# A Reduction from Unbounded Linear Mixed Arithmetic Problems into Bounded Problems

Martin Bromberger

# A Reduction from Unbounded Linear Mixed Arithmetic Problems into Bounded Problems

Martin Bromberger
Max Planck Institute for Informatics and Saarland University
Saarland Informatics Campus, Germany
mbromber@mpi-inf.mpg.de

December 2, 2018

**Abstract**

We present a combination of the Mixed-Echelon-Hermite transformation and the Double-Bounded Reduction for systems of linear mixed arithmetic that preserve satisfiability and can be computed in polynomial time. Together, the two transformations turn any system of linear mixed constraints into a bounded system, i.e., a system for which termination can be achieved easily. Existing approaches for linear mixed arithmetic, e.g., branch-and-bound and cuts from proofs, only explore a finite search space after application of our two transformations. Instead of generating *a priori* bounds for the variables, e.g., as suggested by Papadimitriou, unbounded variables are eliminated through the two transformations. The transformations orient themselves on the structure of an input system instead of computing *a priori* (over-)approximations out of the available constants. Experiments provide further evidence to the efficiency of the transformations in practice. We also present a polynomial method for converting certificates of (un)satisfiability from the transformed to the original system.

## 1 Introduction

Efficient linear arithmetic decision procedures are important for various independent research lines, e.g., optimization, system modeling, and verification. We are interested in feasibility of linear arithmetic problems in the context of the combination of theories, as they occur, e.g., in SMT solving or theorem proving.

The SMT and theorem proving communities have presented several interesting and efficient approaches for pure linear rational arithmetic [18] as well as linear integer arithmetic [5, 8, 16, 20]. SMT research also starts to extend into linear mixed arithmetic [12, 18] because some applications require both rational and integer variables, e.g., planning/scheduling problems and verification of timed automata and hybrid systems.

We are interest in decision procedures for mixed arithmetic because of a possible combination with superposition [1, 4, 19]. In the superposition context, arithmetic constraints are part of the first-order clauses. The problems are typically unbounded due to transformations that turn the input formula into a superposition specific input format. Since these problems are unbounded, the search space becomes infinite, which is the case where termination becomes difficult for most linear arithmetic approaches. Unbounded problems appear also in other areas of automated reasoning. Either because of bad encodings, necessary but complicating transformations, e.g., slacking (see Section 5), or the sheer complexity of the verification goal. Hence, efficient techniques for handling unbounded problems are necessary for a generally reliable combined procedure.

It is theoretically very easy to achieve termination for linear integer and mixed arithmetic because of so called *a priori bounds*. For example, the *a priori* bounds presented by Papadimitriou [22] guarantee that a problem has a mixed solution if and only if the problem extended by the bounds $|x_i| \leq 2n(ma)^{2m+1}$ for every variable $x_i$ has a mixed solution. In these *a priori* bounds, $n$ is the number of variables, $m$ the number of inequalities, and $a$ the largest absolute value of any integer coefficient or constant in the problem. By extending a problem with those *a priori* bounds, we reduce the search space for a branch-and-bound solver (and many other mixed arithmetic decision procedures) to a finite search space. So branch-and-bound is guaranteed to terminate.

However, these bounds are so large that the resulting search space cannot be explored in reasonable time for many practical problems. One reason for the impracticability of *a priori* bounds is that they only take parameter sizes into account and not actually the structure of each problem. *A priori* bounds are not integrated in any state-of-the-art SMT solvers [3, 13, 14, 15, 17] since they are no help in practice. As far as we know, none of the state-of-the-art SMT solvers use any method that guarantees termination for linear integer or mixed arithmetic.

