in \{-2^n, 0, 2^n \,|\, n \in \mathbb{Z}\} \wedge e \in \mathbb{Q}\}$

Observe that $\mathsf{Oct}_X \subseteq \mathsf{Log}_X \subseteq \mathsf{TVPI}_X$.

As noted above, when a variable is eliminated from a subset of TVPI_X (or Log_X or Oct_X), the resulting inequalities are still a subset of TVPI_X (or Log_X or Oct_X). In particular, the action of combining a pair of inequalities in TVPI_X with a common variable to eliminate this variable results in an inequality that is again in TVPI_X. The action of combining inequalities, or computing resultants to use the terminology of Nelson [34], is formalised below:

Definition 5. If $c_1 \equiv a_1x + b_1y \le e_1$, $c_2 \equiv a_2x + b_2z \le e_2$ and $a_1a_2 < 0$ then

$$c = \mathsf{result}(c_1, c_2, x) = |a_2|b_1y + |a_1|b_2z \le |a_2|e_1 + |a_1|e_2$$

otherwise $\mathsf{result}(c_1, c_2, x) = \bot$. This resultant will also be denoted:

$$\frac{c_1 \quad c_2}{c} \; x$$

The term in y is said to *derive* from c_1 and that in z to derive from c_2.

Note that a single pair of inequalities may possess two resultants, as is illustrated by the pair $c_1 \equiv x + y \le 1$ and $c_2 \equiv -2x - 3y \le 1$ for which $\mathsf{result}(c_1, c_2, x) = -y \le 3$ and $\mathsf{result}(c_1, c_2, y) = x \le 4$. Hence it is necessary to stipulate which variable is being eliminated. However, in contexts when the eliminated variable is not named $\mathsf{result}(c_1, c_2)$ will be used without explicit stipulation of the variable. Pairs of inequalities for which the given variable cannot be eliminated are indicated by $\bot$, which can be ignored from that point on.

The result operation defines a resultant calculus, and it is derivations of inequalities in this calculus that are the object of study in much of this paper.

Definition 6. A series of applications of result form a *derivation tree*. If c is a leaf of the derivation tree (that is, not the result of an application of result) then $\mathsf{size}(c) = 0$. If $\mathsf{size}(c_1) = n_1$ and the $\mathsf{size}(c_2) = n_2$, and $c = \mathsf{result}(c_1, c_2, x)$ then $\mathsf{size}(c) = n_1 + n_2 + 1$.

Note the slight abuse of notation. The size is a property of a derivation, not an inequality, hence $\mathsf{size}(c)$ will have differing values for different derivations of c.

Example 1. Let $c_0 \equiv x + y \leq 1$, $c_1 \equiv -2x + u \leq 2$, $c_2 \equiv -4y - x \leq 1$ and $c_3 \equiv -y + z \leq 1$. Consider the following derivation tree of $c \equiv 2u + z \leq 11$:

$$\dfrac{\dfrac{\dfrac{\overset{c_0}{x+y\leq 1} \qquad \overset{c_1}{-2x+u\leq 2}}{2y+u\leq 4}\,x \qquad \overset{c_2}{-4y-x\leq 1}}{2u-x\leq 9}\,y \qquad \dfrac{\overset{c_0}{x+y\leq 1} \qquad \overset{c_3}{-y+z\leq 1}}{x+z\leq 2}\,y}{2u+z\leq 11}\,x$$

In this derivation $\mathsf{size}(c) = 4$.

The resultant operator lifts to sets of inequalities by:

Definition 7. If $C_1, C_2 \subseteq \mathsf{TVPI}_X$ then

$$\mathsf{result}(C_1, C_2) = \left\{ c \;\middle|\; \begin{array}{lll} c_1 \in C_1 \wedge c_2 \in C_2 & \wedge & x \in \mathsf{vars}(c_1) \cap \mathsf{vars}(c_2) \wedge \\ c = \mathsf{result}(c_1, c_2, x) & \wedge & c \neq \bot \end{array} \right\}$$

The following abbreviations are used: $\mathsf{result}(c, C) = \mathsf{result}(\{c\}, C)$ and $\mathsf{result}(C, c) = \mathsf{result}(C, \{c\})$, where $c \in \mathsf{TVPI}_X$ and $C \subseteq \mathsf{TVPI}_X$.

Another fundamental operator is syntactic projection, defined below.

Definition 8. The *syntactic projection*, denoted π_Y for some $Y \subseteq X$, of system of inequalities $S \subseteq \mathsf{TVPI}_X$ is defined as $\pi_Y(S) = \{c \in S \mid \mathsf{vars}(c) \subseteq Y\}$.

That is, the syntactic projection onto Y retains all inequalities whose variables are all in Y and discards all others. In a non-closed system syntactic projection will possibly lose information; syntactic projection is not semantic projection.

Consider the definition of a closed system:

Definition 9. A system $C \subseteq \mathsf{TVPI}_X$ is closed if the following predicate holds:

$$\mathsf{closed}(C) \iff \forall c \in \mathsf{TVPI}_X \,.\, (C \models c \Rightarrow \pi_{\mathsf{vars}(c)}(C) \models c)$$

A closed system is defined so that syntactic and semantic projection coincide. No further implied inequalities can be added to the system that will tighten any two variable projection.

The following example illustrates closed systems.

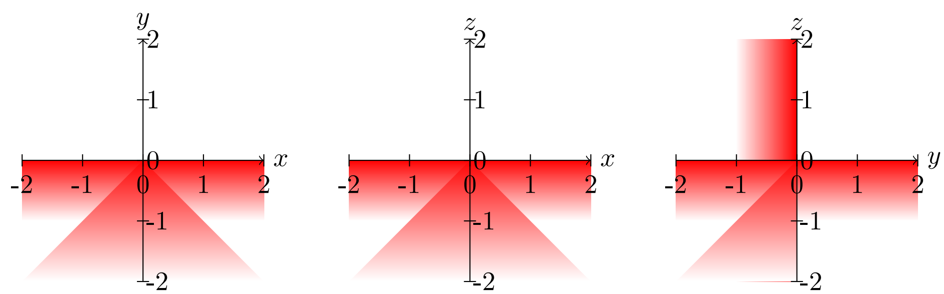


Figure 2: $\pi_{\{x,y\}}(C')$, $\pi_{\{x,z\}}(C')$ and $\pi_{\{y,z\}}(C')$

Example 2. Suppose $C = \{x + y \leq 0, -x + y \leq 0, -y + z \leq 0, 2y + x \leq 2\}$. Put $C' = C \cup \{x + z \leq 0, -x + z \leq 0, y \leq 0, z \leq 0\}$. Then C' is closed. Note that $y \leq 0$ is redundant in $\pi_{\{x,y\}}(C') = \{x + y \leq 0, -x + y \leq 0, y \leq 0\}$ but is irredundant in $\pi_{\{y,z\}}(C') = \{-y+z \leq 0, y \leq 0, z \leq 0\}$ as illustrated in Figure 2. Further note that $2y+x \leq 2$ is redundant in each two variable projection; closed systems can contain truly redundant inequalities.

The second example illustrates closed systems, noting that unary inequalities need to be considered.

Example 3. Suppose $C = \{x+y \leq 0, -x+y \leq 0\}$. Then C is not closed since $C \models y \leq 0$ but $\pi_{\{y\}}(C) = \emptyset \not\models y \leq 0$. However, $C \cup \{y \leq 0\}$ is closed.

The third example illustrates that consistency of the system needs to be considered.

Example 4. Suppose $C = \{x \leq -1, -x \leq -1\}$. Then C is not closed because $C \models \mathsf{false}$ and $\pi_{\emptyset}(C) = \emptyset \not\models \mathsf{false}$. Put $C' = C \cup \{\mathsf{false}\}$. For any other constraint $c \in \mathsf{TVPI}_X$ if $C' \models c$ it follows that $\pi_{\mathsf{vars}(c)}(C') \models c$ hence C' is closed. In particular $C' \models \mathsf{true}$ and $\pi_{\emptyset}(C') = \{\mathsf{false}\} \models \mathsf{true}$.

A system C can be augmented with inequalities to get C' so that $\mathsf{closed}(C')$, that is, the system can be *closed* in a process called *closure*.

If $Y = \{x, y\}$ then the syntactic projection $\pi_Y(S)$ yields a planar system. A planar system over Y can be filtered to remove any redundant inequalities. Similarly, the syntactic projection onto a single variable can be filtered (and this is trivial for projection onto no variables). These operators can be constructed for planar (and zero and one dimensional) polyhedra and lifted to systems of inequalities by taking the union of each filtered planar (and zero and one dimensional) projection. This is formalised in Definition 10. Therefore, only inequalities that are redundant in all one and two variable projections are removed.

The next definition considers how a two (or one or zero) dimensional projection can be filtered. This operation can then be lifted to systems of inequalities.

Definition 10. The mapping $\mathsf{filter} : \wp(\mathsf{TVPI}_X) \to \wp(\mathsf{TVPI}_X)$ is defined:

$$\mathsf{filter}(C) = \cup\{\mathsf{filter}_Y(\pi_Y(C))|Y \subseteq X \wedge |Y| \leq 2\}$$

where:

- $\mathsf{filter}_Y(C) \subseteq C$
- $\mathsf{filter}_Y(C) \equiv \pi_Y(C)$
- for every $C' \subset \mathsf{filter}_Y(C)$, $C' \not\equiv C$.

That no information is lost by filtering is formalised in the following lemma.

Lemma 1. If $X' \subseteq \mathsf{vars}(C)$ and $\pi_{X'}(C) \models c$ then $\pi_{X'}(\mathsf{filter}(C)) \models c$.

Proof . Since $\pi_{X'}(C) = \cup\{\pi_Y(C)|Y \subseteq X' \wedge |Y| \leq 2\}$ and $\pi_Y(C) \equiv \mathsf{filter}_Y(\pi_Y(C))$ for each Y then $\pi_{X'}(C) \equiv \cup\{\mathsf{filter}_Y(\pi_Y(C))|Y \subseteq X' \wedge |Y| \leq 2\}$. Hence $\pi_{X'}(\mathsf{filter}(C)) \models c$. □

Example 5. If $S = \{x + y \leq 0, -x + y \leq 0, y \leq 0\}$ then $\mathsf{filter}(S) = S$.

Example 6. If $S = \{x \leq -1, -x \leq -1, \mathsf{false}\}$ then $\mathsf{filter}(S) = \{\mathsf{false}\}$.

The following lemma demonstrates that a closed system remains closed after filtering.

Lemma 2. If $S \subseteq \mathsf{TVPI}_X$ and $\mathsf{closed}(S)$ then $\mathsf{closed}(\mathsf{filter}(S))$.

Proof . Let $S \subseteq \mathsf{TVPI}_X$ and suppose $\mathsf{closed}(S)$ holds. Let $c \in \mathsf{TVPI}_X$ such that $S \models c$. Then $\pi_{\mathsf{vars}(c)}(S) \models c$. Therefore by Lemma 1, $\pi_{\mathsf{vars}(c)}(\mathsf{filter}(S)) \models c$. □

4. Completion

A closed system of inequalities $I \subseteq \mathsf{TVPI}_X$ [40] can be found by augmenting I with inequalities $\mathsf{result}(I, I)$ until an I' is obtained such that no further (non-redundant) inequalities can be added to $\pi_Y(I)$ for any $|Y| \leq 2$. In this paper, the process of finding a closed system, including an interleaved filtering step to remove unnecessary inequalities, is called *completion*. Nelson [34] used closure as a way of deciding whether a given $I \subseteq \mathsf{TVPI}_X$ is satisfiable over $\mathbb{Q}$ or $\mathbb{R}$ and this provides the starting point for this section.

4.1. Full completion

The following result is Lemma 1b from Nelson [34].

Lemma 3. Suppose $C \subseteq \mathsf{TVPI}_X$, $c \in \mathsf{TVPI}_X$ and that $C \cup \{\neg c\}$ is unsatisfiable, where $C' \cup \{\neg c\}$ is satisfiable for all $C' \subset C$. Then, there exists $X' \subseteq X$ such that $|X'| \leq \lfloor |X|/2 \rfloor + 1$ and $\pi_{X'}(C \cup \mathsf{result}(C, C)) \cup \{\neg c\}$ is unsatisfiable.

Note that for $c \in \mathsf{TVPI}$ the constraint $\neg c$ is a strict two variable inequality, hence $\neg c \notin \mathsf{TVPI}$, given the definition of TVPI used in this paper. However, the results in [34] allow strict inequalities. Hence the following is a corollary of Lemma 3, expressing the result in terms of entailment.

Corollary 1. Suppose $C \subseteq \mathsf{TVPI}_X$, $c \in \mathsf{TVPI}_X$ and that $C \models c$. Then, there exists $X' \subseteq X$ such that $|X'| \leq \lfloor |X|/2 \rfloor + 1$ and $\pi_{X'}(C \cup \mathsf{result}(C, C)) \models c$.

The following defines the congruence relation $\cong$ for TVPI systems. Two systems are congruent when they agree when syntactically projected onto each projection of two or fewer variables.

Definition 11. $I \cong I'$ iff for all $Y \subseteq X$ such that $|Y| \leq 2$, $\pi_Y(I) \equiv \pi_Y(I')$.

The following lemma states that repeatedly applying result and filter leads to stability.

Lemma 4. Let $I \subseteq \mathsf{TVPI}_X$. Put $I_0 = \mathsf{filter}(I)$ and $I_{i+1} = \mathsf{filter}(I_i \cup \mathsf{result}(I_i, I_i))$. Then $I_k \cong I_{k+1}$ where $k = \lceil \lg_2(|X|) \rceil$.

Proof . Note that $I_0 \equiv ... \equiv I_k \equiv I_{k+1}$.

1. For $Y \subseteq X$, $|Y| \leq 2$ consider $c \in \pi_Y(I_{k+1})$. Observe that $I_0 \equiv I_{k+1}$, hence $I_0 \models c$. Put $X_0 = X$. By Corollary 1 it follows that there exists $X_{i+1} \subseteq X_i$ such that $|X_{i+1}| \leq \lfloor |X_i|/2 \rfloor + 1$ and $\pi_{X_{i+1}}(I_i \cup \mathsf{result}(I_i, I_i)) \models c$, hence by Lemma 1 $\pi_{X_{i+1}}(I_{i+1}) \models c$. In particular, $\pi_Y(I_k) \models c$, since $\mathsf{vars}(c) \subseteq Y$.
2. For $Y \subseteq X$, $|Y| \leq 2$ consider $c \in \pi_Y(I_k)$. Since $I_{k+1} \models c$ it follows that $\pi_Y(I_{k+1}) \models c$

Thus $\pi_Y(I_k) \equiv \pi_Y(I_{k+1})$, hence $I_k \cong I_{k+1}$. □

The definition of closed allows systems to contain inequalities that are redundant in the sense that they can be removed from the system whilst every one or two variable syntactic projection will still define the same space. These are the inequalities that are removed by filter. Computationally these inequalities incur a performance hit for operations on TVPI_X systems. This motivates a variation on closed systems, here called $\mathsf{complete}$. This is specified as a function on TVPI_X systems below. A system where $\mathsf{complete}(I) = I$ will be referred to as *complete*, and the process of computing a complete system will be referred to as *completion*.

Definition 12. Let $I \subseteq \mathsf{TVPI}_X$. Put $I_0 = \mathsf{filter}(I)$ and $I_{i+1} = \mathsf{filter}(I_i \cup \mathsf{result}(I_i, I_i))$. Then $\mathsf{complete} : \wp(\mathsf{TVPI}_X) \to \wp(\mathsf{TVPI}_X)$ is defined

$$\mathsf{complete}(I) = I_n \text{ where } I_{n+1} \cong I_n \text{ and for every } 0 \leq m < n, I_{m+1} \ncong I_m.$$

The following lemma shows that the complete function calculates a stable fixed point.

Lemma 5. Let $\mathsf{complete}(I) = I_n$, as in Definition 12. Then for all $m \geq n$ $I_m \cong I_n$.

Proof . Consider I_m where $m > n$ and $I_m \cong I_{m-1}$. Then $I_m = \mathsf{filter}(I_{m-1} \cup \mathsf{result}(I_{m-1}, I_{m-1}))$. Suppose $c \in \mathsf{result}(I_{m-1}, I_{m-1})$. Then, since I_{m-1} is complete, $\pi_{\mathsf{vars}(c)}(I_{m-1}) \models c$. If $c \in I_{m-1}$ then the result hold immediately. If $c \notin I_{m-1}$ then $\mathsf{filter}(I_{m-1} \cup \{c\}) \cong I_{m-1}$ and the result holds. □

The intention is that complete returns closed systems, and this is established by the following theorem.

Theorem 1. Let $C \subseteq \mathsf{TVPI}_X$. Then $\mathsf{closed}(\mathsf{complete}(C))$ holds.

Proof . Since $\mathsf{complete}(C) \equiv C$ it is enough to show that for every $c \in \mathsf{TVPI}_X$, if $\mathsf{complete}(C) \models c$, then $\pi_{\mathsf{vars}(c)}(\mathsf{complete}(C)) \models c$. Following the proof of Lemma 4 $\pi_{\mathsf{vars}(c)}(\mathsf{complete}(C)) \equiv \pi_{\mathsf{vars}(c)}(I_k)$, where $k = \lceil \lg_2(|X|) \rceil$, hence if $C \models c$, then $\pi_{\mathsf{vars}(c)}(\mathsf{complete}(C)) \models c$. □

The remainder of this paper is concerned with calculating completion, hence maintaining closed systems.

4.2. Incremental completion

This work is primarily concerned with *incremental* completion in two variables per inequality systems, that is, considering how to make a system complete after the addition of a single new inequality to a complete system. Formally, where $I \subseteq \mathsf{TVPI}_X$, $\mathsf{complete}(I) = I$ and $c_0 \in \mathsf{TVPI}_X$, the incremental completion is $\mathsf{complete}(I \cup \{c_0\})$, which is a closed system.

As demonstrated in the introduction, when a new inequality is added to a TVPI system the resultant calculus allows many new inequalities to be derived from arbitrarily complicated derivation trees. However, with the exception of false, only those derived from one or two result steps need to be considered. The approach taken here is to consider derivation trees as objects of study.

The following sections give a series of lemmata relating to derivations trees:

- a result on redundancy (Lemma 6), showing that redundant inequalities cannot be used in the derivation of non-redundant inequalities;
- two results on compaction of derivations, showing that a derivation of a non-redundant inequality consisting of three successive applications of the result operation can be rewritten to a derivation of lower depth (one result for binary inequalities, Lemma 7, and one for unary inequalities, Lemma 9);

- a result on linearisation of derivations (Lemma 10), showing that the derivation of a non-redundant inequality can be rewritten so that one premiss of every result step is a leaf;
- two results showing that it is unnecessary to use the new inequality added to the complete system more than once, Lemmata 11 and 12.

The concluding Theorem 2 uses the lemmata to demonstrate that any element of the incremental completion can be generated using at most two result steps (with the exception of false indicating an inconsistent system, which can be detected within the closed system). The proofs of the lemmata and theorem are constructive, and contain rewriting rules for derivations in order to demonstrate their results.

Example 7. Returning to the derivation given in Example 1, first observe that $\{c_1, c_2, c_3\}$ is a complete, hence closed system of inequalities. Adding c_0 to this system allows additional inequalities to be derived using the result calculus. Example 1 gives one such derivation, which involves more than two result steps. The concluding Theorem 2 says that c can either be derived using fewer result steps, or that $c \notin \mathsf{complete}(\{c_0, c_1, c_2, c_3\})$. The proofs of the theorem and lemmata specify reductions to demonstrate this. In this example, case (5) of the proof of Theorem 2 indicates that Lemma 10 (linearisation), following by Lemma 12 (multiple use, which in turn utilises the compaction Lemma 7) can be used to show that $c \notin \mathsf{complete}(\{c_0, c_1, c_2, c_3\})$. The reasoning is illustrated in Example 13 and Example 11, and the redundancy indicated by the latter is described in Example 9. The complete and closed system generated by adding c_0 to $\{c_1, c_2, c_3\}$ is considered in Example 15.

4.3. Inconsistency

As noted above, and demonstrated in this paper, any inequality in $\mathsf{complete}(I \cup \{c_0\})$ can be found in at most two results steps. However, it should be noted that inconsistency of a system might not be detected in these two steps. This is illustrated in the following example.

Example 8. Let $I = \{c_1, c_2\}$, where $c_1 \equiv -x - u \leq 0$, $c_2 \equiv -3x - u \leq -3$, hence $\mathsf{closed}(I)$ and $\mathsf{complete}(I) = I$. Consider adding $c_0 \equiv 2x + u \leq 1$ to I. Observe that the system is then inconsistent:

$$\dfrac{\dfrac{c_0 \quad c_1}{x \leq 1}\,u \qquad \dfrac{c_0 \quad c_2}{-x \leq -2}\,u}{0 \leq -1}\,x$$

where $0 \leq -1 \equiv \mathsf{false}$. Observe that there is no selection of $c_i, c_j \in I$ so that $\mathsf{result}(\mathsf{result}(c_0, c_i), c_j) = \mathsf{false}$.

At the heart of this paper is the result that when incrementally adding a new inequality to a closed system of inequalities, any inequality that can strengthen a two dimensional projection can be generated in at most two result steps.

In the case above, the two dimensional projection is already inconsistent, so generating false does not tighten the projection, hence it is similar to a redundant inequality, hence is not generated. However, false is needed for closure. This does, however, mean that this inconsistency of the projection (hence system) needs to be explicitly noted.

4.4. Redundancy

Redundancy is important in two senses. The first is that using the result calculus, inequalities redundant with respect to the entire system are added in order to make them explicit. In a second sense, within a two variable projection inequalities that are redundant within this projection are not part of the completion (and are unnecessary for a closed system) and can be removed. That is, these inequalities are redundant with respect to a complete system. Redundancy is primarily used in this second sense through much of the rest of this paper, and is defined below.