In this paper, we present satisfiability preserving transformations that reduce unbounded problems into bounded problems. On these bounded problems, most linear mixed decision procedures become terminating, which we show on the example of branch-and-bound. Our reduction works by eliminating unbounded variables. First, we use the Double-Bounded reduction (Section 4) to eliminate all unbounded inequalities from our constraint system. Then we use the Mixed-Echelon-Hermite transformation (Section 3) to shift the variables of our system to ones that are either bounded or do not appear in the new inequalities and are, therefore, eliminated. With Corollary 2 & Lemma 13 we explain how to efficiently convert certificates of (un)satisfiability between the transformed and the original system. Our method is efficient because it is fully guided by the structure of the problem. This is confirmed by experiments (Section 5). We also show how to efficiently determine when a problem is unbounded (Lemma 10). This prevents our solver from applying our transformations on bounded problems.

An extended version of this paper is available on arXiv [7]. It contains an appendix, where we explain how to implement the presented procedures in an incrementally efficient way. This is relevant for the implementation of an efficient SMT theory solver. The extended version also contains several new examples as well as additional implementation tricks.

## 2 Preliminaries

While the difference between matrices, vectors, and their components is always clear in context, we generally use upper case letters for matrices (e.g., $A$), lower case letters for vectors (e.g., $x$), and lower case letters with an index $i$ or $j$ (e.g., $b_i$, $x_j$) as components of the associated vector at position $i$ or $j$, respectively. The only exceptions are the row vectors $a_i^T = (a_{i1}, \dots, a_{in})$ of a matrix $A = (a_1, \dots, a_m)^T$, which already contain an index $i$ that indicates the row's position inside $A$. We also abbreviate the $n$-dimensional origin $(0, \dots, 0)^T$ as $0^n$. Moreover, we denote by $\text{piv}(A, j)$ the row index of the *pivot* of a column $j$, i.e., the smallest row index $i$ with a non-zero entry $a_{ij}$ or $m + j$ if there are no non-zero entries in column $j$.

A system of constraints $Ax \leq b$ is just a set of non-strict inequalities[1] $\{a_1^T x \leq b_1, \dots, a_m^T x \leq b_m\}$ and the *rational solutions* of this system are exactly those points $x \in \mathbb{Q}^n$ that satisfy all inequalities in this set. The row coefficients are given by $A = (a_1, \dots, a_m)^T \in \mathbb{Q}^{m \times n}$, the variables are given by $x = (x_1, \dots, x_n)^T$, and the inequality bounds are given by $b = (b_1, \dots, b_m)^T \in \mathbb{Q}^m$. Moreover, we assume that any constant rows $a_i = 0^n$

[1] All techniques discussed in this paper can be extended to strict inequalities with the help of $\delta$-rationals [18]. We will omit the strict inequalities and focus only on non-strict inequalities due to lack of space.

were eliminated from our system during an implicit preprocessing step. This is a trivial task and eliminates some unnecessarily complicated corner cases.

In this paper, we consider mixed constraint systems, i.e., variables are assigned a type: either rational or integer. Due to convenience, we assume that the first $n_1$ variables $(x_1, \dots, x_{n_1})$ are rational and the remaining $n_2$ variables $(x_{n_1+1}, \dots, x_n)$ are integer, where $n = n_1 + n_2$. A *mixed solution* is a point $x \in (\mathbb{Q}^{n_1} \times \mathbb{Z}^{n_2})$ that satisfy all inequalities in $Ax \leq b$ and we denote by $\mathcal{M}(Ax \leq b) = \{x \in (\mathbb{Q}^{n_1} \times \mathbb{Z}^{n_2}) : Ax \leq b\}$ the *set of mixed solutions* to the system of inequalities $Ax \leq b$. We sometimes need to relax the variables to be completely rational. Therefore, we denote by $\mathcal{Q}(Ax \leq b) = \{x \in \mathbb{Q}^n : Ax \leq b\}$ the *set of rational solutions* to the system of inequalities $Ax \leq b$.