Definition 13. Inequality $c \in \mathsf{TVPI}_X$ is said to be *redundant* with respect to $I \subseteq \mathsf{TVPI}_X$ iff $I \models c$ and $c \notin \mathsf{complete}(I)$.

In any two variable projection, a redundant inequality can be obtained as the positive linear combination of one or two other inequalities. This is represented with the following notation describing the entailment of the inequality in the conclusion by the two in the premisses:

$$\frac{a_1x + b_1y \leq e_1 \quad a_2x + b_2y \leq e_2}{(\lambda_1 a_1 + \lambda_2 a_2)x + (\lambda_1 b_1 + \lambda_2 b_2)y \leq \lambda_1 e_1 + \lambda_2 e_2 + \delta} +_{(\lambda_1,\lambda_2,\delta)}$$

This says that multiplying the first premiss by $\lambda_1 \geq 0$ and the second premiss by $\lambda_2 \geq 0$, summing the results and adding $\delta \geq 0$ to the constant term gives the conclusion.

The following lemma shows that for system of inequalities I, if inequality c is redundant with respect to $\mathsf{complete}(I)$ then the resultant of c with *any* inequality c_0 is redundant with respect to the completion of $I \cup \{c_0\}$ (hence also redundant with respect to $I \cup \{c_0\}$).

This is a powerful result, redundancy really means just that: a redundant inequality is useless for forming non-redundant inequalities, any resultant formed is guaranteed to be redundant. This will be useful in later results, where derivations of inequalities are rewritten and after rewriting it might be that some intermediates are redundant.

Lemma 6 (Redundancy). Let $I \subseteq \mathsf{TVPI}_X$ and let $c, c_0 \in \mathsf{TVPI}_X$ where $x \in \mathsf{vars}(c)$, $x \in \mathsf{vars}(c_0)$. Suppose $c \notin \mathsf{complete}(I)$ and $I \models c$. Then $\mathsf{complete}(I \cup \{c_0\}) \models \mathsf{result}(c_0, c, x)$ and $\mathsf{result}(c_0, c, x) \notin \mathsf{complete}(I \cup \{c_0\})$.

Proof . Without loss of generality there are two cases to consider:

i) Where $|\mathsf{vars}(c)| = 1$.

ii) Where $|\mathsf{vars}(c)| = 2$.

i) By the definition of closed there exists $c_1 \in \mathsf{complete}(I)$, $c_1 \not\equiv c$ such that $\{c_1\} \models c$. Suppose $c \equiv ax \leq e$ and $c_0 \equiv a_0x + b_0y \leq e_0$ (where it might be that $b_0 = 0$). Hence $\mathsf{result}(c_0, c, x) = |a|b_0y \leq |a|e_0 + |a_0|e$. Then put $c_1 \equiv ax \leq e - \delta$, $\delta > 0$ and consider $c_1' = \mathsf{result}(c_0, c_1, x) = |a|b_0y \leq |a|e_0 + |a_0|e - |a_0|\delta$. Observe that $c_1' \models \mathsf{result}(c_0, c, x)$ and the result holds.

ii) Suppose $\mathsf{vars}(c) = \{x, z\}$ and $\mathsf{vars}(c_0) \subseteq \{x, y\}$. Consider $c \equiv ax + bz \leq e$, where $I \models c$, $c_0 \equiv a_0x + b_0y \leq e_0$ (where it might be that $b_0 = 0$) and $c' \equiv \mathsf{result}(c_0, c, x)$. Then (since $I \models c$ and noting Theorem 1) one of the two cases holds:
   (a) There exists $c_1 \in \mathsf{complete}(I)$ such that $c_1 \equiv ax + bz \leq e - \delta$, for some $\delta > 0$ (that is, $\{c_1\} \models c$)
   (b) There exists $c_1, c_2 \in \mathsf{complete}(I)$, $\mathsf{vars}(\{c_1, c_2\}) = \mathsf{vars}(c)$ such that:

$$\frac{c_1 \quad c_2}{c} +_{(\lambda_1,\lambda_2,\delta)}$$

In each case, considering the resultant of c_0 with c_1 and/or c_2 will lead to a demonstration of the redundancy of c'.

   (a) $c' = \mathsf{result}(c_0, c, x)$ is derived as follows:

$$\frac{a_0x + b_0y \leq e_0 \quad ax + bz \leq e}{|a|b_0y + |a_0|bz \leq |a|e_0 + |a_0|e} \; x$$

   Since $\{c_1\} \models c$ observe that $c_1' = \mathsf{result}(c_0, c_1, x)$ is derived:

$$\frac{a_0x + b_0y \leq e_0 \quad ax + bz \leq e - \delta}{|a|b_0y + |a_0|bz \leq |a|e_0 + |a_0|(e - \delta)} \; x$$

   Hence $\{c_1'\} \models c'$, that is $\mathsf{complete}(I \cup \{c_0\}) \models c'$.

   (b) $c' = \mathsf{result}(c_0, c, x)$ is derived as follows:

$$\frac{c_0 \quad \dfrac{c_1 \quad c_2}{c} +_{(\lambda_1,\lambda_2,\delta)}}{c'} \; x$$

   There are four cases to considered, depending on the values of a_1, a_2. Consider the case where $a_1 > 0$ and $a_2 < 0$ and observe that

$$\frac{\dfrac{c_0 \quad c_1}{c_1'} \; x \quad \dfrac{c_1 \quad c_2}{c_{12}} \; x}{c'} +_{(\lambda_1 - \lambda_2 \frac{|a_2|}{|a_1|}, \lambda_2 \frac{|a_0|}{|a_1|}, |a_0|\delta)}$$

   The other cases are similar, demonstrating the result. □

Note that the conclusion of Lemma 6, that $\mathsf{complete}(I \cup \{c_0\}) \models \mathsf{result}(c_0, c, x)$ might also be written as, let $c' = \mathsf{result}(c_0, c, x)$, then $\pi_{\mathsf{vars}(c')}(\mathsf{complete}(I \cup \{c_0\}) \models c'$.

Consider the following example illustrating Lemma 6.

Example 9. Consider again the inequalities from Example 1. Observe that

$$\frac{\overset{c_0}{x + y \leq 1} \qquad \frac{\overset{c_1}{-2x + u \leq 2} \qquad \frac{\overset{c_0}{x + y \leq 1} \quad \overset{c_2}{-4y - x \leq 1}}{3x \leq 5}\, y}{-x + 2u \leq 9}\, +_{2,1,0}}{y + 2u \leq 10}\, x$$

and that this derivation demonstrates that $-x + 2u \leq 9$ is redundant, therefore according to Lemma 6 $y + 2u \leq 10$ should also be redundant. The following rewriting, following the proof of the lemma shows this:

$$\frac{\frac{\overset{c_0}{x + y \leq 1} \quad \overset{c_1}{-2x + u \leq 2}}{2y + u \leq 4}\, x \qquad \frac{\overset{c_1}{-2x + u \leq 2} \quad \frac{\overset{c_0}{x + y \leq 1} \quad \overset{c_2}{-4y - x \leq 1}}{3x \leq 5}\, y}{3u \leq 16}\, x}{y + 2u \leq 10}\, +_{(\frac{1}{2}, \frac{1}{2}, 0)}$$

Hence $y + 2u \leq 10 \notin \mathsf{complete}(\{c_0, c_1, c_2, c_3\})$.

4.5. Compaction of a chain

The results in this section and the next concern chains of resultant calculations, explaining that combining a new inequality c_0 with three inequalities drawn from a complete system is unnecessary. That is, if inequality c is derived in three resultant steps, it can be shown either to be redundant in a two variable projection or to be derivable from fewer resultant steps. Lemma 7 demonstrates this in the case that the derived inequality c contains exactly two variables. Lemma 9 then demonstrates this in the case that the derived inequality c contains only a single variable (and will ensure that the strongest unary inequality is included, even if this is redundant, as in the definition of closed).

Each proof will present its case analysis as a skeleton, that is, showing the variables involved in the chain of resultant steps, abstracting away from numerical details of coefficients and constants.

Lemma 7 (Compaction). Let $c_0 \in \mathsf{TVPI}_X$, $c_1, c_2 \in I \subseteq \mathsf{TVPI}_X$, $c_3 \in I \cup \{c_0\}$ and $x_0, x_1, x_2 \in X$, where $\mathsf{complete}(I) = I$. If $c \in \mathsf{complete}(I \cup \{c_0\})$ and

$$c = \mathsf{result}(\mathsf{result}(\mathsf{result}(c_0, c_1, x_0), c_2, x_1), c_3, x_2)$$

and $|\mathsf{vars}(c)| = 2$ then there exists $d_0, d_1 \in I$ and $y_0, y_1 \in X$ such that

$$c = \mathsf{result}(\mathsf{result}(c_0, d_0, y_0), d_1, y_1)$$

Proof . The structure of the series of resultants being considered is:

$$\frac{\frac{\frac{c_0 \quad c_1}{c_1'}\, x_0 \quad c_2}{c_2'}\, x_1 \quad c_3}{c}\, x_2$$

(1) $\dfrac{\dfrac{\dfrac{\{x,y\} \quad \{x,u\}}{\{y,u\}}\,x \qquad \{u,v\}}{\{y,v\}}\,u \qquad \{v,z\}}{\{y,z\}}\,v$	(2) $\dfrac{\dfrac{\dfrac{\{x,y\} \quad \{x,u\}}{\{y,u\}}\,x \qquad \{u,v\}}{\{y,v\}}\,u \qquad \{y,z\}}{\{v,z\}}\,y$
(3) $\dfrac{\dfrac{\dfrac{\{x,y\} \quad \{x,u\}}{\{y,u\}}\,x \qquad \{y,v\}}{\{u,v\}}\,y \qquad \{u,z\}}{\{v,z\}}\,u$	(4) $\dfrac{\dfrac{\dfrac{\{x,y\} \quad \{x,u\}}{\{y,u\}}\,x \qquad \{y,v\}}{\{u,v\}}\,y \qquad \{v,z\}}{\{u,z\}}\,v$

Figure 3: Possible resultant combinations for generating a binary inequality

The proof proceeds by giving a series of reductions demonstrating that by reordering the application of the result operation either the conclusion of the lemma holds, or $c \notin \mathsf{complete}(I \cup \{c_0\})$, that is the premiss of the lemma does not hold.

There are four possible configurations that lead to different combinations of the variables being eliminated in the sequence of resultant steps. The skeletons of these four cases are given in Figure 3. Only case four is considered in detail.

Where $c_0 \equiv a_0x + b_0y \leq e_0$, $c_1 \equiv a_1x + b_1u \leq e_1$, $c_2 \equiv a_2y + b_2v \leq e_2$ and $c_3 \equiv a_3v + b_3z \leq e_3$, with $x_0 = x, x_1 = y, x_2 = v$. In this case

$$\dfrac{\dfrac{\dfrac{c_0 \quad c_1}{c_1'}\,x \qquad c_2}{c_2'}\,y \qquad c_3}{c}\,v$$

reduces to

$$\dfrac{\dfrac{c_0 \quad c_1}{c_1'}\,x \qquad \dfrac{c_2 \quad c_3}{c_{23}}\,v}{c}\,y$$

Put $d_0 = c_1$, $d_1 = \mathsf{result}(c_2, c_3, v)$, $y_0 = x$ and $y_1 = y$. Since I is complete, either $d_1 \in I$ and the result holds, or $d_1 \notin I$, therefore $\pi_{\mathsf{vars}(d_1)}(I) \models d_1$, hence by Lemma 6 $c \notin \mathsf{complete}(I \cup \{c_0\})$.

However, in the case that the variables v and x are the same an alternative analysis is needed. Here the initial derivation can be reduced to

$$\dfrac{\dfrac{c_0 \quad c_1}{c_1'}\,x \qquad \dfrac{c_2 \quad c_3}{c_{23}}\,v}{c}\,+_{(|a_2||a_3|,|a_1||b_0|,0)}$$

If $c_{23} \in \mathsf{complete}(I)$ then c is redundant because it is a linear combination of c_1' and c_{23}. If $c_{23} \notin \mathsf{complete}(I)$, but $c_1' \in \mathsf{complete}(I \cup \{c_0\})$ then by Lemma 6 it also follows that c is redundant. Likewise if $c_1' \notin \mathsf{complete}(I \cup \{c_0\})$. In all cases, the pre-condition of the lemma that $c \in \mathsf{complete}(I \cup \{c_0\})$ is contradicted. Other variable identities lead to variations on these two cases. □

Example 10. To illustrate case 4 of the proof, as detailed above, suppose $c_0 \equiv x + y \leq 1$ and $c_1, c_2, c_3 \in I \subseteq \mathsf{TVPI}_X$ where $\mathsf{complete}(I) = I$ and

$$c_1 \equiv -x - 2u \leq 1 \qquad c_2 \equiv -y - 5v \leq 1 \qquad c_3 \equiv v - 2z \leq 0$$

Then c in the proof is derived:

$$\frac{\frac{\frac{\overset{c_0}{x+y\leq 1} \quad \overset{c_1}{-x-2u\leq 1}}{-2u+y\leq 2}\,x \quad \overset{c_2}{-y-5v\leq 1}}{-2u-5v\leq 3}\,y \quad \overset{c_3}{v-2z\leq 0}}{-2u-10z\leq 3}\,v$$

This can be reduced to the following derivation of c:

$$\frac{\frac{\overset{c_0}{x+y\leq 1} \quad \overset{c_1}{-x-2u\leq 1}}{-2u+y\leq 2}\,x \quad \frac{\overset{c_2}{-y-5v\leq 1} \quad \overset{c_3}{-3y-2z\leq 0}}{-y-10z\leq 1}\,v}{-2u-10v\leq 3}\,y$$

that is, where $d_0 = c_1$ $d_1 = \mathsf{result}(c_2, c_3, v)$, $y_0 = x$, and $y_1 = y$. Notice that either $d_0 \in I$, or Lemma 6 says that $c \notin \mathsf{complete}(I \cup \{c_0\})$.

Example 11. Returning to the set of inequalities from Example 1, consider the following where x and v coincide, leading to the redundancy of c in $\mathsf{complete}(I \cup \{c_0\})$, again as given in the proof above:

$$\frac{\frac{\frac{\overset{c_0}{x+y\leq 1} \quad \overset{c_1}{-2x+u\leq 2}}{2y+u\leq 4}\,x \quad \overset{c_2}{-4y-x\leq 1}}{2u-x\leq 9}\,y \quad \overset{c_0}{x+y\leq 1}}{2u+y\leq 10}\,x$$

From this, the following can be observed:

$$\frac{\frac{\overset{c_0}{x+y\leq 1} \quad \overset{c_1}{-2x+u\leq 2}}{2y+u\leq 4}\,x \quad \frac{\overset{c_2}{-4y-x\leq 1} \quad \overset{c_0}{x+y\leq 1}}{-3y\leq 2}\,x}{4u+2y\leq 20}\,+_{(4,2,0)}$$

demonstrating that $2u + y \leq 10 \notin \mathsf{complete}(I \cup \{c_o\})$.

4.6. Unary inequalities

Unary inequalities are of particular importance in domains with two variables per inequality since a single unary inequality might be included in many two variable projections. The definition of closure used in this paper makes this explicit by stating that a closed system must include all unary inequalities (see Example 3). The example in the introduction shows why this is. Some of these unary inequalities might well be redundant in the sense that they are redundant in each two variable projection in which they occur. The example in the introduction can be viewed as demonstrating either that unary inequalities need to be generated whether redundant or not, or that all projections onto two or fewer variables need to be considered – then the unary inequalities are not redundant in unary projections.

Lemma 9 augments Lemma 7 by showing that when adding a single new inequality to a complete system all unary inequalities required to ensure that the new system is also complete – including redundant unary inequalities – can be derived in at most two resultant steps.

Before the main result, Lemma 9, a preliminary lemma on the entailment of unary inequalities is given. This is used in the proof of Lemma 9. Lemma 8 says that if two consistent unary inequalities entail a third, then that third unary inequality must be entailed by one of the other two.

Lemma 8. Suppose $\mathsf{vars}(c) = \{x\}$, $\mathsf{vars}(c_1), \mathsf{vars}(c_2) \subseteq \{x\}$ and that $\{c_1, c_2\} \models c$. Then either $c_1 \models c$ or $c_2 \models c$.

Lemma 9 (Unary inequalities). Suppose that $I \subseteq \mathsf{TVPI}_X$ and $\mathsf{complete}(I) = I$. Suppose $c_0 \in \mathsf{TVPI}_X$ and that $c \in \mathsf{complete}(I \cup \{c_0\})$ where

$$c = \mathsf{result}(\mathsf{result}(\mathsf{result}(c_0, c_1), c_2), c_3)$$

and $|\mathsf{vars}(c)| = 1$, $c_1, c_2 \in I$ and $c_3 \in I \cup \{c_0\}$. Then there exists $d_0, d_1 \in I$ such that one of the following holds:

1. $c = \mathsf{result}(\mathsf{result}(c_0, d_0), d_1)$
2. $c = \mathsf{result}(c_0, d_0)$

Proof . Figure 4 gives the structure of three resultant steps, and skeletons (the variables occurring in each inequality) for each of the fifteen possible configurations where three resultant steps end in a unary inequality. Analysis of each case demonstrates the result (often by showing that $c \notin \mathsf{complete}(I \cup \{c_0\})$. Only cases 2 and 3 are detailed here.

2. Where $c_0 \equiv a_0 z + b_0 u \leq e_0$, $c_1 \equiv a_1 y + b_1 u \leq e_1$, $c_2 \equiv a_2 x + b_2 z \leq e_2$ and $c_3 \equiv a_3 y \leq e_3$. Consider:

$$\dfrac{\dfrac{c_0 \qquad \dfrac{c_1 \quad c_3}{c_{13}}\; y}{c'_{13}}\; u \qquad\qquad c_2}{c}\; z$$

   $c_{13} = \mathsf{result}(c_1, c_3, y)$ is a unary inequality, hence either $c_{13} \in I$ and case 1 holds or $c_{13} \notin I$ and by Lemma 6 $c \notin \mathsf{complete}(I \cup \{c_0\})$. Note that the result still holds if $u = x$.

$\dfrac{\dfrac{\dfrac{c_0 \quad c_1}{c_1'} \quad c_2}{c_2'} \quad c_3}{c}$	(1) $\dfrac{\dfrac{\dfrac{\{x,u\} \quad \{z,u\}}{\{x,z\}}\,u \quad \{z,y\}}{\{x,y\}}\,z \quad \{y\}}{\{x\}}\,y$
(2) $\dfrac{\dfrac{\dfrac{\{z,u\} \quad \{y,u\}}{\{z,y\}}\,u \quad \{x,z\}}{\{x,y\}}\,z \quad \{y\}}{\{x\}}\,y$	(3) $\dfrac{\dfrac{\dfrac{\{z,u\} \quad \{y,u\}}{\{z,y\}}\,u \quad \{z\}}{\{y\}}\,z \quad \{x,y\}}{\{x\}}\,y$
(4) $\dfrac{\dfrac{\dfrac{\{z,u\} \quad \{y,u\}}{\{z,y\}}\,u \quad \{z,y\}}{\{y\}}\,z \quad \{x,y\}}{\{x\}}\,y$	(5) $\dfrac{\dfrac{\dfrac{\{z,u\} \quad \{u\}}{\{z\}}\,u \quad \{z,y\}}{\{y\}}\,z \quad \{x,y\}}{\{x\}}\,y$
(6) $\dfrac{\dfrac{\dfrac{\{z,u\} \quad \{z,u\}}{\{z\}}\,u \quad \{z,y\}}{\{y\}}\,z \quad \{x,y\}}{\{x\}}\,y$	(7) $\dfrac{\dfrac{\dfrac{\{x,u\} \quad \{z,u\}}{\{x,z\}}\,u \quad \{z,y\}}{\{x,y\}}\,z \quad \{x,y\}}{\{x\}}\,y$
(8) $\dfrac{\dfrac{\dfrac{\{z,u\} \quad \{y,u\}}{\{z,y\}}\,u \quad \{x,z\}}{\{x,y\}}\,z \quad \{x,y\}}{\{x\}}\,y$	(9) $\dfrac{\dfrac{\dfrac{\{y,u\} \quad \{z,u\}}{\{y,z\}}\,u \quad \{z,y\}}{\{y\}}\,z \quad \{x,y\}}{\{x\}}\,y$
(10) $\dfrac{\dfrac{\dfrac{\{z,u\} \quad \{x,u\}}{\{x,z\}}\,u \quad \{z,y\}}{\{x,y\}}\,z \quad \{y\}}{\{x\}}\,y$	(11) $\dfrac{\dfrac{\dfrac{\{y,u\} \quad \{z,u\}}{\{z,y\}}\,u \quad \{x,z\}}{\{x,y\}}\,z \quad \{y\}}{\{x\}}\,y$
(12) $\dfrac{\dfrac{\dfrac{\{z,u\} \quad \{x,u\}}{\{x,z\}}\,u \quad \{z,y\}}{\{x,y\}}\,z \quad \{x,y\}}{\{x\}}\,y$	(13) $\dfrac{\dfrac{\dfrac{\{y,u\} \quad \{z,u\}}{\{z,y\}}\,u \quad \{x,z\}}{\{x,y\}}\,z \quad \{x,y\}}{\{x\}}\,y$
(14) $\dfrac{\dfrac{\dfrac{\{u\} \quad \{u,z\}}{\{z\}}\,u \quad \{z,y\}}{\{y\}}\,z \quad \{x,y\}}{\{x\}}\,y$	(15) $\dfrac{\dfrac{\dfrac{\{y,u\} \quad \{z,u\}}{\{z,y\}}\,u \quad \{z\}}{\{y\}}\,z \quad \{x,y\}}{\{x\}}\,y$

Figure 4: Possible derivations for generating a unary inequality from c_0, c_1, c_2, c_3

3. Where $c_0 \equiv a_0 z + b_0 u \leq e_0$, $c_1 \equiv a_1 y + b_1 u \leq e_1$, $c_2 \equiv a_2 z \leq e_2$ and

$c_3 \equiv a_3x + b_3y \leq e_3$. The original derivation is:

$$\frac{\dfrac{\dfrac{a_0z + b_0u \leq e_0 \quad a_1y + b_1u \leq e_1}{|b_1|a_0z + |b_0|a_1y \leq |b_1|e_0 + |b_0|e_1}\,u \qquad a_2z \leq e_2}{|a_2||b_0|a_1y \leq |a_2||b_1|e_0 + |a_2||b_0|e_1 + |a_0||b_1|e_2}\,z \qquad a_3x + b_3y \leq e_3}{|a_1||a_2||b_0|a_3x \leq |a_2||b_1||b_3|e_0 + |a_2||b_0||b_3|e_1 + |a_0||b_1||b_3|e_2 + |a_1||a_2||b_0|e_3}\,y$$

Hence the same inequality might be derived

$$\frac{\dfrac{a_0z + b_0u \leq e_0 \quad a_2z \leq e_2}{|a_2|b_0u \leq |a_2|e_0 + |a_0|e_2}\,z \qquad \dfrac{a_1y + b_1u \leq e_1 \quad a_3x + b_3y \leq e_3}{|a_1|a_3x + |b_3|b_1u \leq |b_3|e_1 + |a_1|e_3}\,y}{|a_1||a_2||b_0|a_3x \leq |a_2||b_1||b_3|e_0 + |a_2||b_0||b_3|e_1 + |a_0||b_1||b_3|e_2 + |a_1||a_2||b_0|e_3}\,u$$

If $c_{13} = \mathsf{result}(c_1, c_3, y) \in I$ then case 1 holds. If $c_{13} \notin I$ then by Lemma 6 $c \notin \mathsf{complete}(I \cup \{c_0\})$. Note that the result still holds if $z = x$.
If in the derivation immediately above $u = x$, then c_2' and c_{13} are given by:

$$\frac{a_0z + b_0x \leq e_0 \quad a_2z \leq e_2}{|a_2|b_0x \leq |a_2|e_0 + |a_0|e_2}\,z$$

and

$$\frac{a_1y + b_1x \leq e_1 \quad a_3x + b_3y \leq e_3}{(|a_1|a_3 + |b_3|b_1)x \leq |b_3|e_1 + |a_1|e_3}\,y$$

Hence (scaling with $\lambda_1 = |b_1||b_3|$ and $\lambda_2 = |a_2||b_0|$ respectively)

$$\begin{array}{c} |a_2||b_0|(|a_1|a_3 + |b_3|b_1)x + |a_2||b_1||b_3|b_0x \\ \leq \\ |a_2||b_0|(|b_3|e_1 + |a_1|e_3) + |b_1||b_3|(|a_2|e_0 + |a_0|e_2) \end{array}$$

Noting that $b_0b_1 < 0$ this gives

$$|a_1||a_2||b_0|a_3x \leq |b_1||b_3||a_2|e_0 + |a_2||b_0||b_3|e_1 + |a_0||b_1||b_3|e_2 + |a_1||a_2||b_0|e_3$$

Hence by Lemma 8 $c_2' \models c$ or $c_{13} \models c$. In the former case then either case 2 has been demonstrated or $c \notin \mathsf{complete}(I \cup \{c_0\})$. In the latter case, either $c \in I$ or $c \notin \mathsf{complete}(I \cup \{c_0\})$. In all cases the result holds. □

Example 12. Suppose that $\mathsf{complete}(I) = I$. Further suppose that $c_0 \equiv z + u \leq 1$, $c_1 \equiv y - u \leq 2$, $c_2 \equiv -2z \leq -1$, $c_3 \equiv x - y \leq -1$, where $c_1, c_2, c_3 \in I$, then $c = \mathsf{result}(\mathsf{result}(\mathsf{result}(c_0, c_1, u), c_2, z), c_3, y) = 2x \leq 3$. As in case 2 of the lemma, c can be derived as

$$\frac{\dfrac{\dfrac{\overset{c_0}{z + u \leq 1} \quad \overset{c_1}{y - u \leq 2}}{z + y \leq 3}\,u \qquad \overset{c_2}{-2z \leq -1}}{2y \leq 5}\,z \qquad \overset{c_3}{x - y \leq -1}}{2x \leq 3}\,y$$

With $d_0 = c_2$ and $d_1 \equiv x - u \leq 1 = \mathsf{result}(c_1, c_3, y)$, if $d_1 \in I$ then the result is demonstrated by the following rewriting:

$$\dfrac{\dfrac{\overset{c_0}{z+u\leq 1} \qquad \overset{c_2}{-2z\leq -1}}{2u\leq 1}\,z \qquad \dfrac{\overset{c_1}{y-u\leq 2} \qquad \overset{c_3}{x-y\leq -1}}{x-u\leq 1}\,y}{2x\leq 3}\,u$$

However, if the variables x and u coincide, then observe that from

$$\dfrac{\dfrac{\dfrac{\overset{c_0}{z+x\leq 1} \qquad \overset{c_1}{y-x\leq 2}}{z+y\leq 3}\,x \qquad \overset{c_2}{-2z\leq -1}}{2y\leq 5}\,z \qquad \overset{c_3}{x-y\leq -1}}{2x\leq 3}\,y$$

it can be observed that

$$\dfrac{\overset{c_0}{z+x\leq 1} \qquad \overset{c_2}{-2z\leq -1}}{2x\leq 1}\,z \qquad \dfrac{\overset{c_1}{y-x\leq 2} \qquad \overset{c_3}{x-y\leq -1}}{0\leq 1}\,y$$

and that $\{2x \leq 1, 0 \leq 1\} \models 2x \leq 3$, hence by Lemma 8 either $2x \leq 1 \models 2x \leq 3$ or $0 \leq 1 \models 2x \leq 3$, and it is easy to see that it is the former in this case.

4.7. Linearisation of a tree

The next step in demonstrating the correctness of the incremental completion algorithm is to demonstrate that any derivation tree of inequality c can be replaced by a linear derivation of c (see the following definition).

Definition 14. A derivation is said to be *linear* if each result step in the derivation has at most one premiss with size greater than 0. By convention the right premiss will have size 0.

Lemma 10 shows that a derivation whose final resultant calculation has linear premisses can be replaced by a linear derivation. Any inequality can either be derived linearly, or is redundant, or is false (see the example in section 4.3).

Lemma 10 (Linearisation). Let $I \subseteq \mathsf{TVPI}_X$. Suppose that $c_0, c_1, \ldots, c_j \in I$ and $d_0, d_1, \ldots, d_k \in I$ where $j, k \in \mathbb{N}$. Where $c'_0 = c_0$ and $d'_0 = d_0$, define $c'_{i+1} = \mathsf{result}(c'_i, c_{i+1}, x_i)$, where $x_i \in X$ and also define $d'_{i+1} = \mathsf{result}(d'_i, d_{i+1}, y_i)$, where $y_i \in X$. Consider $c = \mathsf{result}(c'_j, d'_k, z)$, where $z \in X$. At least one of the following holds:

1. There exists $f_0, f_1, \ldots, f_\ell \in I$, where $f_0 = f'_0 = c_0$, $f'_{i+1} = \mathsf{result}(f'_i, f_{i+1}, w_i)$, $w_i \in X$, and $c \equiv f'_\ell$
2. c is redundant with respect to $\mathsf{complete}(I)$, that is, $\mathsf{complete}(I) \models c$ and $c \notin \mathsf{complete}(I)$
3. $c \equiv \mathsf{false}$

Proof . If $c = \mathsf{result}(c'_j, d_0, z)$, then case 1 is immediate. If $c = \mathsf{result}(c_0, d'_k, z)$, note the symmetry of the premisses (so that $c = \mathsf{result}(c'_j, d'_k, z) = \mathsf{result}(d'_k, c'_j, z)$), hence case 1 is again immediate.

In other cases, the derivation can be written, and it is argued that repeated rewritings will establish the result. The core case and one restricted case are detailed here.

Where $j, k \geq 0$ consider $c = \mathsf{result}(c'_{j+1}, d'_{k+1}, z)$. That is, where result is applied to two inequalities that are not leaves of the derivation tree. With x, y, z, w distinct variables and all coefficients are non-zero, then

$$\dfrac{a_1x + b_1z \leq e_1 \quad \dfrac{a_2z + b_2w \leq e_2 \quad a_3w + b_3y \leq e_3}{|a_3|a_2z + |b_2|b_3y \leq |a_3|e_2 + |b_2|e_3}\, w}{|a_2||a_3|a_1x + |b_1||b_2|b_3y \leq |a_2||a_3|e_1 + |a_3||b_1|e_2 + |b_1||b_2|e_3}\, z$$

can be rewritten to:

$$\dfrac{\dfrac{a_1x + b_1z \leq e_1 \quad a_2z + b_2w \leq e_2}{|a_2|a_1x + |b_1|b_2w \leq |a_2|e_1 + |b_1|e_2}\, z \quad a_3w + b_3y \leq e_3}{|a_2||a_3|a_1x + |b_1||b_2|b_3y \leq |a_2||a_3|e_1 + |a_3||b_1|e_2 + |b_1||b_2|e_3}\, w$$

That is, $c = \mathsf{result}(c'_{j+1}, \mathsf{result}(d'_k, d_k, w), z) = \mathsf{result}(\mathsf{result}(c'_{j+1}, d'_k, z), d_k, w)$. There is a symmetric case when z derives from the second premiss of d'_{k+1}.

The four variables may coincide. The only case detailed here is when y and z coincide, then $\mathsf{result}(c'_{j+1}, d'_{k+1}, z)$ can be rewritten to $\mathsf{result}(d'_{k+1}, c'_{j+1}, z)$.

Associate a weight to the derivation of inequality c, $c = \mathsf{result}(c_1, c_2, v)$. This weight is an ordered triple $(n, |\mathsf{vars}(c_1)|, \mathsf{size}(c_2))$, where n is the number of inequalities above c in the derivation with a right premiss with size greater than 0. In the original derivation of c, observe that at most one result operation has $\mathsf{size}(c_2) > 0$ (that is, $n \leq 1$). Each rewriting step results in at most one result step with the size of the second argument greater than zero (again $n \leq 1$). Hence the weights are totally ordered. The rewriting process has terminated if there are no result steps with right premiss with size greater than 0 (i.e. when $n = 0$, or when it is observed that the concluding inequality is redundant). Now observe that for each rewriting step either c is found to be redundant or the weight of the derivation of c is strictly less than previously. Hence by induction the result holds. □

Example 13. Consider again the derivation from Example 1. Figure 5 gives this derivation, indicating which inequalities are taking the role of d_0 and d_1 in Lemma 10, and illustrates its linearisation.

The weight of the first derivation of $2u + z \leq 11$ is $(1, 2, 1)$. Here, the rewrite rule applies: the variable eliminated in the result is x deriving from the first premiss, $d_0 = c_0$. After rewriting, the weight of the derivation of $2u + z \leq 11$ is $(0, 2, 0)$, strictly less than before, and the resulting derivation is linear.

Original derivation:

$$\dfrac{\dfrac{\dfrac{\overset{c_0}{x+y\le 1} \qquad \overset{c_1}{-2x+u\le 2}}{2y+u\le 4}\,x \qquad \overset{c_2}{-4y-x\le 1}}{2u-x\le 9}\,y \qquad \dfrac{\overset{d_0=c_0}{x+y\le 1}\qquad \overset{d_1=c_3}{-y+z\le 1}}{x+z\le 2}\,y}{2u+z\le 11}\,x$$

Linearisation of the original derivation:

$$\dfrac{\dfrac{\dfrac{\dfrac{\overset{c_0}{x+y\le 1} \qquad \overset{c_1}{-2x+u\le 2}}{2y+u\le 4}\,x \qquad \overset{c_2}{-4y-x\le 1}}{2u-x\le 9}\,y \qquad \overset{d_0=c_0}{x+y\le 1}}{2u+y\le 10}\,x \qquad \overset{d_1=c_3}{-y+z\le 1}}{2u+z\le 11}\,y$$

Figure 5: Linearisation of a tree composed of two chains into a single chain

4.8. Multiple use

This section analyses what happens when a derivation uses the new inequality more than once.

The following shows that from the perspective of completing a system when a new constraint c_0 is added, it is fruitless to recombine c_0 with any inequality that results from a derivation that emanates from c_0. It is sufficient to consider chains that start at c_0 with at most two intermediate inequalities; as will be seen in the concluding theorem, chains with more intermediates can be collapsed down using the preceding lemmata.

Lemma 11 (Multiple use: part 1). Let $c_0, c_1 \in \mathsf{TVPI}_X$, where $\mathsf{vars}(c_0) = \{x, y\}$. If $c = \mathsf{result}(\mathsf{result}(c_0, c_1, x), c_0, y)$ then there exists $c' = \mathsf{result}(c_0, c_1, y)$ such that $c \equiv c'$.

Proof . Where $c_0 \equiv a_0x + b_0y \le e_0$, $c_1 \equiv a_1x + b_1y \le e_1$, suppose that $c_1' \equiv \mathsf{result}(c_0, c_1, x)$ and $c \equiv \mathsf{result}(c_1', c_0, y)$, that is:

$$\dfrac{\dfrac{a_0x+b_0y\le e_0 \qquad a_1x+b_1y\le e_1}{(|a_1|b_0+|a_0|b_1)y \le |a_1|e_0+|a_0|e_1}\,x \qquad a_0x+b_0y\le e_0}{|(|a_1|b_0+|a_0|b_1)|a_0x \le |a_1||b_0|e_0+|a_0||b_0|e_1+|(|a_1|b_0+|a_0|b_1)|e_0}\,y$$

Notice that:

1. this is the only possible configuration. If $b_0 = 0$ then the second resultant step is not possible, and if $\mathsf{vars}(c_1) \neq \{x, y\}$ one of the two resultant steps is not possible.

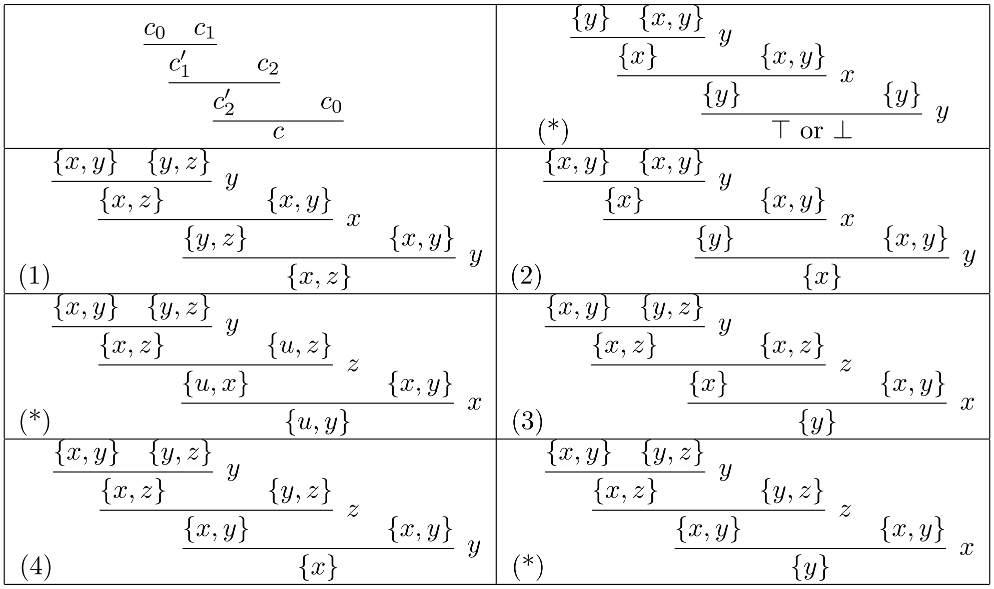


Figure 6: Resultant combination sequences involving c_0, c_1, c_2

2. $a_0 a_1 < 0$
3. $(|a_1|b_0 + |a_0|b_1)b_0 < 0$, hence $b_0 b_1 < 0$
4. Hence $|a_0 b_1| > |a_1 b_0|$

Since $b_0 b_1 < 0$ $c' = \mathsf{result}(c_0, c_1, y)$ can be obtained as follows:

$$\frac{a_0 x + b_0 y \leq e_0 \quad a_1 x + b_1 y \leq e_1}{(|b_1|a_0 + |b_0|a_1)x \leq |b_1|e_0 + |b_0|e_1}\ y$$

It can be demonstrated that $c = |a_0|c'$ (i.e. $c \equiv c'$), establishing the result.□

The final lemma considers the inequality derived by using the new inequality twice, together with two other inequalities.

Lemma 12 (Multiple use: part 2). Suppose that $I \subseteq \mathsf{TVPI}_X$, $\mathsf{complete}(I) = I$ and $c_1, c_2 \in I$. If $c \in \mathsf{complete}(I \cup \{c_0\})$, where $c = \mathsf{result}(\mathsf{result}(\mathsf{result}(c_0, c_1), c_2), c_0)$, with $|\mathsf{vars}(c)| \geq 1$, then there is $d_0 \in I$ such that $c \equiv \mathsf{result}(c_0, d_0)$.

Proof . The potential combinations of variables occurring (and being eliminated) in c_0, c_1 and c_2 are given in Figure 6. The three $(*)$ cases are not possible. Each of the four potential cases has already been considered in Lemma 7 (compaction) and Lemma 9 (unary) and the results follows. □

The following example illustrates the lemma. In addition, Example 11 earlier in the paper follows one case of the proof.

Example 14. Consider:

$$c_0 \equiv x - y \le 1 \qquad c_1 \equiv y + z \le 1 \qquad c_2 \equiv -2x - z \le 2$$

Then

$$\dfrac{\dfrac{\dfrac{\overset{c_0}{x - y \le 1} \quad \overset{c_1}{y + z \le 1}}{x + z \le 2}\,y \qquad \overset{c_2}{-2x - z \le 2}}{-x \le 4}\,z \qquad \overset{c_0}{x - y \le 1}}{-y \le 5}\,x$$

and also

$$\dfrac{\overset{c_0}{x - y \le 1} \qquad \dfrac{\overset{c_1}{y + z \le 1} \quad \overset{c_2}{-2x - z \le 2}}{y - 2x \le 3}\,z}{-y \le 5}\,x$$

That is, $c = \mathsf{result}(c_0, \mathsf{result}(c_1, c_2))$ as prescribed by the lemma.

4.9. The Incremental Completion Theorem

The findings can now be summarised in a single statement. The result follows by using the lemmata above to show that the result of any derivation tree for $c \in \mathsf{complete}(I \cup \{c_0\})$, where the leaf inequalities are drawn from $I \cup \{c_0\}$, can be collapsed into a chain that coincides with one of the three cases. The strength of the result is that *only* these simple chains need be considered when computing incremental closure.

Theorem 2. Consider adding $c_0 \in \mathsf{TVPI}_X$ to $I \subseteq \mathsf{TVPI}_X$ where $\mathsf{complete}(I) = I$. If $c \in \mathsf{complete}(I \cup \{c_0\})$ and $c \neq \mathsf{false}$, then one of the following holds:

1. $c \in I \cup \{c_0\}$
2. $c = \mathsf{result}(c_0, c_1)$ where $c_1 \in I$
3. $c = \mathsf{result}(\mathsf{result}(c_0, c_1), c_2)$ where $c_1, c_2 \in I$

Proof . Suppose that $c = \mathsf{result}(c_1, c_2, x)$ and that $c \in \mathsf{complete}(I \cup \{c_0\})$ (as assumed in the lemmata above). Since $\mathsf{size}(c_1), \mathsf{size}(c_2) < \mathsf{size}(c)$ assume inductively that c_1, c_2 are obtained as in the theorem. Then, using Lemmata 6–12, it can be shown that the result holds for c.

The premisses of the resultant can be derived in one of three ways (as in the theorem). Noting the symmetry between the two premisses, this leaves six cases to be examined.

1. $c_1 \in I \cup \{c_0\}$ and $c_2 \in I \cup \{c_0\}$. Since c_1 and c_2 cannot both be c_0 assume that $c_1 = c_0$, then $c = \mathsf{result}(c_0, c_2, x)$ and the result holds immediately. Alternatively, $c_1 \in I$ and $c_2 \in I$ and again the result is immediate.
2. $c_1 = \mathsf{result}(c_0, c_3, x)$ and $c_2 \in I \cup \{c_0\}$. That is,

$$\dfrac{\dfrac{c_0 \quad c_3}{c_2} \qquad c_0}{c}$$

Hence using Lemma 11 (multiple use, part 1) $c \equiv \mathsf{result}(c_0, c_3)$ and the result holds. Otherwise, if $c_2 \in I$ then the result holds immediately.

3. $c_1 = \mathsf{result}(\mathsf{result}(c_0, c_3), c_4)$ and $c_2 \in I \cup \{c_0\}$. That is (noting the symmetry of the premisses), one of two cases holds (the second case being where $c_2 \in I$):

$$\dfrac{\dfrac{\dfrac{c_0 \quad c_3}{c_3'} \quad c_4}{c_1} \quad c_0}{c} \qquad\qquad \dfrac{\dfrac{\dfrac{c_0 \quad c_3}{c_3'} \quad c_4}{c_1} \quad c_2}{c}$$

   The first case follows immediately from Lemma 12 (multiple use, part 2). The second case follows immediately using Lemma 7 (compaction) or Lemma 9 (unary inequalities).

4. $c_1 = \mathsf{result}(c_0, c_3)$ and $c_2 = \mathsf{result}(c_0, c_4)$. That is:

$$\dfrac{\dfrac{c_0 \quad c_3}{c_1} \quad \dfrac{c_0 \quad c_4}{c_2}}{c}$$

   Applying Lemma 10 (linearisation), and since $c \in \mathsf{complete}(I \cup \{c_0\})$, results in either:

$$\dfrac{\dfrac{\dfrac{c_0 \quad c_3}{c_1} \quad c_0}{c_1'} \quad c_4}{c} \qquad\qquad \dfrac{\dfrac{\dfrac{c_0 \quad c_3}{c_1} \quad c_4}{c_{14}} \quad c_0}{c}$$

   In the first of these cases Lemma 11 (multiple use) on the derivation of c_1' leads to:

$$\dfrac{\dfrac{c_0 \quad c_3}{c_1'} \quad c_4}{c}$$

   and the result holds. The second case is an instance of case 3 of this proof.