Since $Ax \leq b$ and $A'x \leq b'$ are just sets, we can write their combination as $(Ax \leq b) \cup (A'x \leq b')$. A special system of inequalities is a system of equations $Dx = c$, which is equivalent to the combined system of inequalities $(Dx \leq c) \cup (-Dx \leq -c)$. We say that a constraint system implies an inequality $h^T x \leq g$, where $h \in \mathbb{Q}^n$, $h \neq 0^n$, and $g \in \mathbb{Q}$, if $h^T x \leq g$ holds for all $x \in \mathcal{Q}(Ax \leq b)$. In the same manner, a constraint system implies an equality $h^T x = g$, where $h \in \mathbb{Q}^n$, $h \neq 0^n$, and $g \in \mathbb{Q}$, if $h^T x = g$ holds for all $x \in \mathcal{Q}(Ax \leq b)$. A constraint implied by $Ax \leq b$ is *explicit* if it does appear in $Ax \leq b$. Otherwise, it is called *implicit*.

Most deductions on linear inequalities are based on Farkas' Lemma:

**Lemma 1** (Farkas' Lemma [6])**.** *$\mathcal{Q}(Ax \leq b) = \emptyset$ iff there exists a $y \in \mathbb{Q}^m$ with $y \geq 0^m$ and $y^T A = 0^n$ so that $y^T b < 0$, i.e., there exists a non-negative linear combination of inequalities in $Ax \leq b$ that results in an inequality $y^T Ax \leq y^T b$ that is constant and unsatisfiable. If such a $y$ exists, then we call it a* certificate of unsatisfiability*.*

We also frequently use the following lemma, which is just a reformulation of Farkas' Lemma:

**Lemma 2** (Linear Implication Lemma)**.** *Let $\mathcal{Q}(Ax \leq b) \neq \emptyset$, $h \in \mathbb{Q}^n \setminus \{0^n\}$, and $g \in \mathbb{Q}$. Then, $Ax \leq b$ implies $h^T x \leq g$ iff there exists a $y \in \mathbb{Q}^m$ with $y \geq 0^m$ and $y^T A = h^T$ so that $y^T b \leq g$, i.e., there exists a non-negative linear combination of inequalities in $Ax \leq b$ that results in the inequality $h^T x \leq g$.*

As we mentioned in the introduction, this paper describes equisatisfiable transformations for constraint systems. We transform the systems in such a way that most linear mixed decision procedures become terminating and still retain their general efficiency. We even show this on the example of branch-and-bound. Although we do not have the time to discuss all facets of branch-and-bound [23], we still want to give a short summary of the algorithm. Branch-and-bound is a recursive algorithm that computes mixed solutions for constraint systems. In each call of the algorithm, it first computes a rational solution $s$ to a constraint system $Ax \leq b$[2]. If there are none, then we know that $Ax \leq b$ has no mixed solution. We are also done in the case that $s$ is a mixed solution. Otherwise, we select one of the integer variables $x_i$ assigned to a fractional value $s_i \notin \mathbb{Z}$ and call branch-and-bound recursively on $(Ax \leq b) \cup (x_i \geq \lceil s_i \rceil)$ and $(Ax \leq b) \cup (x_i \leq \lfloor s_i \rfloor)$. If none of the recursive calls returns a mixed solution, then $Ax \leq b$ also does not have a mixed solution. Likewise, if one of them returns a mixed solution $s$, then it also is a mixed solution to $Ax \leq b$.

Branch-and-bound alone is already complete on bounded constraint systems, i.e., systems where all directions are bounded:

**Definition 1** (Bounded Direction)**.** *A* direction/*vector $h \in \mathbb{Q}^n \setminus \{0^n\}$ is* bounded *in the constraint system $Ax \leq b$ if there exist $l, u \in \mathbb{Q}$ such that $Ax \leq b$ implies $h^T x \leq u$ and $-h^T x \leq -l$. Otherwise, it is called unbounded.*

**Definition 2** (Bounded System)**.** *A constraint* system $Ax \leq b$ *is* bounded *if all directions $h \in \mathbb{Q}^n \setminus \{0^n\}$ are bounded. Otherwise, it is called unbounded.*

For bounded systems, branch-and-bound is one of the most popular and efficient algorithms. It may, however, diverge if the system has unbounded directions. Even so, not all unbounded systems are equally difficult. For instance, a system where all directions are unbounded has always a mixed solution:

**Lemma 3** (Absolutely Unbounded [10])**.** *If all directions are unbounded in a constraint* system $Ax \leq b$*, then the constraint system has an integer solution.*

In a previous article, we described two cube tests that detect and solve constraint systems with infinite lattice width (another name for absolutely unbounded systems) in polynomial time [10]. The case of absolutely unbounded systems is, therefore, trivial and branch-and-bound can be easily extended so it also becomes complete for absolutely unbounded systems.