5. $c_1 = \mathsf{result}(\mathsf{result}(c_0, c_4), c_5)$ and $c_2 = \mathsf{result}(c_0, c_3)$. That is (noting the symmetry of the premisses):

$$\dfrac{\dfrac{\dfrac{c_0 \quad c_4}{c_4'} \quad c_5}{c_1} \quad \dfrac{c_0 \quad c_3}{c_2}}{c}$$

   Applying Lemma 10 (linearisation) for the derivation of c leads to one of the following:

$$\dfrac{\dfrac{\dfrac{\dfrac{c_0 \quad c_4}{c_4'} \quad c_5}{c_1} \quad c_0}{c_1'} \quad c_3}{c} \qquad\qquad \dfrac{\dfrac{\dfrac{\dfrac{c_0 \quad c_4}{c_4'} \quad c_5}{c_1} \quad c_3}{c_1'} \quad c_0}{c}$$

   In the first case consider the derivation of c_1', this is an instance of case 3. Then taking this together with the final resultant step gives the result using Lemma 7 (compaction) or Lemma 9 (unary inequalities).

In the second case noting that $c \in \mathsf{complete}(I \cup \{c_0\})$ and applying Lemma 7 (compaction) or Lemma 9 (unary inequalities) to the derivation of c_1' leads to:

$$\dfrac{\dfrac{\dfrac{c_0 \quad c_6}{c_6'} \quad c_7}{c_1'} \quad c_0}{c}$$

where $c_6, c_7 \in I$. This is then an instance of case 3.

6. $c_1 = \mathsf{result}(\mathsf{result}(c_0, c_3), c_4)$ and $c_2 = \mathsf{result}(\mathsf{result}(c_0, c_5), c_6)$. That is:

$$\dfrac{\dfrac{\dfrac{c_0 \quad c_3}{c_3'} \quad c_4}{c_1} \quad \dfrac{\dfrac{c_0 \quad c_5}{c_5'} \quad c_6}{c_2}}{c}$$

Applying Lemma 10 (linearisation) to this gives:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{c_0 \quad c_3}{c_3'} \quad c_4}{c_1} \quad c_7}{c_7'} \quad c_8}{c_8'} \quad c_9}{c}$$

where $c_0 \in \{c_7, c_8, c_9\}$ and $\{c_7, c_8, c_9\} - \{c_0\} = \{c_5, c_6\}$. Consider each of the three possibilities for the occurrence of c_0.
If $c_0 = c_7$. Consider the derivation of c_7'. This can be treated as in case 3. Following this with one (or two) application(s) of Lemma 7 (compaction) or Lemma 9 (unary inequalities) gives the result.
If $c_0 = c_8$. An application of Lemma 7 (compaction) or Lemma 9 (unary inequalities) to the derivation of c_7', then using case 3, and a second application of Lemma 7 (compaction) or Lemma 9 (unary inequalities) gives the result.
If $c_0 = c_9$. Two applications of Lemma 7 (compaction) or Lemma 9 (unary inequalities) to the derivation of c_8', followed by an argument as in case 3 gives the result.

In all cases the result has been shown to hold. □

Since $\mathsf{closed}(\mathsf{complete}(I \cup \{c_0\}))$ by Theorem 1 it has been established that incremental closure can be found using these newly generated inequalities.

Example 15. Consider again the inequalities from Example 1. Where $c_1 \equiv -2x + u \leq 2$, $c_2 \equiv -4y - x \leq 1$, $c_3 \equiv -y + z \leq 1$ and $I = \{c_1, c_2, c_3\}$, note that $\mathsf{complete}(I) = I$. Where $c_0 \equiv x + y \leq 1$, consider $\mathsf{complete}(I \cup \{c_0\})$. As stated in the theorem, new inequalities can be derived from one or two applications of result. This gives the nine new inequalities as below, the first seven of which are contained in the completion, and the last two are redundant:

$$\frac{\overset{c_0}{x+y\le 1} \qquad \overset{c_1}{-2x+u\le 2}}{2y+u\le 4}\ x$$

$$\frac{\overset{c_0}{x+y\le 1} \qquad \overset{c_2}{-4y-x\le 1}}{-3y\le 2}\ x$$

$$\frac{\overset{c_0}{x+y\le 1} \qquad \overset{c_2}{-4y-x\le 1}}{3x\le 5}\ y$$

$$\frac{\overset{c_0}{x+y\le 1} \qquad \overset{c_3}{-y+z\le 1}}{x+z\le 2}\ y$$

$$\frac{\dfrac{\overset{c_0}{x+y\le 1} \qquad \overset{c_2}{-4y-x\le 1}}{3x\le 5}\ y \qquad \overset{c_1}{-2x+u\le 2}}{3u\le 16}\ x$$

$$\frac{\dfrac{\overset{c_0}{x+y\le 1} \qquad \overset{c_3}{-y+z\le 1}}{x+z\le 2}\ y \qquad \overset{c_1}{-2x+u\le 2}}{2z+u\le 6}\ x$$

$$\frac{\dfrac{\overset{c_0}{x+y\le 1} \qquad \overset{c_3}{-y+z\le 1}}{x+z\le 2}\ y \qquad \overset{c_2}{-4y-x\le 1}}{z-4y\le 3}\ x$$

$$\frac{\dfrac{\overset{c_0}{x+y\le 1} \qquad \overset{c_1}{-2x+u\le 2}}{2y+u\le 4}\ x \qquad \overset{c_2}{-4y-x\le 1}}{2u-x\le 9}\ y$$

$$\frac{\dfrac{\overset{c_0}{x+y\le 1} \qquad \overset{c_1}{-2x+u\le 2}}{2y+u\le 4}\ x \qquad \overset{c_3}{-y+z\le 1}}{u+2z\le 6}\ y$$

Observe that $I \cup \{c_0\}$ augmented with these nine inequalities is closed (and that $I \cup \{c_0\}$ augmented with the first seven is complete). It is interesting to note that the unary inequalities in x and y are redundant in the $\{x, y\}$ projection, yet are not redundant in the closed system.

function $\mathsf{inc\_complete}(I \subseteq \mathsf{TVPI}_X$ where $\mathsf{complete}(I) = I$, $I = I' \cup B$, $c_0 \in \mathsf{TVPI}_X)$
(1) $A := \{c_0\}$
(2) for each $c_1 \in I$
(3) $\quad C_0' := \mathsf{result}(\{c_0\}, \{c_1\})$
(4) $\quad A := A \cup C_0'$
(5) $\quad$ for each $c_2 \in I$
(6) $\quad\quad C_0'' := \mathsf{result}(C_0', \{c_2\})$
(7) $\quad\quad A := A \cup C_0''$
(8) return $\mathsf{filter}(A, I', B)$

Figure 7: Incremental Completion

5. Algorithm for Incremental Closure

The previous section shows that when a new inequality is added to a complete (hence closed) system of TVPI inequalities, every new inequality in the completion of the augmented system can be generated using at most two instances of the result operation (except for false). The augmented system of inequalities might well contain redundancy, or be inconsistent. To maintain a complete representation inconsistency needs to be detected, and redundancies removed. This section gives an algorithm that shows how the system of inequalities might be maintained so that inconsistency is detected and redundancy removed, hence the resulting incrementally augmented system is complete and closed. It further gives the complexity of incremental completion, and relates the result presented to other weakly relational domains.

The outline of the algorithm is as follows. A complete (hence closed) system is a set of collections of ordered inequalities, one for each two variable projection. When a new inequality is added to the system, a set of new inequalities is generated as in Theorem 2. Each of these new inequalities is then added to the previously closed system. This involves finding the projection or projections to which it must be added, inserting it so that updated inequalities are correctly ordered and removing any inequalities that are now redundant. The three components of the algorithm are presented in Figures 7 and 8 and are discussed from the inside out, starting with the update of a two variable projection with insert, then how this is used to filter the augmented system with filter, and finishing with incremental completion, inc_complete, as given in Figure 7.

The input complete system consists of a pair of mappings, one from $X \times X \to \mathsf{TVPI}_X$ and another from $X \to \mathsf{TVPI}_X$, where the first maps a pair of variables to a two variable projection, and the second maps a variable to its bounds. It should also be noted that the arithmetic for the indices of the elements of I is modulo ℓ, i.e. c_ℓ is c_0 and c_{-1} is $c_{\ell-1}$. When performing the complexity analysis of the algorithms in Figures 7 and 8 it is assumed that I_c (in insert) is represented as an AVL-tree, hence insertion, deletion, next, previous, split and join are all $O(\lg(|I_c|))$ [4].

```
function filter(A ⊆ TVPI_X, I ⊆ TVPI_X, B ⊆ TVPI_X)
for each c ∈ reduce(A)
    P' = find_projections(vars(c), I)
    for each P ∈ P'
        I' = insert(c, P)
        if I' = false
            return false
        else
            B' = extract_bounds(I')
            B = update_bounds(B,B')
            I = (I \ P) ∪ I'
        end if
return (I, B)

function insert(c ∈ TVPI_X, I_c ⊆ TVPI_X)
Let I_c = ⟨c_0, ..., c_{ℓ-1}⟩
if I_c = ⟨⟩
    return {c}
else
    i = find_position(c, I_c)
    if {c_{i-1}, c_i} ⊨ c
        return I_c
    else if I_c = ⟨c_0⟩
        if c ⊨ c_0
            D = {c_0}
        else if {c_0, c} ≡ false
            return false
        else
            D = ∅
        end if
    else
        m = min{ℓ > k ≥ 1 | {c_{i-k-1}, c} ⊭ c_{i-k}}
        n = min{ℓ ≥ k ≥ 0 | {c, c_{i+k+1}} ⊭ c_{i+k}}
        if {c_{i-m}, c, c_{i+n}} ≡ false
            return false
        end if
        D = {c_{i+k} | 1 - m ≤ k ≤ n - 1}
    end if
    return (I_c \ D) ∪ {c}
end if
```

Figure 8: Filtering

5.1. Insertion

The function insert in Figure 8 inserts a new inequality into a non-redundant two variable projection. If that two variable projection is unconstrained, then the set containing the new inequality is returned on line (2). The inequalities of the projection are ordered by angle (the actual 0 angle is irrelevant) and line (4) finds the least i such that if c were inserted into I_c it would become c_i shuffling c_j one position along to c_{j+1} for each $i \leq j \leq \ell - 1$. Line (5) tests whether the new inequality is redundant and if so it is discarded. Suppose that c_j is redundant and c_j lies between c_{j-1} and c (symmetrically, c_j lies between c and c_{j+1}), then observe that $\{c_{j-1}, c\} \models c_j$ (and symmetrically $\{c, c_{j+1}\} \models c_j$). By convexity the redundant inequalities must be contiguous. In the case that c is non-redundant, a set of redundant inequalities $D \subseteq I_c$ is constructed. Lines (16) and (17) define offsets m and n so that set of inequalities $D = \{c_{i-m+1}, ..., c_{i+n-1}\}$ defined on line (21) is redundant with the addition of the new inequality. Note that m and n are well-defined because I_c has two or more inequalities (on lines (16), (17)), hence at least one of these must satisfy the disentailment condition. Having found the position that c should be inserted into the ordered I_c, line (18) considers c together with its neighbours (the two neighbours could coincide); if this system is inconsistent, then $I_c \cup \{c\}$ is also inconsistent and false is returned. Lines (8)-(14) address the case where I_c consists solely of a single inequality, c_0. If $c \models c_0$ then line (9) sets $D = \{c_0\}$, else if c_0, c are unsatisfiable together then false is return on line (11), otherwise, no inequality should be deleted from I_c. Line (23) constructs and returns the irredundant set of inequalities that results.

Example 16. Consider the two variable projection defined by $\langle c_0, c_1, c_2, c_3, c_4 \rangle$ in case a) of Figure 9. Suppose that inequality c is added to this, as in case b). On line (4) of insert, find_position will set $i = 2$. On lines (16) and (17) of insert, indices $m = 2$ and $n = 1$ are set. In the first case $\{c_{2-2-1}, c\} \not\models c_{2-2}$ (that is, where $k = 2$) and also $\{c_0, c\} \models c_1$. In the second case $\{c, c_{2+1+1}\} \not\models c_{2+1}$ and also $\{c_3, c\} \models c_2$. This describes the situation where c_1, c_2 are redundant as on line (21), so the updated system is $\langle c_0, c, c_3, c_4 \rangle$, described by the shaded red area in case b) of Figure 9.

Now consider the unbounded two variable projection defined by $\langle c_0, c_1, c_2, c_3, c_4 \rangle$ in case a) of Figure 10. Suppose that inequality c is added to this, as in case b). On line (4) of insert, find_position will set $i = 0$ (hence $i - 1 = 4$). On lines (16) and (17) of insert, indices $m = 5$ and $n = 0$ are set. In the first case $\{c_{0-5-1}, c\} \not\models c_{0-5}$ (that is, where $k = 5$) and also $\{c_0, c\} \models c_1$. In the second case $\{c, c_{0+0+1}\} \not\models c_{0+0}$, that is when $k = 0$, its minimum possible value. This describes the situation where c_4, c_3, c_2, c_1 are redundant as on line (21), so the updated system is $\langle c_0, c \rangle$, described by the shaded red area in case b) of Figure 10.

Consider the complexity of a call to insert. Where A is the set of newly generated inequalities, and I the input system of inequalities (minus unary bounds that do not further constrain the system), note that $|I_c| \leq |A| + |I|$. The function find_position has cost $O(\lg(|I_c|))$. The lookup for c_i, c_{i+1} costs $O(\lg(|I_c|))$, and the entailment checks are constant time. The cases for $|I_c| = 1$ are all constant time operations. In the definition of m, n on lines (16) and (17),

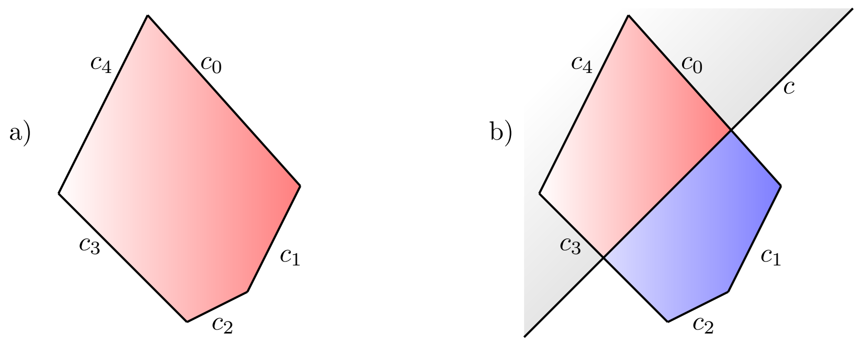


Figure 9: Insertion and filtering

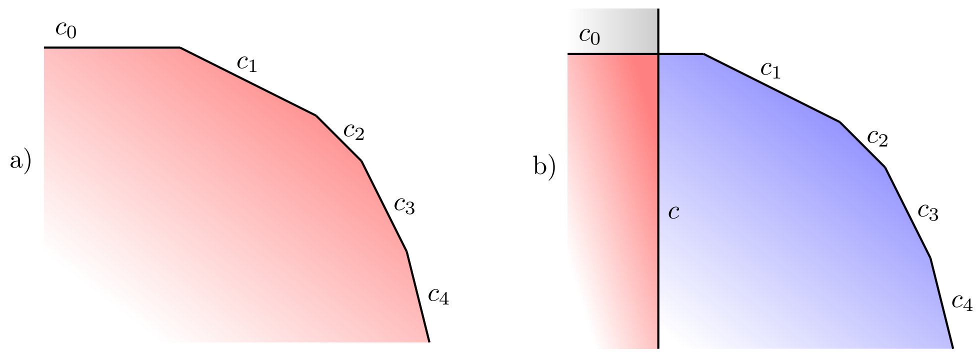


Figure 10: Insertion and filtering for an unbounded two variable projection

each entailment check requires a single lookup with cost $O(\lg(|I_c|))$. Hence the overall cost of defining m, n is $O(d \lg(|I_c|))$, where $d = |D|$, is the number of inequalities to be deleted. The construction of the updated and returned I_c on lines (21) and (23) requires two split operations and a join operation, which is again $O(\lg(|I_c|))$. Hence a single call to insert has cost $O(d \lg(|I_c|) + \lg(|I_c|))$. That is, $O(d \lg(|I_c|))$.

5.2. Filtering

The function filter in Figure 8 controls the addition of a newly generated set of two variable inequalities to a complete system represented as a pair of systems, where I represents the two variable projections and B represents the one variable projections. On line (1), reduce ensures that the set of new inequalities considered does not contain any pair of unary inequalities where one entails the other. Each new inequality is considered one at a time.

On line (2), find_projections gives the two variable projections on which the new inequality might impact. The set of projections found is a singleton if $|\mathsf{vars}(c)| = 2$, and a set of size $|X| - 1$ if $|\mathsf{vars}(c)| = 1$. Hence, the number of calls to insert on line (4) of filter is bounded by $|A|+|X|^2$ (where A is the set of newly generated inequalities). After the call to insert, if the resulting system is inconsistent false is returned, otherwise the TVPI system is updated. First on line (8), extract_bounds finds all bounds implied by the updated projection I' and on line (9) the one variable projections are tightened accordingly. Second on line (10), the two variable projection considered is updated to the new space described by I'. After considering each new inequality the updated complete system is returned on line (12).

Now consider the complexity of the operations involved in filter. The cost of find_projections is $O(|X|\lg(|X|))$, that of extract_bounds $O(|I|+|A|)$, that of update_bounds $O(|X|)$ and of updating I $O(\lg(|X|))$, hence the dominating cost comes from insert. Notice that within a two variable projection each inequality occurrence can only be deleted once. Note $|P|$ is bounded by $|A|+|I|+4$. Hence summing the cost of each insertion gives an overall of cost of

$$\begin{aligned}\sum_{i=1}^{(|X|-1)^2} |d|\lg(|A|+|I|+4) &= \lg(|A|+|I|+4)\sum_{i=1}^{n^2}|d| \\ &= \lg(|A|+|I|+4)(|A|+4(|X|-1)^2)\end{aligned}$$

Hence complexity is $O((|A|+|I|+|X|^2)\lg(|A|+|I|)$.

5.3. Incremental Completion

In Figure 7 the function inc_complete controls the process. It is invoked with the first argument I being a consistent set of inequalities, the union of binary inequalities I' and unary inequalities B. The set A of new inequalities is initialised with the incrementally added inequality c_0 on line (1). Lines (2)-(7) describe the generation of new inequalities using one or two applications of the result operation (as in Theorem 2), before passing these to filter on line (8). This operation has complexity $O(|I|^2)$, where $|A| \leq |I|^2$. The number of inequalities in A is quadratic in the number of inequalities in I and it is expected that $|A|$ is the dominating term and the complexity may be described as $O(|A|\lg(|A|))$. The incremental completion operation requires a quadratic number of applications of result, which when TVPI inequalies are represented using integers is a strongly polynomial operation. Hence a single application of incremental completion is strongly polynomial. As noted in Section 6, the number of successive applications of incremental completion is typically small.

5.4. Restrictions to Octagons and Logahedra

The work is this paper is aimed at the TVPI class of linear inequalities. As discussed earlier, Octagons and Logahedra are subclasses of TVPI. Working with Octagons as the representation, each I_c has at most 8 inequalities. Therefore the cost of each of the AVL operations insertion, deletion, next, previous, split and join is constant time. Redoing the complexity analysis above leads to the conclusion that the complexity of incremental completion for Octagons is $O(|A|)$. Again since each projection has at most 8 inequalities $|A| \leq 4|X|(|X|-1)$, that is the is complexity $O(|X|^2)$. This is the same as the complexity of incremental completion given by [33]. Similarly, any bounded Logahedral representation ensures that I_c has at most a fixed number of inequalities, hence as for Octagons, incremental completion has complexity $O(|X|^2)$.

The represention of systems of inequalities in this paper is compact, in the sense that the only inequalities represented are those needed to support a closed system; redundant inequalities are not stored. This is interesting, since it says that Octagons and Logahedra can be implemented using a compact representation, rather than using matrices (where entries in the matrix may represent a redundant inequality), without a penalty in terms of complexity. It has been noted [2] that Johnson's all pairs shortest path algorithm [24] can be used to calculate closure when Octagons are represented as a graph. However, observe that the output corresponds to the entire matrix representation (which is not compact in the terminology used in this paper), even though the input was a graph.

6. Experimental Results

The performance of TVPI is predicated on how TVPI systems grow with incremental completion. To assess this growth, incremental completion has been implemented, and experiments performed to investigate the size of the resulting closed systems after a number of inequalities have been incrementally added. Calling incremental completion repeatedly is the worst-case for TVPI since the system can grow on each invocation, and the size of representation impacts on both the memory footprint and the running time. Recall that, as discussed in Section 2.4, in the context of abstract interpretation, other operations (join and projection) for TVPI systems rest on closure.

6.1. Implementation

Incremental completion has been implemented[1] in Java 8 making use the BigInteger class for arbitrary precision integer arithmetic. Recalling that a linear inequality over the rationals is a representative of an equivalence class of inequalities, the implementation works with a representation where all coefficients are integral – inequalities with rational coefficients and constants can be rewritten, using a suitable integer multiplier, hence integer arithmetic is sufficient.

Although the insert algorithm is neutral regarding the particular angular ordering, the implementation assigns an angle to an inequality $ax + by \leq c$, where $a \neq 0$ or $b \neq 0$, which is the angle through which the inequality $x \leq 0$ has to be rotated anti-clockwise so its half-space coincides with that of $ax + by \leq 0$. Following this convention, the relative order of two inequalities $a_1x + b_1y \leq e_1$

[1] The code is available at `https://www.cs.kent.ac.uk/people/staff/amk/tvpi.zip` and includes a program `VisualiseInsert` which graphically illustrates the irredundant inequalities found by the insert (filtering) algorithm.

and $a_2x + b_2y \le e_2$ can then be calculated without recourse to trigonometric operations as follows:

$$\mathsf{compare}(a_1x+b_1y \le e_1, a_2x+b_2y \le e_2) = \begin{cases} -1 & \text{if } \mathsf{class}(a_1,b_1) < \mathsf{class}(a_2,b_2) \\ 1 & \text{else if } \mathsf{class}(a_1,b_1) > \mathsf{class}(a_2,b_2) \\ -1 & \text{else if } a_2b_1 < a_1b_2 \\ 1 & \text{else if } a_2b_1 > a_1b_2 \\ 0 & \text{otherwise} \end{cases}$$

where

$$\mathsf{class}(a,b) = \begin{cases} 0 & \text{if } a > 0 \wedge b = 0 \\ 1 & \text{else if } a > 0 \wedge b > 0 \\ 2 & \text{else if } a = 0 \wedge b > 0 \\ 3 & \text{else if } a < 0 \wedge b > 0 \\ 4 & \text{else if } a < 0 \wedge b = 0 \\ 5 & \text{else if } a < 0 \wedge b < 0 \\ 6 & \text{else if } a = 0 \wedge b < 0 \\ 7 & \text{otherwise} \end{cases}$$

6.2. Experimental setup

To exercise incremental completion, TVPI systems of fixed dimension d were randomly generated where the coefficients and constants of each inequality were integers randomly drawn from the ranges [-16, 15] and [0, 31] respectively (enlarging these ranges makes little difference to the results), and pairs of variables were randomly selected from $\{x_0, \ldots, x_{d-1}\}$. Non-negative constants ensured that each inequality was satisfied by the origin, hence each randomly generated TVPI system was satisfiable as a whole, so as to avoid the system collapsing to false (which can be represented in constant space). The Apron [23] implementation of Octagons applies symbolic reasoning to reduce an arbitrary constraint into a system of simpler linear inequalities which are then added to an octagon one-by-one by applying incremental closure [33]. Experiments with the abstract interpretation plugin for Frama-C, EVA [5], instantiated with Apron, suggest that incremental closure is rarely called more than 8 times back-to-back. Therefore, for a given fixed dimension, either 2, 4, 8, or 16, a TVPI system with 8 inequalities were randomly generated. Incremental completion was then applied 7 times to add each inequality, in turn, so as to derive a closed system. The size of resulting output (closed) system was then recorded for 4096 randomly generated (input) systems of 8 inequalities. The same experiment was repeated for input systems of 12, 16, . . . , 32 random inequalities. Figure 11 records how the size of the closed system depends on the dimension and the number of input inequalities. The pair of graphs in the first row of Figure 11 corresponds to dimension 2, the second row to dimension 4, etc. The left hand graphs record how often the output systems are of a given number of inequalities (size). The right hand graphs present a different perspective on this frequency information, recording the proportion of the 4096 input systems whose output does not exceed a given size. Figure 12 repeats the frequency experiments for Logahedra and Octagons where the coefficients are randomly drawn from $\{-2, -1, 0, 1, 2\}$ and $\{-1, 0, 1\}$ respectively and the constants are integers again drawn from [0, 31].

6.3. Results

The first row of Figure 11 suggests that for the planar case the number of inequalities in the output is typically around 6, irrespective of the size of the input. For 4 dimensional systems, there is a divergence in output size with input size, but the output size has a stable profile for 16 or more input inequalities. A similar phenomenon occurs for dimension 8 at 24 or more input inequalities. The cumulative distribution graphs of Figure 11 are annotated with the 95th percentile line, showing that 95% of the time the output was no more than 3, 4 and 12 times the size of the input for the dimensions 4, 8 and 16 respectively. Thus growth is dependent on dimension.

The incremental completion algorithm is applicable to both Logahedra, where the coefficients are powers, and Octagons, since these are instances of TVPI systems. To aid growth comparisons, the ranges of Figure 12, which gives the size frequency distributions for Logahedra and Octagons, coincide with those of Figure 11. Compared to general TVPI, the frequency distributions for Logahedra are shifted to the left, and spike higher, a pattern which is accentuated further for Octagons. The output is thus smaller, which can be explained because of the likelihood of one two variable inequality being entailed by another. On the other hand, the growth rate for TVPI is no worse than 2 or 3 fold that of Octagons.

7. Conclusion

This paper has shown how a key operation for manipulating the TVPI class of inequalities — incremental closure — can been derived by a systematic examination of the structure of derivation trees that arise when a new inequality is added to a closed system. By studying derivation trees an algorithm for incremental closure is synthesised which sits on a firm theoretical foundation. An experimental evalution studies the growth of systems of inequalities built by successive applications of incremental completion.

The algorithm manipulates systems of inequalities which are compact, that is, the only inequalities represented are those needed to support a closed system. This chimes with the desire to derive memory efficient decision procedures [16].

The presented algorithm is incremental. Although motivated by the design of abstract domains, incrementality is a key attribute for any theory deployed in an SMT solver. For the target application in abstract interpretation, it is sufficient to work with non-strict inequalities only. The extension of the results in this work to additionally allow strict inequalities is straightforward. Hence the techniques in this paper are suitable for incorporating with the theory component of an SMT solver. The extension of the algorithms presented to maintain a certificate which tracks how inequalities are derived, and in particular when inconsistency is detected, is also easy to achieve.

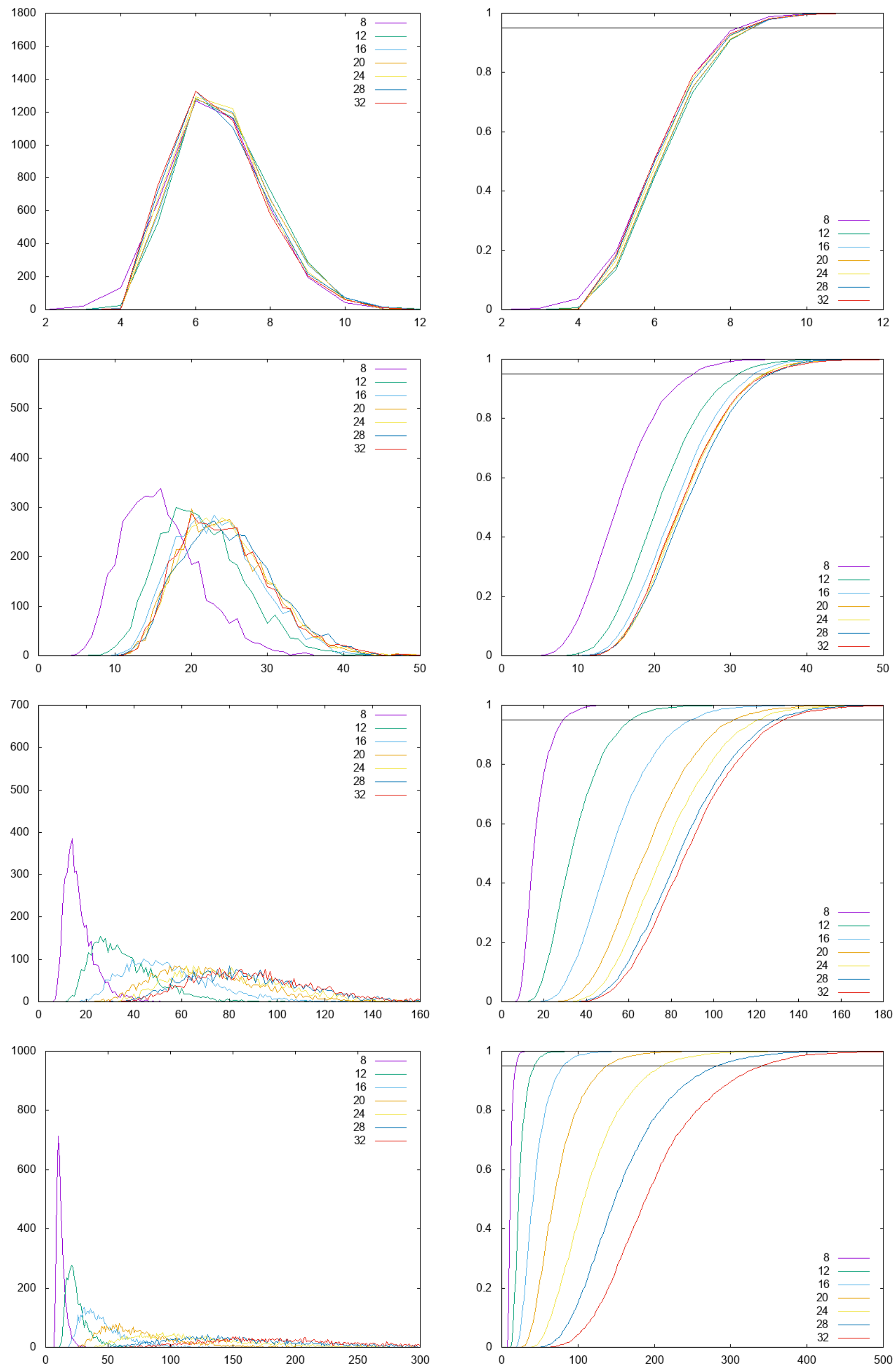


Figure 11: Frequency and cumulative distributions for 2-, 4-, 8- and 16-dimensional completion

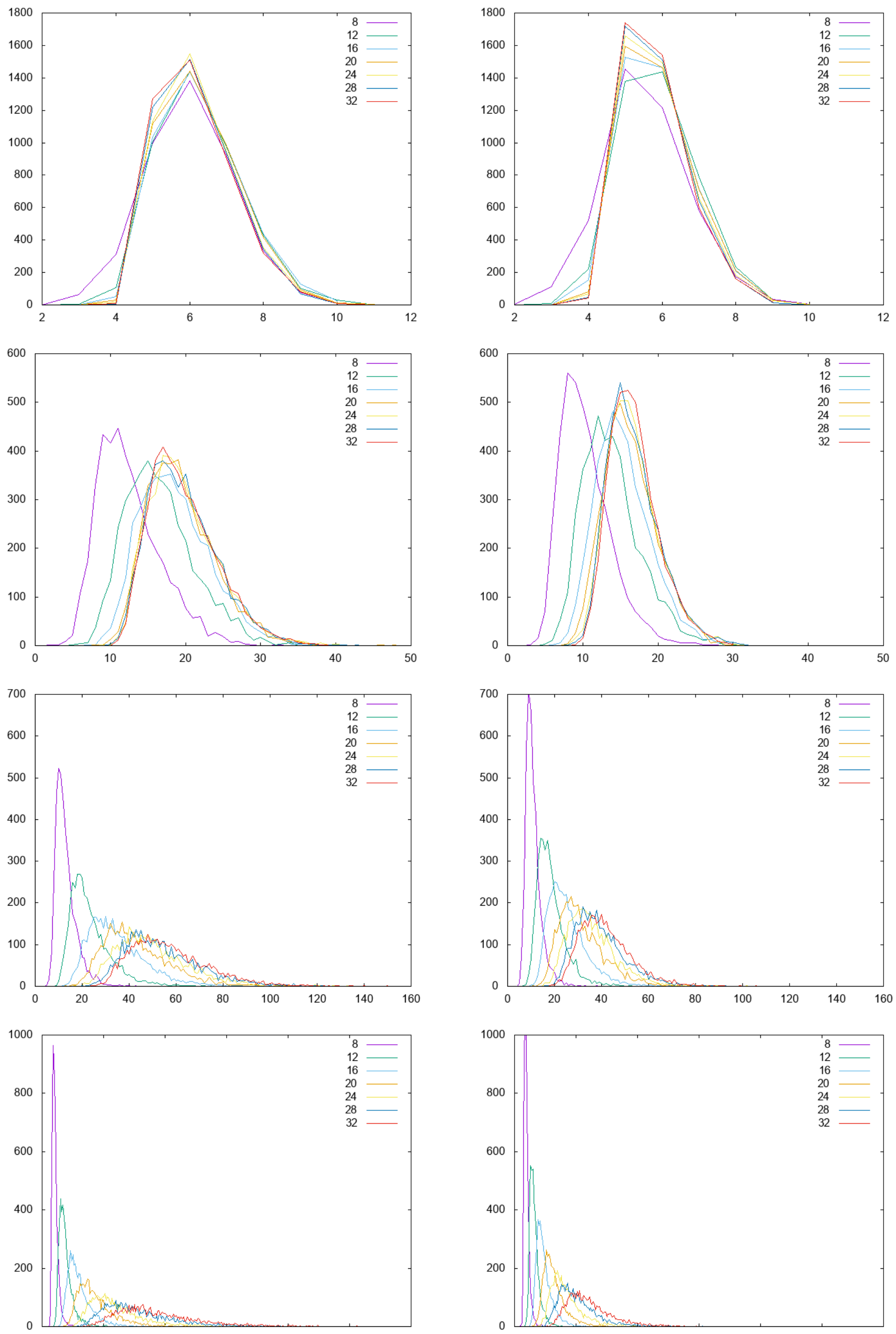


Figure 12: Frequency distributions for Logahedra (left) and Octagons (right)

As well as providing a systematic construction of the incremental closure algorithm, the analysis of the derivations in the calculus defined by the result operation provides a scaffold on which to build other algorithms, such as those maintaining certificates, for linear inequalities.

Acknowledgements

Andy King was supported by EPSRC grants EP/K031929/1 and EP/N020243/1. The authors would like to thank the anonymous referees for their insightful comments, Jason Yau for discussion of the implementation, and Aziem Chawdhary for his experimental work on EVA.

References

[1] B. Aspvall and Y. Shiloach. A Polynomial Time Algorithm for Solving Systems of Linear Inequalities with Two Variables per Inequality. *SIAM Journal of Computing*, 9(4):827–845, 1980.

[2] R. Bagnara, P. M. Hill, and E. Zaffanella. Weakly-relational Shapes for Numeric Abstractions: Improved Algorithms and Proofs of Correctness. *Formal Methods in System Design*, 35(3):279–323, 2009.

[3] R. Bellman. On A Routing Problem. *Quarterly of Applied Mathematics*, 16(1):87–90, 1958.

[4] G. E. Blelloch, D. Ferizovic, and Y. Sun. Just Join for Parallel Ordered Sets. In *Symposium on Parallelism in Algorithms and Architectures*, pages 253–264. ACM Press, 2016.

[5] D. Bühler, P. Cuoq, B. Yakobowski, M. Lemerre, A. Maroneze, V. Perrelle, and V. Prevosto. *The EVA Plug-in*. CEA LIST, Software Reliability Laboratory, Saclay, France, F-91191, 2017. `https://frama-c.com/download/frama-c-value-analysis.pdf`.

[6] R. Clarisó and J. Cortadella. The Octahedron Abstract Domain. *Science of Computer Programming*, 64(1):115–139, 2007.

[7] E. Cohen and N. Megiddo. Improved Algorithms for Linear Inequalities with Two Variables per Inequality. *SIAM Journal of Computing*, 23(6):1313–1347, 1994.

[8] P. Cousot and R. Cousot. Static Determination of Dynamic Properties of Programs. In *Proceedings of the Second International Symposium on Programming*, pages 106–130, 1976.

[9] P. Cousot and N. Halbwachs. Automatic Discovery of Linear Restraints among Variables of a Program. In *Principles of Programming Languages*, pages 84–97. ACM Press, 1978.

[10] A. Darte, Y. Robert, and F. Vivien. *Scheduling and Automatic Parallelization*. Birkhaüser, 2000.

[11] D. Dill. Timing Assumptions and Verification of Finite-State Concurrent Systems. In *Computer Aided Verification*, volume 407 of *Lecture Notes in Computer Science*, pages 187–212. Springer, 1989.

[12] P. Feautrier. Dataflow Analysis of Array and Scalar References. *International Journal of Parallel Programming*, 20:23–53, 1991.

[13] P. Feautrier. Some Efficient Solutions to the Affine Scheduling Problem. I. One-Dimensional Time. *International Journal of Parallel Programming*, 21:313–348, 1992.

[14] R. W. Floyd. Algorithm 97: Shortest Path. *Communications of the ACM*, 5(6):345, 1962.

[15] J. B. J. Fourier. *Oeuvres de Fourier*, chapter Second Extrait, pages 325–328. Gauthier-Villars et Fils, 1890. Reprint from Histoire de l'Academie 1824.

[16] G. Gange, J. A. Navas, P. Schachte, H. Søndergaard, and P. J. Stuckey. Exploiting Sparcity in Difference Bound Matrices. In *Static Analysis Symposium*, volume 9837 of *Lecture Notes in Computer Science*, pages 189–211. Springer, 2016.

[17] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[18] K. Ghorbal, E. Goubault, and S. Putot. The Zonotope Abstract Domain Taylor1+. In *Computer Aided Verification*, volume 5649 of *Lecture Notes in Computer Science*, pages 627–633. Springer, 2009.

[19] W. Harvey and P. J. Stuckey. A Unit Two Variable per Inequality Integer Constraint Solver for Constraint Logic Programming. *Australian Computer Science Communications*, 19(1):102–111, 1997.

[20] D. S. Hochbaum. Monotonizing Linear Programs with Up to Two Nonzeroes per Column. *Operational Research Letters*, 32(1):49–58, 2004.

[21] D. S. Hochbaum and J. Naor. Simple and Fast Algorithms for Linear and Integer Programs with Two Variables per Inequality. *SIAM Journal of Computing*, 23(6):1179–1192, 1994.

[22] J. M. Howe and A. King. Logahedra: a New Weakly Relational Domain. In *International Symposium on Automated Technology for Verification and Analysis*, volume 5799 of *Lecture Notes in Computer Science*, pages 306–320. Springer, 2009.

[23] B. Jeannet and A. Miné. APRON: A library of numerical abstract domains for static analysis. In *CAV*, volume 5643 of *Lecture Notes in Computer Science*, pages 661–667, 2009.

[24] D. B. Johnson. Efficient algorithms for shortest paths in sparse networks. *Journal of the ACM*, 24(1):1–13, 1977.

[25] R. M. Karp, R. E. Miller, and S. Winograd. The Organization of Computations for Uniform Recurrence Equations. *Journal of the ACM*, 14:563–590, 1967.

[26] J. C. Lagarias. The Computational Complexity of Simultaneous Diophantine Approximation Problems. *SIAM Journal of Computing*, 14:196–209, 1985.

[27] S. K. Lahiri and M. Musuvathi. An Efficient Decision Procedure for UTVPI Constraints. In *Frontiers on Combining Systems*, volume 3717 of *Lecture Notes in Computer Science*, pages 168–183. Springer, 2005.

[28] K. G. Larsen, F. Larsson, P. Pettersson, and W. Yi. Efficient Verification of Real-time Systems: Compact Data Structure and State-space Reduction. In *IEEE Real-Time Systems Symposium*, pages 14–24. IEEE Computer Society, 1997.

[29] V. Laviron and F. Logozzo. SubPolyhedra: a Family of Numerical Abstract Domains for the (More) Scalable Inference of Linear Inequalities. *International Journal on Software Tools for Technology Transfer*, 13(6):585–601, 2011.

[30] F. Logozzo and M. Fähndrich. Pentagons: A Weakly Relational Abstract Domain for the Efficient Validation of Array Accesses. *Science of Computer Programming*, 75(9):796–807, 2010.

[31] M. Measche and B. Berthomieu. Time Petri-nets for Analyzing and Verifying Time Dependent Communication Protocols. In *Protocol Specification, Testing and Verification III*, pages 161–172. North-Holland, 1983.

[32] A. Miné. A New Numerical Abstract Domain Based on Difference-Bound Matrices. In *Programs as Data Objects*, volume 2053 of *Lecture Notes in Computer Science*, pages 155–172. Springer, 2001.

[33] A. Miné. The Octagon Abstract Domain. *Higher-Order and Symbolic Computation*, 19(1):31–100, 2006.

[34] C. G. Nelson. An $n^{\lg n}$ Algorithm for the Two-Variable-Per-Constraint Linear Programming Satisfiability Problem. Technical Report STAN-CS-78-689, Stanford University, Computer Science Department, 1978.

[35] V. R. Pratt. Two Easy Theories Whose Combination is Hard. `http://boole.stanford.edu/pub/sefnp.pdf`, 1977.

[36] F. Quilleré, S. Rajopadhye, and D. Wilde. Generation of Efficient Nested Loops from Polyhedra. *International Journal of Parallel Programming*, 28:469–498, 2000.

[37] S. Sankaranarayanan, H. B. Sipma, and Z. Manna. Constraint Based Linear Relations Analysis. In *Static Analysis Symposium*, volume 3148 of *Lecture Notes in Computer Science*, pages 53–68. Springer, 2004.

[38] A. Schutt and P. J. Stuckey. Incremental Satisfiability and Implication for UTVPI Constraints. *INFORMS Journal on Computing*, 22(4):514–527, 2010.

[39] A. Simon and A. King. Taming the Wrapping of Integer Arithmetic. In *Static Analysis Symposium*, volume 4634 of *Lecture Notes in Computer Science*, pages 121–136. Springer, 2007.

[40] A. Simon, A. King, and J. M. Howe. Two Variables per Linear Inequality as an Abstract Domain. In *Logic Based Program Development and Transformation*, volume 2664 of *Lecture Notes in Computer Science*, pages 71–89. Springer, 2002.

[41] W. P. Thurston. On Proof And Progress in Mathematics. *Bulletin of the American Mathematical Society*, 30(2):167–177, 1994.

[42] R. Upadrasta and A. Cohen. Sub-polyhedral scheduling using (unit-)two-variable-per-inequality polyhedra. In *Principles of Programming Langauges*, pages 483–496. ACM Press, 2013.

[43] S. Warshall. A Theorem on Boolean Matrices. *Journal of the ACM*, 9(1):11–12, 1962.

[44] K. D. Wayne. A Polynomial Combinatorial Algorithm for Generalized Minimum Cost Flow. In *Symposium on Theory of Computing*, pages 11–18. ACM Press, 1999.