[2] A rational solution can be computed in polynomial time [23].

The actual difficult case is when some directions are bounded and others unbounded. We call these systems *partially unbounded*. Here, branch-and-bound and most other algorithms diverge or become inefficient in practice. The transformations, which we present, are designed to efficiently handle this subclass of problems.

## 3 Mixed-Echelon-Hermite Transformation

Our overall goal is to present an equisatisfiable transformation that turns any constraint system into a system that is bounded, i.e., a system on which branch-and-bound and many other arithmetic decision procedures terminate. In this section, we only present such a transformation for a subset of constraint systems, which we call *double-bounded constraint systems*. We then show in the next section that each constraint system can be reduced to an equisatisfiable double-bounded system. We also show how to efficiently transform a mixed solution from the double-bounded reduction to a mixed solution for the original system.

**Definition 3** (Double-Bounded Constraint System). *A constraint system $Dx \leq u$ is* double-bounded *if $Dx \leq u$ implies $Dx \geq l$ for $l \in \mathbb{Q}^m$. For such a double-bounded system, we call the bounds $u$ the* upper bounds *of $Dx$ and the bounds $l$ the lower bounds of $Dx$. Moreover, we typically write $l \leq Dx \leq u$ instead of $Dx \leq u$ although the* lower bounds *$l$ are only implicit.*

Note that only the inequalities in a double-bounded constraint system are guaranteed to be bounded. Variables might still be unbounded. For instance, in the constraint system $1 \leq 3x_1 - 3x_2 \leq 2$ both inequalities are bounded but the variables $x_1$ and $x_2$ are not. Moreover, the above constraint system is also an example where branch-and-bound diverges. This means that even bounding all inequalities does not yet guarantee termination. So for our purposes, a double-bounded constraint system is still too complex.

This changes, however, if we also require that the coefficient matrix $D$ of our constraint system is a *lower triangular matrix with gaps*:

**Definition 4** (Lower Triangular Matrix with Gaps). *A matrix $A \in \mathbb{Q}^{m \times n}$ is* lower triangular with gaps *if it holds for each column $j$ that $piv(A, j) > m$ or that $piv(A, j) < piv(A, k)$ for all columns $k$ with $j < k \leq n$, i.e., column $j$ either has only zero entries or all pivoting entries right of $j$ have a higher row index.*

A matrix is lower triangular if and only if the row indices of its pivots are strictly increasing, i.e., $\mathrm{piv}(A, 1) < \ldots < \mathrm{piv}(A, n)$. If we also allow it to have gaps, only the row indices of pivots with non-zero columns have to be strictly increasing. Now we get termination for free because of our restrictions:

**Lemma 4** (Lower Triangular Double-Bounded Systems)**.** *Let $D \in \mathbb{Q}^{m\times n}$ be a lower triangular matrix with gaps and $l \leq Dx \leq u$ be a double-bounded constraint system. Then each variable $x_j$ is either bounded, i.e., $l \leq Dx \leq u$ implies that $l'_j \leq x_j \leq u'_j$ or its column in $D$ has only zero entries.*