A. Appendices

This section contains the full detailed proofs of results with outline proofs in the main body of the paper.

Lemma 6 (Redundancy). Let $I \subseteq \mathsf{TVPI}_X$ and let $c, c_0 \in \mathsf{TVPI}_X$ where $x \in \mathsf{vars}(c)$, $x \in \mathsf{vars}(c_0)$. Suppose $c \notin \mathsf{complete}(I)$ and $I \models c$. Then $\mathsf{complete}(I \cup \{c_0\}) \models \mathsf{result}(c_0, c, x)$ and $\mathsf{result}(c_0, c, x) \notin \mathsf{complete}(I \cup \{c_0\})$.

Proof 6. Without loss of generality there are two cases to consider:

i) Where $|\mathsf{vars}(c)| = 1$.
ii) Where $|\mathsf{vars}(c)| = 2$.

i) By the definition of closed there exists $c_1 \in \mathsf{complete}(I)$, $c_1 \not\equiv c$ such that $\{c_1\} \models c$. Suppose $c \equiv ax \leq e$ and $c_0 \equiv a_0x + b_0y \leq e_0$ (where it might be that $b_0 = 0$). Hence $\mathsf{result}(c_0, c, x) = |a|b_0y \leq |a|e_0 + |a_0|e$. Then put $c_1 \equiv ax \leq e - \delta$, $\delta > 0$ and consider $c_1' = \mathsf{result}(c_0, c_1, x) = |a|b_0y \leq |a|e_0 + |a_0|e - |a_0|\delta$. Observe that $c_1' \models \mathsf{result}(c_0, c, x)$ and the result holds.

ii) Suppose $\mathsf{vars}(c) = \{x, z\}$ and $\mathsf{vars}(c_0) \subseteq \{x, y\}$. Consider $c \equiv ax + bz \leq e$, where $I \models c$, $c_0 \equiv a_0x + b_0y \leq e_0$ (where it might be that $b_0 = 0$) and $c' \equiv \mathsf{result}(c_0, c, x)$. Then (since $I \models c$ and noting Theorem 1) one of the two cases holds:

(a) There exists $c_1 \in \mathsf{complete}(I)$ such that $c_1 \equiv ax + bz \leq e - \delta$, for some $\delta > 0$ (that is, $\{c_1\} \models c$)

(b) There exists $c_1, c_2 \in \mathsf{complete}(I)$, $\mathsf{vars}(\{c_1, c_2\}) = \mathsf{vars}(c)$ such that:

$$\frac{c_1 \quad c_2}{c} +_{(\lambda_1, \lambda_2, \delta)}$$

In each case, considering the resultant of c_0 with c_1 and/or c_2 will lead to a demonstration of the redundancy of c'.

(a) $c' = \mathsf{result}(c_0, c, x)$ is derived as follows:

$$\frac{a_0x + b_0y \leq e_0 \quad ax + bz \leq e}{|a|b_0y + |a_0|bz \leq |a|e_0 + |a_0|e} \; x$$

Since $\{c_1\} \models c$ observe that $c_1' = \mathsf{result}(c_0, c_1, x)$ is derived:

$$\frac{a_0x + b_0y \leq e_0 \quad ax + bz \leq e - \delta}{|a|b_0y + |a_0|bz \leq |a|e_0 + |a_0|(e - \delta)} \; x$$

Hence $\{c_1'\} \models c'$, that is $\mathsf{complete}(I \cup \{c_0\}) \models c'$.

(b) $c' = \mathsf{result}(c_0, c, x)$ is derived as follows, where c is given by:

$$\frac{a_1x + b_1z \leq e_1 \quad a_2x + b_2z \leq e_2}{(\lambda_1a_1 + \lambda_2a_2)x + (\lambda_1b_1 + \lambda_2b_2)z \leq \lambda_1e_1 + \lambda_2e_2 + \delta} +_{(\lambda_1, \lambda_2, \delta)}$$

and

$$\frac{a_0x + b_0y \leq e_0 \quad c}{c'} \; x$$

with

$$c' = \begin{array}{l} |\lambda_1a_1 + \lambda_2a_2|b_0y + |a_0|(\lambda_1b_1 + \lambda_2b_2)z \\ \quad \leq |\lambda_1a_1 + \lambda_2a_2|e_0 + |a_0|(\lambda_1e_1 + \lambda_2e_2) + |a_0|\delta \end{array}$$

Note that $a = \lambda_1a_1 + \lambda_2a_2$, $b = \lambda_1b_1 + \lambda_2b_2$, $e = \lambda_1e_1 + \lambda_2e_2 + \delta$. Observe from the resultant step that $a_0(\lambda_1a_1 + \lambda_2a_2) < 0$. There are two cases to consider:

i. $a_1 > 0, a_2 \geq 0$ (symmetrically, $a_1 < 0, a_2 \leq 0$)

ii. $a_1 > 0, a_2 < 0$ (symmetrically, $a_1 < 0, a_2 > 0$)

To demonstrate that in each case c' is redundant:

i. Consider $c'_1 = \mathsf{result}(c_0, c_1, x)$ and $c'_2 = \mathsf{result}(c_0, c_2, x)$:

$$\frac{a_0 x + b_0 y \le e_0 \quad a_1 x + b_1 z \le e_1}{|a_1| b_0 y + |a_0| b_1 z \le |a_1| e_0 + |a_0| e_1}\ x$$

and

$$\frac{a_0 x + b_0 y \le e_0 \quad a_2 x + b_2 z \le e_2}{|a_2| b_0 y + |a_0| b_2 z \le |a_2| e_0 + |a_0| e_2}\ x$$

It will be demonstrated that:

$$\frac{c'_1 \quad c'_2}{c'}\ +_{(\lambda_1, \lambda_2, |a_0|\delta)}$$

Observe that, scaling the coefficients of c'_1 by $\lambda_1 \ge 0$ and of c'_2 by $\lambda_2 \ge 0$:

$$\begin{aligned} \lambda_1 |a_1| b_0 + \lambda_2 |a_2| b_0 &= |\lambda_1 a_1 + \lambda_2 a_2| b_0 \\ \lambda_1 |a_0| b_1 + \lambda_2 |a_0| b_2 &= |a_0| (\lambda_1 b_1 + \lambda_2 b_2) \end{aligned}$$

and also that:

$$\begin{aligned} &\lambda_1 |a_1| e_0 + \lambda_1 |a_0| e_1 + \lambda_2 |a_2| e_0 + \lambda_2 |a_0| e_2 + \delta |a_0| \\ &\qquad = (\lambda_1 |a_1| + \lambda_2 |a_2|) e_0 + \lambda_1 |a_0| e_1 + \lambda_2 |a_0| e_2 + \delta |a_0| \\ &\qquad = |\lambda_1 a_1 + \lambda_2 a_2| e_0 + |a_0| (\lambda_1 e_1 + \lambda_2 e_2) + |a_0| \delta. \end{aligned}$$

hence $\{c'_1, c'_2\} \models c'$ and the result holds, except where $a_2 = 0$. Here, put $c'_2 = |a_0| c_2$ and the result goes through as above.

ii. It will be demonstrated that (when $a_2 < 0$):

$$\frac{\dfrac{c_0 \quad c_1}{c'_1}\ x \qquad \dfrac{c_1 \quad c_2}{c_{12}}\ x}{c'}\ +_{(\lambda_0, \lambda_3, |a_0|\delta)}$$

First consider $c'_1 = \mathsf{result}(c_0, c_1, x)$ (without loss of generality, the conditions imply that $\lambda_1 |a_1| > \lambda_2 |a_2|$)

$$\frac{a_0 x + b_0 y \le e_0 \quad a_1 x + b_1 z \le e_1}{|a_1| b_0 y + |a_0| b_1 z \le |a_1| e_0 + |a_0| e_1}\ x$$

Also note that if $a_1 a_2 < 0$ then the resolvant of c_1 and c_2, $c_{12} = \mathsf{result}(c_1, c_2, x)$, is as follows:

$$\frac{a_1 x + b_1 z \le e_1 \quad a_2 x + b_2 z \le e_2}{|a_2| b_1 z + |a_1| b_2 z \le |a_2| e_1 + |a_1| e_2}\ x$$

Now appropriate values $(\lambda_0, \lambda_3, \delta')$ can be found that will show that c' is redundant. First, putting

$$\lambda_0 = \frac{|a|}{|a_1|} = \frac{|\lambda_1 a_1 + \lambda_2 a_2|}{|a_1|} = \frac{|\lambda_1 a_1| - |\lambda_2 a_2|}{|a_1|} = \lambda_1 - \lambda_2 \frac{|a_2|}{|a_1|}$$

means that $\lambda_0 c'_1$ has the same y coefficient as c'.
Second, putting

$$\lambda_3 = \lambda_2 \frac{|a_0|}{|a_1|}$$

observe that

$$\begin{aligned}
\lambda_0|a_0|b_1 + \lambda_3|a_2|b_1 + \lambda_3|a_1|b_2 &= \lambda_1|a_0|b_1 + \lambda_2|a_0|b_2 - \lambda_2\frac{|a_0||a_2|b_1}{|a_1|} \\
&\qquad + \lambda_2\frac{|a_0||a_2|b_1}{|a_1|} \\
&= |a_0|(\lambda_1 b_1 + \lambda_2 b_2) \\
&= |a_0|b
\end{aligned}$$

Now consider the constant term.

$$\begin{aligned}
&\lambda_0(|a_1|e_0 + |a_0|e_1) + \lambda_3(|a_2|e_1 + |a_1|e_2) \\
&\quad = (\lambda_1 - \lambda_2\frac{|a_2|}{|a_1|})(|a_1|e_0 + |a_0|e_1) + \lambda_2\frac{|a_0|}{|a_1|}(|a_2|e_1 + |a_1|e_2) \\
&\quad = \lambda_1|a_1|e_0 + \lambda_1|a_0|e_1 - \lambda_2\frac{|a_2|}{|a_1|}|a_1|e_0 - \lambda_2\frac{|a_2|}{|a_1|}|a_0|e_1 \\
&\qquad\quad + \lambda_2\frac{|a_0|}{|a_1|}|a_2|e_1 + \lambda_2\frac{|a_0|}{|a_1|}|a_1|e_2 \\
&\quad = \lambda_1|a_1|e_0 + \lambda_1|a_0|e_1 - \lambda_2|a_2|e_0 + \lambda_2|a_0|e_2 \\
&\quad = |\lambda_1 a_1 + \lambda_2 a_2|e_0 + |a_0|(\lambda_1 e_1 + \lambda_2 e_2)
\end{aligned}$$

with last step following since $a_2 < 0$. This shows that $\delta' = |a_0|\delta$.
Hence, $\{c'_1, c_{12}\} \models c'$ and the result follows.

In all cases c' has been shown to be redundant. □

Lemma 7 (Compaction). Let $c_0 \in \mathsf{TVPI}_X$, $c_1, c_2 \in I \subseteq \mathsf{TVPI}_X$, $c_3 \in I \cup \{c_0\}$ and $x_0, x_1, x_2 \in X$, where $\mathsf{complete}(I) = I$. If $c \in \mathsf{complete}(I \cup \{c_0\})$ and

$$c = \mathsf{result}(\mathsf{result}(\mathsf{result}(c_0, c_1, x_0), c_2, x_1), c_3, x_2)$$

and $|\mathsf{vars}(c)| = 2$ then there exists $d_0, d_1 \in I$ and $y_0, y_1 \in X$ such that

$$c = \mathsf{result}(\mathsf{result}(c_0, d_0, y_0), d_1, y_1)$$

Proof 7. The structure of the series of resultants being considered is:

$$\dfrac{\dfrac{\dfrac{c_0 \quad c_1}{c'_1}\, x_0 \qquad c_2}{c'_2}\, x_1 \qquad c_3}{c}\, x_2$$

The proof proceeds by giving a series of reductions demonstrating that by reordering the application of the result operation one of the conclusions of the lemma holds. A reduction will be given for every possible configuration of the initial series of resultants.

There are four possible configurations that lead to different combinations of the variables being eliminated in the sequence of resultant steps. The skeletons of these four cases are given in Figure 3. These are treated with the greatest generality possible. If some of the variables occurring in the sequence of resultant steps are identical these four possible configurations gives rise to a number of subcases that need to be considered.

First, the cases where all variables occurrences are distinct are considered.

1. Where $c_0 \equiv a_0x + b_0y \leq e_0$, $c_1 \equiv a_1x + b_1u \leq e_1$ $c_2 \equiv a_2u + b_2v \leq e_2$ and $c_3 \equiv a_3v + b_3z \leq e_3$, with $x_0 = x, x_1 = u, x_2 = v$ (and the variables x, y, u, v, z are distinct), the derivation gives:

   $$c = |a_1||a_2||a_3|b_0y + |a_0||b_1||b_2|b_3z \leq \\ |a_1||a_2||a_3|e_0 + |a_0||a_2||a_3|e_1 + |a_0||a_3||b_1|e_2 + |a_0||b_1||b_2|e_3$$

   where $a_0a_1 < 0$, $b_1a_2 < 0$ and $b_2a_3 < 0$. The reduction here is to the following:

   $$\cfrac{\cfrac{c_0 \qquad \cfrac{c_1 \quad c_2}{c_{12}}\,u}{c'_{12}}\,x \qquad c_3}{c}\,v$$

   that is $c = \mathsf{result}(\mathsf{result}(c_0, \mathsf{result}(c_1, c_2, u), x), c_3, v)$. Put $d_0 = \mathsf{result}(c_1, c_2, u)$, $d_1 = c_3$, $y_0 = x$ and $y_1 = v$. Since I is complete, either $d_0 \in I$ and the result holds or $d_0 \notin I$ therefore $I \models d_0$, hence by Lemma 6 $\mathsf{result}(c_0, \mathsf{result}(c_1, c_2, u), x) \notin \mathsf{complete}(I \cup \{c_0\})$ thence again by Lemma 6 $c \notin \mathsf{complete}(I \cup \{c_0\})$.
   An alternative reduction, that $c = \mathsf{result}(\mathsf{result}(\mathsf{result}(c_0, c_1, x), c_2, u), c_3, v)$ reduces to $c = \mathsf{result}(\mathsf{result}(c_0, c_1, x), \mathsf{result}(c_2, c_3, v), u)$, is also available:

   $$\cfrac{\cfrac{c_0 \quad c_1}{c'_1}\,x \qquad \cfrac{c_2 \quad c_3}{c_{23}}\,v}{c}\,u$$

   Put $d_0 = c_1 \in I$, $d_1 = \mathsf{result}(c_2, c_3, v)$, $y_0 = x$ and $y_1 = u$. Since I is complete, either $d_1 \in I$ and the result holds or $d_1 \notin I$ and therefore $I \models d_1$, hence by Lemma 6 $c \notin \mathsf{complete}(I \cup \{c_0\})$.
2. Consider the case where c_0, c_1, c_2 are as above and $c_3 \equiv a_3y + b_3z \leq e_3$ and $x_0 = x, x_1 = u, x_2 = y$. In this case $c = \mathsf{result}(\mathsf{result}(\mathsf{result}(c_0, c_1, x), c_2, u), c_3, y)$ reduces to $c = \mathsf{result}(\mathsf{result}(c_0, \mathsf{result}(c_1, c_2, u), x), c_3, y)$. The argument that the result holds is the same as for case 1.
3. Consider the case where c_0, c_1 are as above and $c_2 \equiv a_2y + b_2v \leq e_2$, $c_3 \equiv a_3u + b_3z \leq e_3$ and $x_0 = x, x_1 = y, x_2 = u$. In this case

   $$c = \mathsf{result}(\mathsf{result}(\mathsf{result}(c_0, c_1, x), c_2, y), c_3, u)$$

reduces to

$$c = \mathsf{result}(\mathsf{result}(c_0, \mathsf{result}(c_1, c_3, u), x), c_2, y)$$

Again, the argument that the result holds is as for case 1. As in case 1, an alternative is possible, where the reduction is to

$$c = \mathsf{result}(\mathsf{result}(c_0, c_2, y), \mathsf{result}(c_1, c_3, u))$$

4. Consider the case where c_0, c_1 are as above and $c_2 \equiv a_2y + b_2v \leq e_2$, $c_3 \equiv a_3v + b_3z \leq e_3$ and $x_0 = x, x_1 = y, x_2 = v$. In this case

$$c = \mathsf{result}(\mathsf{result}(\mathsf{result}(c_0, c_1, x), c_2, y), c_3, v)$$

reduces to

$$c = \mathsf{result}(\mathsf{result}(c_0, c_1, x), \mathsf{result}(c_2, c_3, v), y)$$

Here, the argument that the result holds is similar to that for the alternative reduction in case 1.

In each of the above four reductions the variables are assumed to be distinct. When this restriction is lifted there arise a number of subcases to be considered. Note that $|\mathsf{vars}(c)| = 2$, hence any identities which give $|\mathsf{vars}(c)| \neq 2$ are not possible.

1. When the eliminated variables are x, u, v it is possible that $z = x$ or $z = u$ or $v = x$. In the first two cases the reductions go through unchanged. In the third case it should be noted that only the second of the two reductions given goes through. It is also possible that both $v = x$ and $z = u$, and again, the second of the two reductions goes through.
2. Next, consider the case where the eliminated variables are x, u, y.
   (a) Suppose that x and z are the same variable. Then the reduction still holds.
   (b) Suppose that z and u are the same variable. Then the reduction still holds.
   (c) Suppose that x and v are the same variable. Then $c_0 \equiv a_0x + b_0y \leq e_0$, $c_1 \equiv a_1x + b_1u \leq e_1$, $c_2 \equiv a_2u + b_2x \leq e_2$ and $c_3 \equiv a_3y + b_3z \leq e_3$. This means that

$$c = |a_0||a_3||b_1|b_2x + |a_1||a_2||b_0|b_3z \leq \\ |a_1||a_2||a_3|e_0 + |a_0||a_2||a_3|e_1 + |a_0||a_3||b_1|e_2 + |a_1||a_2||b_0|e_3$$

   Now consider the following two eliminations, defining c_3' and c_{12}:

$$\frac{a_0x + b_0y \leq e_0 \quad a_3y + b_3z \leq e_3}{|a_3|a_0x + |b_0|b_3z \leq |a_3|e_0 + |b_0|e_3}\ y$$

   and

$$\frac{a_1x + b_1u \leq e_1 \quad a_2u + b_2x \leq e_2}{|a_2|a_1x + |b_1|b_2x \leq |a_2|e_1 + |b_1|e_2}\ u$$

Since $a_0 a_1 < 0$

$$\frac{c'_3 \quad c_{12}}{c} +_{(|a_1||a_2|, |a_0||a_3|, 0)}$$

If $c_{12} \in \mathsf{complete}(I)$ then c is redundant because it is a linear combination of c'_3 and c_{12}. If $c_{12} \notin \mathsf{complete}(I)$ but $c'_3 \in \mathsf{complete}(I \cup \{c_0\})$ then by Lemma 6 it also follows that c is redundant. Likewise if $c'_3 \notin \mathsf{complete}(I \cup \{c_0\})$. In all cases, the pre-condition of the lemma that $c \in \mathsf{complete}(I \cup \{c_0\})$ is contradicted. This case also goes through when both $x = v$ and $z = u$.

3. Now consider the case where the eliminated variables are x, y, u.
   (a) Suppose that x and z are the same variables, then $c_0 \equiv a_0 x + b_0 y \leq e_0$, $c_1 \equiv a_1 x + b_1 u \leq e_1$, $c_2 \equiv a_2 y + b_2 v \leq e_2$ and $c_3 \equiv a_3 u + b_3 x \leq e_3$. This means that

   $$\begin{aligned} c = {} & |a_1||a_3||b_0|b_2 v + |a_0||a_2||b_1|b_3 x \leq \\ & |a_1||a_2||a_3|e_0 + |a_0||a_2||a_3|e_1 + |a_1||a_3||b_0|e_2 + |a_0||a_2||b_1|e_3 \end{aligned}$$

   Now consider the following two eliminations, defining c'_2 and c_{13}:

   $$\frac{a_0 x + b_0 y \leq e_0 \quad a_2 y + b_2 v \leq e_2}{|a_2|a_0 x + |b_0|b_2 v \leq |a_2|e_0 + |b_0|e_2} \; y$$

   and

   $$\frac{a_1 x + b_1 u \leq e_1 \quad a_3 u + b_3 x \leq e_3}{|a_3|a_1 x + |b_1|b_3 x \leq |a_3|e_1 + |b_1|e_3} \; u$$

   then

   $$\frac{c'_2 \quad c_{13}}{c} +_{(|a_1||a_3|, |a_0||a_2|, 0)}$$

   Hence, c is redundant, contradicting the pre-condition of the lemma that $c \in \mathsf{complete}(I \cup \{c_0\})$.
   (b) Suppose that y and z are the same variable. Then note that only the second of the two reductions given holds.
   (c) Suppose that v and x are the same variable, then the reduction still holds.
   (d) Suppose that x and v, and y and z, are the same variable. Then $c_0 \equiv a_0 x + b_0 y \leq e_0$, $c_1 \equiv a_1 x + b_1 u \leq e_1$, $c_2 \equiv a_2 y + b_2 x \leq e_2$ and $c_3 \equiv a_3 u + b_3 y \leq e_3$. This means that

   $$\begin{aligned} c = {} & |a_1||a_3||b_0|b_2 x + |a_0||a_2||b_1|b_3 y \leq \\ & |a_1||a_2||a_3|e_0 + |a_0||a_2||a_3|e_1 + |a_1||a_3||b_0|e_2 + |a_0||a_2||b_1|e_3 \end{aligned}$$

   Now consider the following two eliminations, defining c'_2 and c_{13}:

   $$\frac{a_0 x + b_0 y \leq e_0 \quad a_2 y + b_2 x \leq e_2}{|a_2|a_0 x + |b_0|b_2 x \leq |a_2|e_0 + |b_0|e_2} \; y$$

and

$$\frac{a_1x + b_1u \le e_1 \quad a_3u + b_3y \le e_3}{|a_3|a_1x + |b_1|b_3y \le |a_3|e_1 + |b_1|e_3}\ u$$

then

$$\frac{c_2' \quad c_{13}}{c}\ +_{(|a_1||a_3|,|a_0||a_2|,0)}$$

Hence, c is redundant, contradicting the pre-condition of the lemma that $c \in \mathsf{complete}(I \cup \{c_0\})$.

4. Now consider the case where the eliminated variables are x, y, v.
   (a) Suppose that y and z are the same variable. Then: $c_0 \equiv a_0x + b_0y \le e_0$, $c_1 \equiv a_1x + b_1u \le e_1$, $c_2 \equiv a_2y + b_2v \le e_2$ and $c_3 \equiv a_3v + b_3y \le e_3$. This means that

$$\begin{aligned} c = {} & |a_0||a_2||a_3|b_1u + |a_1||b_0||b_2|b_3y \le \\ & |a_1||a_2||a_3|e_0 + |a_0||a_2||a_3|e_1 + |a_1||a_3||b_0|e_2 + |a_1||b_0||b_2|e_3 \end{aligned}$$

   Now consider the following two eliminations, defining c_1' and c_{23}:

$$\frac{a_0x + b_0y \le e_0 \quad a_1x + b_1u \le e_1}{|a_1|b_0y + |a_0|b_1u \le |a_1|e_0 + |a_0|e_1}\ x$$

   and

$$\frac{a_2y + b_2v \le e_2 \quad a_3v + b_3y \le e_3}{|a_3|a_2y + |b_2|b_3y \le |a_3|e_2 + |b_2|e_3}\ v$$

   then

$$\frac{c_1' \quad c_{23}}{c}\ +_{(|a_2||a_3|,|a_1||b_0|,0)}$$

   Hence, c is redundant, contradicting the pre-condition of the lemma that $c \in \mathsf{complete}(I \cup \{c_0\})$.
   (b) Suppose that x and z are the same variable, then the reduction still holds.
   (c) Suppose that x and v are the same variable, then the reduction still holds. □

Lemma 8. Suppose $\mathsf{vars}(c) = \{x\}$, $\mathsf{vars}(c_1), \mathsf{vars}(c_2) \subseteq \{x\}$ and that $\{c_1, c_2\} \models c$. Then either $c_1 \models c$ or $c_2 \models c$.

Proof 8. If $|\mathsf{vars}(c_i)| = 0$ for $i = 1$ or $i = 2$ then the result follows immediately. Suppose that c_1, c_2 and c are:

$$c_1 \equiv a_1x \le e_1 \qquad c_2 \equiv a_2x \le e_2 \qquad c \equiv (a_1 + a_2)x \le e_1 + e_2 + \delta \equiv ax \le e$$

Then for $a_1, a_2 \neq 0$, it is supposed that any multipliers λ_i have been absorbed into coefficients a_1, a_2 and that $\delta \ge 0$):

First suppose that $\mathsf{sign}(a_1) = \mathsf{sign}(a_2)$, hence $\mathsf{sign}(a_1 + a_2) = \mathsf{sign}(a)$ and it can further be assumed that $|a_1| = |a_2| = 1$. Without loss of generality $\frac{e_1}{|a_1|} \leq \frac{e_2}{|a_2|}$ (hence $c_1 \models c_2$). Then:

$$2x \leq 2\frac{e_1}{|a_1|} \leq \frac{e_1}{|a_1|} + \frac{e_2}{|a_2|} \leq e_1 + e_2 + \delta$$

and $c_1 \models c$.

Next suppose that $\mathsf{sign}(a_1) \neq \mathsf{sign}(a_2)$. Consistency implies that:

$$0 \leq e_1 + \frac{e_2|a_1|}{|a_2|}$$

Without loss of generality assume that $\mathsf{sign}(a_1) = \mathsf{sign}(a_1 + a_2)$, hence $|a_1| \geq |a_2|$ and $|a_1 + a_2| = |a_1| - |a_2|$.

Observe that:

$$\begin{aligned} 0 &\leq e_1 + \frac{e_2|a_1|}{|a_2|} \\ \frac{e_1|a_1|}{|a_2|} &\leq e_1 + \frac{e_2|a_1|}{|a_2|} + \frac{e_1|a_1|}{|a_2|} \\ e_1|a_1| - e_1|a_2| &\leq e_1|a_1| + e_2|a_1| \\ e_1(|a_1| - |a_2|) &\leq (e_1 + e_2)|a_1| \end{aligned}$$

Hence

$$\frac{e_1}{|a_1|} \leq \frac{e_1 + e_2}{|a_1 - a_2|} = \frac{e_1 + e_2}{|a_1| + |a_2|}$$

and the result holds. □

Lemma 9 (Unary inequalities). Suppose that $I \subseteq \mathsf{TVPI}_X$ and $\mathsf{complete}(I) = I$. Suppose $c_0 \in \mathsf{TVPI}_X$ and that $c \in \mathsf{complete}(I \cup \{c_0\})$ where

$$c = \mathsf{result}(\mathsf{result}(\mathsf{result}(c_0, c_1), c_2), c_3)$$

and $|\mathsf{vars}(c)| = 1$, $c_1, c_2 \in I$ and $c_3 \in I \cup \{c_0\}$. Then there exists $d_0, d_1 \in I$ such that one of the following holds:

1. $c = \mathsf{result}(\mathsf{result}(c_0, d_0), d_1)$
2. $c = \mathsf{result}(c_0, d_0)$

Proof 9. Figure 4 gives the structure of three resultant steps, and skeletons (the variables occurring in each inequality) for each of the fifteen possible configurations where three resultant steps end in a unary inequality. Each case will be analysed in turn (many of these can be treated similarly). The algebraic detail is shown in cases where detailed manipulation is required.

1. Where $c_0 \equiv a_0x + b_0u \le e_0$, $c_1 \equiv a_1z + b_1u \le e_1$, $c_2 \equiv a_2z + b_2y \le e_2$ and $c_3 \equiv a_3y \le e_3$. Consider:

$$\cfrac{\cfrac{c_0 \quad c_1}{c_1'}\,u \qquad \cfrac{c_2 \quad c_3}{c_{23}}\,y}{c}\,z$$

$c_{23} = \mathsf{result}(c_2, c_3, y)$ is a unary inequality, hence either $c_{23} \in I$ and case 1 holds or $c_{23} \notin I$ and by Lemma 6 $c \notin \mathsf{complete}(I \cup \{c_0\})$. Note that the result still holds if $u = y$.

2. Where $c_0 \equiv a_0z + b_0u \le e_0$, $c_1 \equiv a_1y + b_1u \le e_1$, $c_2 \equiv a_2x + b_2z \le e_2$ and $c_3 \equiv a_3y \le e_3$. Consider:

$$\cfrac{\cfrac{c_0 \quad \cfrac{c_1 \quad c_3}{c_{13}}\,y}{c_{13}'}\,u \qquad c_2}{c}\,z$$

$c_{13} = \mathsf{result}(c_1, c_3, y)$ is a unary inequality, hence either $c_{13} \in I$ and case 1 holds or $c_{13} \notin I$ and by Lemma 6 $c \notin \mathsf{complete}(I \cup \{c_0\})$. Note that the result still holds if $u = x$.

3. Where $c_0 \equiv a_0z + b_0u \le e_0$, $c_1 \equiv a_1y + b_1u \le e_1$, $c_2 \equiv a_2z \le e_2$ and $c_3 \equiv a_3x + b_3y \le e_3$. The original derivation is:

$$\cfrac{\cfrac{\cfrac{a_0z + b_0u \le e_0 \quad a_1y + b_1u \le e_1}{|b_1|a_0z + |b_0|a_1y \le |b_1|e_0 + |b_0|e_1}\,u \qquad a_2z \le e_2}{|a_2||b_0|a_1y \le |a_2||b_1|e_0 + |a_2||b_0|e_1 + |a_0||b_1|e_2}\,z \qquad a_3x + b_3y \le e_3}{|a_1||a_2||b_0|a_3x \le |a_2||b_1||b_3|e_0 + |a_2||b_0||b_3|e_1 + |a_0||b_1||b_3|e_2 + |a_1||a_2||b_0|e_3}\,y$$

Hence the same inequality might be derived

$$\cfrac{\cfrac{a_0z + b_0u \le e_0 \quad a_2z \le e_2}{|a_2|b_0u \le |a_2|e_0 + |a_0|e_2}\,z \qquad \cfrac{a_1y + b_1u \le e_1 \quad a_3x + b_3y \le e_3}{|a_1|a_3x + |b_3|b_1u \le |b_3|e_1 + |a_1|e_3}\,y}{|a_1||a_2||b_0|a_3x \le |a_2||b_1||b_3|e_0 + |a_2||b_0||b_3|e_1 + |a_0||b_1||b_3|e_2 + |a_1||a_2||b_0|e_3}\,u$$

If $c_{13} = \mathsf{result}(c_1, c_3, y) \in I$ then case 1 holds. If $c_{13} \notin I$ then by Lemma 6 $c \notin \mathsf{complete}(I \cup \{c_0\})$. Note that the result still holds if $z = x$.
If in the derivation immediately above $u = x$, then c_2' and c_{13} are given by:

$$\cfrac{a_0z + b_0x \le e_0 \quad a_2z \le e_2}{|a_2|b_0x \le |a_2|e_0 + |a_0|e_2}\,z$$

and

$$\cfrac{a_1y + b_1x \le e_1 \quad a_3x + b_3y \le e_3}{(|a_1|a_3 + |b_3|b_1)x \le |b_3|e_1 + |a_1|e_3}\,y$$

Hence (scaling with $\lambda_1 = |b_1||b_3|$ and $\lambda_2 = |a_2||b_0|$ respectively)

$$\begin{aligned}&|a_2||b_0|(|a_1|a_3 + |b_3|b_1)x + |a_2||b_1||b_3|b_0x \le \\ &\qquad |a_2||b_0|(|b_3|e_1 + |a_1|e_3) + |b_1||b_3|(|a_2|e_0 + |a_0|e_2)\end{aligned}$$

Noting that $b_0 b_1 < 0$ this gives

$$|a_1||a_2||b_0|a_3 x \leq |b_1||b_3||a_2|e_0 + |a_2||b_0||b_3|e_1 + |a_0||b_1||b_3|e_2 + |a_1||a_2||b_0|e_3$$

Hence by Lemma 8 $c'_2 \models c$ or $c_{13} \models c$. In the former case then either case 2 has been demonstrated or $c \notin \mathsf{complete}(I \cup \{c_0\})$. In the latter case, either $c \in I$ or $c \notin \mathsf{complete}(I \cup \{c_0\})$. In all cases the result holds.

4. Where $c_0 \equiv a_0 z + b_0 u \leq e_0$, $c_1 \equiv a_1 y + b_1 u \leq e_1$, $c_2 \equiv a_2 z + b_2 y \leq e_2$ and $c_3 \equiv a_3 x + b_3 y \leq e_3$. The original derivation gives:

$$\begin{aligned} c = {} & |(|a_2||b_0|a_1 + |a_0||b_1|b_2)|a_3 x \leq \\ & |a_2||b_1||b_3|e_0 + |a_2||b_0||b_3|e_1 + |a_0||b_1||b_3|e_2 + \\ & |(|a_2||b_0|a_1 + |a_0||b_1|b_2)|e_3 \end{aligned}$$

There are three subcases to consider:
(a) where $a_1 b_3 < 0$, but $b_2 b_3 \geq 0$
(b) where $a_1 b_3 \geq 0$, but $b_2 b_3 < 0$
(c) where $a_1 b_3 < 0$ and $b_2 b_3 < 0$
Consider each of these in turn:
(a) Here

$$\dfrac{\dfrac{c_0 \quad \dfrac{c_1 \quad c_2}{c_{12}}\, y}{c'_{12}}\, z \qquad \dfrac{c_1 \quad c_3}{c_{13}}\, y}{c'}\, u$$

where c' is:

$$\begin{aligned} & |(|a_1||a_2|b_0 + |a_0||b_2|b_1)||a_1|a_3 x \leq \\ & \quad |(|a_1||a_2|b_0 + |a_0||b_2|b_1)|(|b_3|e_1 + |a_1|e_3) + \\ & \quad |b_1||b_3|(|a_1||a_2|e_0 + |a_0||b_2|e_1 + |a_0||a_1|e_2) \end{aligned}$$

Note that $b_0 b_1 < 0$, $a_1 b_2 < 0$ and $|a_1||a_2||b_0| > |a_0||b_2||b_1|$. Dividing through by $|a_1|$ shows that $c' \equiv c$ and case 1 of the result holds, except in the case where one or both of $\mathsf{result}(c_1, c_2, y)$ and $\mathsf{result}(c_1, c_3, y)$ are redundant. In these cases Lemma 6 shows that $c' \equiv c \notin \mathsf{complete}(I \cup \{c_0\})$. Note that the result still holds if $z = x$.
However, if $u = x$, instead consider derivations of c_{13} and c'_{21}

$$\dfrac{a_1 y + b_1 x \leq e_1 \quad a_3 x + b_3 y \leq e_3}{(|b_3|b_1 + |a_1|a_3)x \leq |b_3|e_1 + |a_1|e_3}\, y$$

and

$$\dfrac{\dfrac{a_0 z + b_0 x \leq e_0 \quad a_2 z + b_2 y \leq e_2}{|a_2|b_0 x + |a_0|b_2 y \leq |a_2|e_0 + |a_0|e_2}\, z \qquad a_1 y + b_1 x \leq e_1}{(|a_1||a_2|b_0 + |a_0||b_2|b_1)x \leq |a_1||a_2|e_0 + |a_0||a_1|e_2 + |a_0||b_2|e_1}\, y$$

Combining with $\lambda_1 = |(|a_2||b_0|a_1 + |a_0||b_1|b_2)|$ and $\lambda_2 = |b_1||b_3|$

$$\begin{array}{l} (|(|a_2||b_0|a_1 + |a_0||b_1|b_2)||b_3|b_1 + |(|a_2||b_0|a_1 + |a_0||b_1|b_2)||a_1|a_3 \\ |a_1||a_2||b_1||b_3|b_0 + |a_0||b_1||b_2||b_3|b_1)x \\ \qquad\qquad \leq \\ \qquad\qquad (||a_2||b_0|a_1 + |a_0||b_1|b_2|)(|b_3|e_1 + |a_1|e_3)+ \\ \qquad\qquad |b_1||b_3|(|a_1||a_2|e_0 + |a_0||a_1|e_2 + |a_0||b_2|e_1) \end{array}$$

Which, noting that $(|a_2||b_0|b_1 + |a_0||b_1|b_2)b_3 < 0$ and that $b_0b_1 < 0$ simplifies to (after dividing through by $|a_1|$):

$$\begin{array}{c} |(|a_2||b_0|a_1 + |a_0||b_1|b_2)|a_3x \\ \leq \\ |a_2||b_1||b_3|e_0 + |a_2||b_0||b_3|e_1 + |a_0||b_1||b_3|e_2 + |(|a_2||b_0|a_1 + |a_0||b_1|b_2)|e_3 \end{array}$$

By Lemma 8 this demonstrates that either $c_{13} \models c$ or $c'_{21} \models c$. In the first case either $c \in I$ or $c \notin \mathsf{complete}(I \cup \{c_0\})$. In the second case either case 1 of the lemma holds or $c \notin \mathsf{complete}(I \cup \{c_0\})$.

(b) This is symmetric to the previous case.

(c) Here, the following:

$$\dfrac{\dfrac{c_0 \quad \dfrac{c_1 \quad c_3}{c_{13}}\,y}{c'_{13}}\,u \qquad \dfrac{c_2 \quad c_3}{c_{23}}\,y}{c'}\,z$$

where c' is

$$\begin{array}{l} (|a_1||a_2||b_0||b_3|a_3 + |b_1||b_3||a_0||b_2|a_3)x \leq \\ \qquad |b_1||a_2||b_3||b_3|e_0 + |b_0||a_2||b_3||b_3|e_1+ \\ \qquad |a_1||b_0||a_2||b_3|e_3 + |a_0||b_1||b_3||b_3|e_2 + |a_0||a_2||b_1||b_3|e_3 \end{array}$$

Note that $\mathsf{sign}(a_1) = \mathsf{sign}(b_2)$, hence $|a_1||a_2||b_0|a_3 + |b_1||a_0||b_2|a_3 = |(|a_2||b_0|a_1 + |b_1||a_0|b_2)|a_3$ after dividing through by $|b_3|$. This demonstrates that $c' \equiv c$ and that case 1 of the lemma holds unless one or both of $\mathsf{result}(c_1, c_3, y)$ and $\mathsf{result}(c_2, c_3, y)$ are redundant, in which case by Lemma 6 $c \notin \mathsf{complete}(I \cup \{c_0\})$.

If $u = x$, then consider the derivations of c'_{23} and c_{13}

$$\dfrac{\dfrac{a_0z + b_0x \leq e_0 \quad a_2z + b_2y \leq e_2}{|a_2|b_0x + |a_0|b_2y \leq |a_2|e_0 + |a_0|e_2}\,z \qquad a_3x + b_3y \leq e_3}{(|a_2||b_3|b_0 + |a_0||b_2|a_3)x \leq |a_2||b_3|e_0 + |a_0||b_3|e_2 + |a_0||b_2|e_3}\,y$$

and

$$\dfrac{a_1y + b_1x \leq e_1 \quad a_3x + b_3y \leq e_3}{(|b_3|b_1 + |a_1|a_3)x \leq |b_3|e_1 + |a_1|e_3}\,y$$

With $\lambda_1 = |b_1|$ and $\lambda_2 = |a_2||b_0|$ and noting that $b_0b_1 < 0$, this gives

$$\begin{array}{l} (|a_1||a_2||b_0| + |a_0||b_1||b_2|)a_3x \leq \\ \qquad |a_2||b_1||b_3|e_0 + |a_2||b_0||b_3|e_1 + |a_2||b_1||b_3|e_2+ \\ \qquad (|a_0||b_1||b_2| + |a_1||a_2||b_0|)e_3 \end{array}$$

and since $\mathsf{sign}(a_1) = \mathsf{sign}(b_2)$

$$\begin{aligned}|(|a_2||b_0|a_1 + |a_0||b_1|b_2)|a_3x \leq \\ &|a_2||b_1||b_3|e_0 + |a_2||b_0||b_3|e_1 + |a_2||b_1||b_3|e_2 + \\ &(|a_2||b_0|a_1 + |a_0||b_1|b_2)e_3\end{aligned}$$

Which by Lemma 8 demonstrates that either $c'_{23} \models c$ or $c_{13} \models c$. In the first case either case 1 of the lemma holds or $c' \equiv c \notin \mathsf{complete}(I \cup \{c_0\})$, in which case the lemma does not apply to c. In the second case $c' \equiv c \in I$ or $c' \equiv c \notin \mathsf{complete}(I \cup \{c_0\})$. This gives the result. If $z = x$, then consider derivations of c'_{13} and c_{23}

$$\dfrac{\dfrac{a_0x + b_0u \leq e_0 \quad a_1y + b_1u \leq e_1}{|b_1|a_0x + |b_0|a_1y \leq |b_1|e_0 + |b_0|e_1}\, u \qquad a_3x + b_3y \leq e_3}{(|b_1||b_3|a_0 + |a_1||b_0|a_3)x \leq |b_1||b_3|e_0 + |b_0||b_3|e_1 + |b_0||a_1|e_3}\, y$$

and

$$\dfrac{a_2x + b_2y \leq e_2 \quad a_3x + b_3y \leq e_3}{(|b_3|a_2 + |b_2|a_3)x \leq |b_3|e_2 + |b_2|e_3}\, y$$

With $\lambda_1 = |a_2|$ and $\lambda_2 = |a_0||b_1|$ and noting that $a_0a_2 < 0$ this gives

$$\begin{aligned}(|a_1||a_2||b_0| + |a_0||b_1||b_2|)a_3x \leq \\ &|a_2||b_1||b_3|e_0 + |a_2||b_0||b_3|e_1 + |a_0||b_1||b_3|e_2 + \\ &(|a_2||b_0||a_1| + |a_0||b_1||b_2|)e_3\end{aligned}$$

and since $\mathsf{sign}(a_1) = \mathsf{sign}(a_2)$

$$\begin{aligned}|(|a_1||b_0|a_2 + |a_0||b_1|b_2)|a_3x \leq \\ &|a_2||b_1||b_3|e_0 + |a_2||b_0||b_3|e_1 + |a_0||b_1||b_3|e_2 + \\ &(|a_2||b_0||a_1| + |a_0||b_1||b_2|)e_3\end{aligned}$$

and by Lemma 8 this demonstrates that either $c'_{13} \models c$ or $c_{23} \models c$. In the first case either case 1 of the lemma holds or $c \notin \mathsf{complete}(I \cup \{c_0\})$. In the second case $c \in I$ or $c \notin \mathsf{complete}(I \cup \{c_0\})$. This gives the result.

5. Where $c_0 \equiv a_0z + b_0u \leq e_0$, $c_1 \equiv a_1u \leq e_1$, $c_2 \equiv a_2z + b_2y \leq e_2$, $c_3 \equiv a_3x + b_3y \leq e_3$, the initial derivation is:

$$\dfrac{\dfrac{\dfrac{c_0 \quad c_1}{c'_1}\, u \qquad c_2}{c'_2}\, z \qquad c_3}{c}\, y$$

This can be rewritten to:

$$\dfrac{\dfrac{c_0 \quad c_1}{c'_1}\, u \qquad \dfrac{c_2 \quad c_3}{c_{23}}\, y}{c}\, z$$

If $c_{23} \in I$ this demonstrates case 1 of the result, or else c_{23} is redundant and by Lemma 6 $c \notin \mathsf{complete}(I \cup \{c_0\})$. Note that the result still holds if $u = y$ or if $u = x$.
If $z = x$ consider c_1' and c_{23}

$$\frac{c_0 \quad c_1}{c_1'}\; u \qquad \frac{c_2 \quad c_3}{c_{23}}\; y$$

then $\{c_1', c_{23}\} \models c$ and again by Lemma 8 either $c_1' \models c$ or $c_{23} \models c$. If the former then either case 2 of the result holds or $c \notin \mathsf{complete}(I \cup \{c_0\})$. In the latter then either $c \in I$ or $c \notin \mathsf{complete}(I \cup \{c_0\})$.

6. As case 5.
7. As case 5.
8. Where $c_0 \equiv a_0 z + b_0 u \leq e_0$, $c_1 \equiv a_1 y + b_1 u \leq e_1$, $c_2 \equiv a_2 x + b_2 z \leq e_2$, $c_3 \equiv a_3 x + b_3 y \leq e_3$, the initial derivation is:

$$\frac{\dfrac{\dfrac{c_0 \quad c_1}{c_1'}\; u \quad c_2}{c_2'}\; z \quad c_3}{c}\; y$$

This can be rewritten to:

$$\frac{\dfrac{c_0 \quad c_2}{c_2'}\; z \quad \dfrac{c_1 \quad c_3}{c_{13}}\; y}{c}\; u$$

If $c_{13} \in I$ this demonstrates case 1 of the result. If $c_{13} \notin I$ by Lemma 6 $c \notin \mathsf{complete}(I \cup \{c_0\})$. Note that if $u = x$ then $\{c_2', c_{13}\} \models c$ and again by Lemma 8 either $c_2' \models c$ or $c_{13} \models c$. If the former then either case 2 of the lemma holds or $c \notin \mathsf{complete}(I \cup \{c_0\})$. If the latter then either $c \in I$ or $c \notin \mathsf{complete}(I \cup \{c_0\})$.

9. Where $c_0 \equiv a_0 y + b_0 u \leq e_0$, $c_1 \equiv a_1 z + b_1 u \leq e_1$, $c_2 \equiv a_2 z + b_2 y \leq e_2$, $c_3 \equiv a_3 x + b_3 y \leq e_3$, the initial derivation is:

$$\frac{\dfrac{\dfrac{c_0 \quad c_1}{c_1'}\; z \quad c_2}{c_2'}\; u \quad c_3}{c}\; y$$

This can be rewritten to:

$$\frac{\dfrac{c_0 \quad \dfrac{c_1 \quad c_2}{c_{12}}\; z}{c_{12}'}\; u \quad c_3}{c}\; y$$

If $c_{12} \in I$ this demonstrates case 1 of the result. Else $c_{12} \notin I$ and by Lemma 6 $c \notin \mathsf{complete}(I \cup \{c_0\})$. Note that the result still holds if $u = x$ or if $z = x$.

10. As case 1.
11. Where $c_0 \equiv a_0 y + b_0 u \leq e_0$, $c_1 \equiv a_1 z + b_1 u \leq e_1$, $c_2 \equiv a_2 x + b_2 z \leq e_2$, $c_3 \equiv a_3 y \leq e_3$, the initial derivation is:

$$\dfrac{\dfrac{\dfrac{c_0 \quad c_1}{c_1'}\,u \qquad c_2}{c_2'}\,z \qquad c_3}{c}\,y$$

This can be rewritten to:

$$\dfrac{\dfrac{c_0 \quad c_3}{c_3'}\,y \quad \dfrac{c_1 \quad c_2}{c_{12}}\,z}{c}\,u$$

Either $c_{12} \in I$ and this demonstrates case 1 of the result, or $c_{12} \notin I$ and by Lemma 6 $c \notin \mathsf{complete}(I \cup \{c_0\})$. Note that if $u = x$ then $\{c_3', c_{12}\} \models c$ and again by Lemma 8 either $c_3' \models c$ or $c_{12} \models c$. For the former, either case 2 of the lemma holds or $c \notin \mathsf{complete}(I \cup \{c_0\})$. For the latter, either $c \in I$ or $c \notin \mathsf{complete}(I \cup \{c_0\})$.
12. As case 5.
13. As case 11.
14. As case 9.
15. As case 9. □

Lemma 10 (Linearisation). Let $I \subseteq \mathsf{TVPI}_X$. Further, suppose that $c_0, c_1, \ldots, c_j \in I$ and $d_0, d_1, \ldots, d_k \in I$ where $j, k \in \mathbb{N}$. Where $c_0' = c_0$ and $d_0' = d_0$, define $c_{i+1}' = \mathsf{result}(c_i', c_{i+1}, x_i)$, where $x_i \in X$ and also define $d_{i+1}' = \mathsf{result}(d_i', d_{i+1}, y_i)$, where $y_i \in X$. Consider $c = \mathsf{result}(c_j', d_k', z)$, where $z \in X$. Then there exists $f_0, f_1, \ldots, f_\ell \in I$, where $f_0 = f_0' = c_0$, $f_{i+1}' = \mathsf{result}(f_i', f_{i+1}, w_i)$, $w_i \in X$, and at least one of the following holds:

1. $c \equiv f_\ell'$
2. c is redundant with respect to $\mathsf{complete}(I)$, that is, $\mathsf{complete}(I) \models c$ and $c \notin \mathsf{complete}(I)$
3. $c \equiv \mathsf{false}$

Proof 10. If $c = \mathsf{result}(c_j', d_0, z)$, then case 1 is immediate. If $c = \mathsf{result}(c_0, d_k', z)$, note the symmetry of the premisses (so that $c = \mathsf{result}(c_j', d_k', z) = \mathsf{result}(d_k', c_j', z)$), hence case 1 is again immediate.

The remainder of the result is presented as a series of rewriting rules for derivations for which it will be argued that repeated application will establish the result. The basic case is that when all variables are distinct; this is followed by cases where these variables may coincide with each other.

Where $j, k \geq 0$ consider $c = \mathsf{result}(c_{j+1}', d_{k+1}', z)$.

- Suppose x, y, z, w are distinct variables. Then (where all coefficients are non-zero) there are two possibilities, corresponding to the premiss of d_{k+1}' that z derives from:

1.

$$\dfrac{a_1x + b_1z \le e_1 \qquad \dfrac{a_2z + b_2w \le e_2 \quad a_3w + b_3y \le e_3}{|a_3|a_2z + |b_2|b_3y \le |a_3|e_2 + |b_2|e_3}\ w}{|a_2||a_3|a_1x + |b_1||b_2|b_3y \le |a_2||a_3|e_1 + |a_3||b_1|e_2 + |b_1||b_2|e_3}\ z$$

which can be rewritten to:

$$\dfrac{\dfrac{a_1x + b_1z \le e_1 \quad a_2z + b_2w \le e_2}{|a_2|a_1x + |b_1|b_2w \le |a_2|e_1 + |b_1|e_2}\ z \qquad a_3w + b_3y \le e_3}{|a_2||a_3|a_1x + |b_1||b_2|b_3y \le |a_2||a_3|e_1 + |a_3||b_1|e_2 + |b_1||b_2|e_3}\ w$$

That is,

$$c = \mathsf{result}(c'_{j+1}, \mathsf{result}(d'_k, d_k, w), z) = \mathsf{result}(\mathsf{result}(c'_{j+1}, d'_k, z), d_k, w)$$

2. Symmetrically

$$\dfrac{a_1x + b_1z \le e_1 \qquad \dfrac{a_2y + b_2w \le e_2 \quad a_3w + b_3z \le e_3}{|a_3|a_2y + |b_2|b_3z \le |a_3|e_2 + |b_2|e_3}\ w}{|b_2||b_3|a_1x + |a_3||b_1|a_2y \le |b_2||b_3|e_1 + |b_1||a_3|e_2 + |b_1||b_2|e_3}\ z$$

which can be rewritten to:

$$\dfrac{\dfrac{a_1x + b_1z \le e_1 \quad a_3w + b_3z \le e_3}{|b_3|a_1x + |b_1|a_3w \le |b_3|e_1 + |b_1|e_3}\ z \qquad a_2y + b_2w \le e_2}{|b_2||b_3|a_1x + |a_3||b_1|a_2y \le |b_2||b_3|e_1 + |a_3||b_1|e_2 + |b_1||b_2|e_3}\ w$$

That is,

$$c = \mathsf{result}(c'_{j+1}, \mathsf{result}(d'_k, d_k, w), z) = \mathsf{result}(\mathsf{result}(c'_{j+1}, d_k, z), d'_k, w)$$

- Now suppose that in the above $x = y$ and the other variables are still distinct. Observe that the two rewritings are still valid (since x and y are not eliminated).

- Next suppose that in the above $x = z$ and that the other variables are still distinct. Observe that the initial derivations can describe this situation by putting $a_1 = 0$ and that the two rewritings are still valid.

- Next suppose that in the above $y = w$ and that the other variables are still distinct. Observe that the initial derivations can describe this situation by putting $b_3 = 0$ in the first case and $a_2 = 0$ in second, and that the two rewritings are still valid.

- Notice that if in the above $z = w$ and the other variables are still distinct, then the final resultant eliminating z is not possible.

- If $y = z$ and the remaining variables are still distinct, then $\mathsf{result}(c'_{j+1}, d'_{k+1}, z)$ can be rewritten to $\mathsf{result}(d'_{k+1}, c'_{j+1}, z)$.

- Suppose that $x = w$ and x, y, z are distinct

$$\dfrac{a_1x + b_1z \le e_1 \quad \dfrac{a_2z + b_2x \le e_2 \quad a_3x + b_3y \le e_3}{|a_3|a_2z + |b_2|b_3y \le |a_3|e_2 + |b_2|e_3}\ x}{|a_2||a_3|a_1x + |b_1||b_2|b_3y \le |a_2||a_3|e_1 + |a_3||b_1|e_2 + |b_1||b_2|e_3}\ z$$

allows c_{12} and c_3:

$$\dfrac{a_1x + b_1z \le e_1 \quad a_2z + b_2x \le e_2}{(|a_2|a_1 + |b_1|b_2)x \le |a_2|e_1 + |b_1|e_2}\ z \qquad \text{and} \qquad a_3x + b_3y \le e_3$$

Similarly,

$$\dfrac{a_1x + b_1z \le e_1 \quad \dfrac{a_2y + b_2x \le e_2 \quad a_3x + b_3z \le e_3}{|a_3|a_2y + |b_2|b_3z \le |a_3|e_2 + |b_2|e_3}\ x}{|b_2||b_3|a_1x + |a_3||b_1|a_2y \le |b_2||b_3|e_1 + |a_3||b_1|e_2 + |b_1||b_2|e_3}\ z$$

allows c_{13} and c_2:

$$\dfrac{a_1x + b_1z \le e_1 \quad a_3x + b_3z \le e_3}{(|b_3|a_1 + |b_1|a_3)x \le |b_3|e_1 + |b_1|e_3}\ z \qquad \text{and} \qquad a_2y + b_2x \le e_2$$

  In the first case, since $b_2a_3 < 0$, $|a_3||b_1|b_2x + |b_1||b_2|a_3x = 0$. Observe that $|a_3|(|a_2|a_1 + |b_1|b_2)x + |b_1||b_2|(a_3x + b_3y) = |a_2||a_3|a_1x + |a_3||b_1|b_2x + |b_1||b_2|a_3x + |b_1||b_2|b_3y = |a_2||a_3|a_1x + |b_1||b_2|b_3y$. Hence $\{c_{12}, c_3\} \models c$ that is, case 2 holds.

  Note that this holds whether or not $|a_2|a_1 + |b_1|b_2 = 0$. If the equality holds then c_{12} is either true or false.

  For the second rewriting, case 2 holds by an analogous argument.

- Suppose that $x = w$ and $y = z$, then $\mathsf{result}(c'_{j+1}, d'_{k+1}, z)$ can be rewritten to $\mathsf{result}(d'_{k+1}, c'_{j+1}, z)$.

- Suppose that $x = y$ and $z = w$. This is not possible.

- Suppose that $x = z$ and $y = w$. Then observe that the two rewritings are still valid.

- Suppose that $x = y = w$. Then $\mathsf{result}(c'_{j+1}, d'_{k+1}, z)$ can be rewritten to $\mathsf{result}(d'_{k+1}, c'_{j+1}, z)$.

- Suppose that $x = y = z$, with w distinct. That is, $c \equiv \mathsf{true}$ or $c \equiv \mathsf{false}$. Then case 2 and case 3 apply respectively.

- Suppose that $x = z = w$, with y distinct. This is not possible.

- Suppose that $y = z = w$, with x distinct. This is not possible.

- Suppose that $x = y = z = w$. This is not possible.

Associate a weight to the derivation of inequality c, $c = \mathsf{result}(c_1, c_2, v)$. This weight is an ordered triple $(n, |\mathsf{vars}(c_1)|, \mathsf{size}(c_2))$, where n is the number of inequalities above c in the derivation with a right premiss with size greater than 0. In the original derivation of c, observe that at most one result operation has $\mathsf{size}(c_2) > 0$ (that is, $n \leq 1$). Now observe that each rewriting step results in at most one result step with the size of the second argument greater than zero (again $n \leq 1$). Hence the weights are totally ordered. The rewriting process has terminated if there are no result steps with right premiss with size greater than 0 (i.e. when $n = 0$), or when it is observed that the concluding inequality is redundant. Now observe that for each rewriting step either c is found to be redundant or the weight of the derivation of c is strictly less than previously. Hence by induction the result holds. □

Lemma 11 (Multiple use: part 1). Let $c_0, c_1 \in \mathsf{TVPI}_X$, where $\mathsf{vars}(c_0) = \{x, y\}$. If $c = \mathsf{result}(\mathsf{result}(c_0, c_1, x), c_0, y)$ then there exists $c' = \mathsf{result}(c_0, c_1, y)$ such that $c \equiv c'$.

Proof 11. Where $c_0 \equiv a_0x + b_0y \leq e_0$, $c_1 \equiv a_1x + b_1y \leq e_1$, suppose that $c_1' \equiv \mathsf{result}(c_0, c_1, x)$ and $c \equiv \mathsf{result}(c_1', c_0, y)$, that is:

$$\cfrac{\cfrac{a_0x + b_0y \leq e_0 \quad a_1x + b_1y \leq e_1}{(|a_1|b_0 + |a_0|b_1)y \leq |a_1|e_0 + |a_0|e_1}\ x \qquad a_0x + b_0y \leq e_0}{|(|a_1|b_0 + |a_0|b_1)|a_0x \leq |a_1||b_0|e_0 + |a_0||b_0|e_1 + |(|a_1|b_0 + |a_0|b_1)|e_0}\ y$$

Notice that:

1. this is the only possible configuration. If $b_0 = 0$ then the second resultant step is not possible, and if $\mathsf{vars}(c_1) \neq \{x, y\}$ one of the two resultant steps is not possible.
2. $a_0a_1 < 0$
3. $(|a_1|b_0 + |a_0|b_1)b_0 < 0$, hence $b_0b_1 < 0$
4. Hence $|a_0b_1| > |a_1b_0|$

Since $b_0b_1 < 0$ $c' = \mathsf{result}(c_0, c_1, y)$ can be obtained as follows:

$$\frac{a_0x + b_0y \leq e_0 \quad a_1x + b_1y \leq e_1}{(|b_1|a_0 + |b_0|a_1)x \leq |b_1|e_0 + |b_0|e_1}\ y$$

It will be demonstrated below that $c = |a_0|c'$ (i.e. $c \equiv c'$). There are four cases to consider, depending on the signs of a_0, b_0, a_1, b_1.

1. $b_0 > 0, b_1 < 0, a_0 > 0, a_1 < 0$. First consider the coefficient:

$$\begin{aligned} |(|a_1|b_0 + |a_0|b_1)|a_0 &= a_0(-|a_1|b_0 - |a_0|b_1) \\ &= a_0(|b_0|a_1 + |b_1|a_0) \\ &= |a_0|(|b_0|a_1 + |b_1|a_0) \end{aligned}$$

Second consider the constant:

$$\begin{aligned} |a_1||b_0|e_0 + |a_0||b_0|e_1 + |(|a_1|b_0 + |a_0|b_1)|e_0 &= |a_1||b_0|e_0 + |a_0||b_0|e_1 \\ &\quad - |a_1|b_0e_0 - |a_0|b_1e_0 \\ &= |a_0|(|b_0|e_1 + |b_1|e_0) \end{aligned}$$

2. $b_0 > 0, b_1 < 0, a_0 < 0, a_1 > 0$. First consider the coefficient:

$$\begin{aligned} |(|a_1|b_0 + |a_0|b_1)|a_0 &= a_0(-|a_1|b_0 - |a_0|b_1) \\ &= a_0(-|b_0|a_1 - |b_1|a_0) \\ &= |a_0|(|b_0|a_1 + |b_1|a_0) \end{aligned}$$

Second consider the constant:

$$\begin{aligned} |a_1||b_0|e_0 + |a_0||b_0|e_1 + |(|a_1|b_0 + |a_0|b_1)|e_0 &= |a_1||b_0|e_0 + |a_0||b_0|e_1 \\ &\quad - |a_1|b_0e_0 - |a_0|b_1e_0 \\ &= |a_0|(|b_0|e_1 + |b_1|e_0) \end{aligned}$$

3. $b_0 < 0, b_1 > 0, a_0 > 0, a_1 < 0$. First consider the coefficient:

$$\begin{aligned} |(|a_1|b_0 + |a_0|b_1)|a_0 &= a_0(|a_1|b_0 + |a_0|b_1) \\ &= a_0(|b_0|a_1 + |b_1|a_0) \\ &= |a_0|(|b_0|a_1 + |b_1|a_0) \end{aligned}$$

Second consider the constant:

$$\begin{aligned} |a_1||b_0|e_0 + |a_0||b_0|e_1 + |(|a_1|b_0 + |a_0|b_1)|e_0 &= |a_1||b_0|e_0 + |a_0||b_0|e_1 \\ &\quad + |a_1|b_0e_0 + |a_0|b_1e_0 \\ &= |a_0|(|b_0|e_1 + |b_1|e_0) \end{aligned}$$

4. $b_0 < 0, b_1 > 0, a_0 < 0, a_1 > 0$. First consider the coefficient:

$$\begin{aligned} |(|a_1|b_0 + |a_0|b_1)|a_0 &= a_0(|a_1|b_0 + |a_0|b_1) \\ &= a_0(-|b_0|a_1 - |b_1|a_0) \\ &= |a_0|(|b_0|a_1 + |b_1|a_0) \end{aligned}$$

Second consider the constant:

$$\begin{aligned} |a_1||b_0|e_0 + |a_0||b_0|e_1 + |(|a_1|b_0 + |a_0|b_1)|e_0 &= |a_1||b_0|e_0 + |a_0||b_0|e_1 \\ &\quad + |a_1|b_0e_0 + |a_0|b_1e_0 \\ &= |a_0|(|b_0|e_1 + |b_1|e_0) \end{aligned}$$

In each case the result holds. □

Lemma 12 (Multiple use: part 2). Suppose that $I \subseteq \mathsf{TVPI}_X$, $\mathsf{complete}(I) = I$ and $c_1, c_2 \in I$. If $c \in \mathsf{complete}(I \cup \{c_0\})$, where $c = \mathsf{result}(\mathsf{result}(\mathsf{result}(c_0, c_1), c_2), c_0)$, with $|\mathsf{vars}(c)| \geq 1$, then there is $d_0 \in I$ such that $c \equiv \mathsf{result}(c_0, d_0)$.

Proof (Lemma 12). The potential combinations of variables occurring (and being eliminated) in c_0, c_1 and c_2 are given in Figure 6. The four numbered derivation skeletons correspond to the cases of the proof, the three $(*)$ cases are not possible.

Each of the four potential cases has already been considered in an earlier lemma.

1. This case is covered by Lemma 7 (compaction), case 4(a). This showed that:

$$\dfrac{\dfrac{c_0 \quad c_1}{c_1'} \qquad \dfrac{c_2 \quad c_0}{c_{20}}}{c}\;+$$

   hence $c \notin \mathsf{complete}(I \cup \{c_0\})$ and the lemma does not apply. Note that c_1' and c_{20} are both formed as in the lemma.
2. This case is covered by Lemma 9 (unary inequalities), case 6. This again showed that:

$$\dfrac{\dfrac{c_0 \quad c_1}{c_1'} \qquad \dfrac{c_2 \quad c_0}{c_{20}}}{c}\;+$$

   hence $c \notin \mathsf{complete}(I \cup \{c_0\})$ and the lemma does not apply. Again, c_1' and c_{20} are both formed as in the lemma.
3. This case is covered by Lemma 9 (unary inequalities), case 9. This showed that:

$$\dfrac{\dfrac{c_0 \quad \dfrac{c_1 \quad c_2}{c_{12}}}{c_{12}'} \qquad c_0}{c}$$

   Since $c \in \mathsf{complete}(I \cup \{c_0\})$, c_{12} is not redundant and this is now an instance of Lemma 11.
4. This case matches Lemma 9 (unary inequalities), case 12. However, a different analysis is needed. Where $c_0 \equiv a_0x+b_0y \leq e_0$, $c_1 \equiv a_1y+b_1z \leq e_1$ and $c_2 \equiv a_2y + b_2z \leq e_2$, $c = \mathsf{result}(\mathsf{result}(\mathsf{result}(c_0, c_1, y), c_2, z), c_0, y)$ is given by

$$\begin{array}{c}(|a_1||b_0||b_2| + |a_2||b_0||b_1|)a_0x \\ \leq \\ (|a_1||b_0||b_2| + |a_2||b_0||b_1|)e_0 + |b_0||b_0||b_2|e_1 + |b_0||b_0||b_1|e_2\end{array}$$

   Now observe that $\mathsf{result}(c_0, \mathsf{result}(c_1, c_2, z), y)$ is given by

$$|(|b_2|a_1 + |b_1|a_2)|a_0x \leq |(|b_2|a_1 + |b_1|a_2)|e_0 + |b_0||b_2|e_1 + |b_0||b_1|e_2$$

   and since $a_1a_2 > 0$

$$(|a_1||b_2| + |a_2||b_1|)a_0x \leq (|a_1||b_2| + |a_2||b_1|)e_0 + |b_0||b_2|e_1 + |b_0||b_1|e_2$$

   and multiplying through by $|b_0|$ gives the result. □