*Proof.* Proof by induction. Assume that the above property already holds for all variables $x_k$ with $k < j$. Let $p = \operatorname{piv}(D, j)$. If $p > m$, then the column $j$ of $D$ is zero and we are done. If $p \leq m$, then the pivoting entry $d_{pj}$ of column $j$ is non-zero. Because of Definition 4 and our induction hypothesis, this also means that each column $k$ with $k < j$ has either a zero entry in row $p$ or the variable $x_k$ is bounded by our induction hypothesis, i.e., $l \leq Dx \leq u$ implies $l'_k \leq x_k \leq u'_k$. Since Definition 4 also implies that row $p$ has only zero entries to the right of $d_{pj}$, the row $p$ has only one unbounded variable with a non-zero entry, viz., $x_j$. This means we can transform the row $l_p \leq d_p^T x \leq u_p$ into the following two inequalities: $l_p - \sum_{k=1}^{j-1} d_{pk}x_k \leq d_{pj}x_j$ and $u_p - \sum_{k=1}^{j-1} d_{pk}x_k \geq d_{pj}x_j$, where the variables $x_k$ on the left sides are either bounded or $d_{pk} = 0$. Hence, we can derive an upper and lower bound for $x_j$ via bound propagation/refinement [21]. □

**Corollary 1** (BnB-LTDB-Termination)**.** *Branch-and-bound terminates on every double-bounded system $l \leq Dx \leq u$ where $D$ is lower triangular with gaps.*

Our next goal is to efficiently transform every double-bounded system $l \leq Dx \leq u$ into an equisatisfiable system that also has a lower triangular coefficient matrix with gaps. We start by defining a class of transformations that do not only preserve mixed equisatisfiability, but are also very expressive.

**Definition 5** (Mixed Column Transformation Matrix [12])**.** *Given a mixed constraint system. A matrix $V \in \mathbb{Q}^{n\times n}$ is a* mixed column transformation matrix *if it is invertible and consists of an invertible matrix $V_{(\mathbb{Q})} \in \mathbb{Q}^{n_1\times n_1}$, a unimodular matrix $V_{(\mathbb{Z})} \in \mathbb{Z}^{n_2\times n_2}$, and a matrix $V_{(M)} \in \mathbb{Q}^{n_1\times n_2}$ such that*

$$V = \begin{pmatrix} V_{(\mathbb{Q})} & V_{(M)} \\ 0^{n_2\times n_1} & V_{(\mathbb{Z})} \end{pmatrix}.$$

The inverse of a mixed column transformation matrix $V$ is also a mixed column transformation matrix and can be used to undo the transformation $V$:

**Lemma 5** (Mixed Column Transformation Inversion [12])**.** *Given a mixed constraint system. Let $V \in \mathbb{Q}^{n\times n}$ be a mixed column transformation matrix. Then $V^{-1}$ is also a mixed column transformation matrix.*

This means that each mixed column transformation matrix defines a bijection from $(\mathbb{Q}^{n_1} \times \mathbb{Z}^{n_2})$ to $(\mathbb{Q}^{n_1} \times \mathbb{Z}^{n_2})$. Hence, they guarantee mixed equisatisfiability:

**Lemma 6** (Mixed Column Transformation Equisatisfiability [12])**.** *Let $Ax \leq b$ be a mixed constraint system. Let $V \in \mathbb{Q}^{n\times n}$ be a mixed column transformation matrix. Then every solution $y \in \mathcal{M}((AV)y \leq b))$ can be converted into a solution $Vy = x \in \mathcal{M}(Ax \leq b)$ and vice versa.*

Moreover, the mixed column transformation matrix $V$ also establishes a direct relationship between the linear combinations of the original constraint system and the transformed one:

**Lemma 7** (Mixed Column Transformation Implications)**.** *Let $Ax \leq b$ be a constraint system. Let $V \in \mathbb{Q}^{n\times n}$ be a mixed column transformation matrix. Let $Ax \leq b$ imply $h^T x \leq g$. Then $AVz \leq b$ implies $h^T Vz \leq g$.*

*Proof.* By Lemma 2, $Ax \leq b$ implies $h^T x \leq g$ iff there exists a non-negative linear combination $y \in \mathbb{Q}^n$ such that $y \geq 0$, $y^T A = h^T$ and $y^T b \leq g$. Multiplying $y^T A = h^T$ with $V$ results in $y^T AV = h^T V$ and thus $y$ is also the non-negative linear combination of inequalities $AVz \leq b$ that results in $h^T Vz \leq g$. □

**Corollary 2** (Mixed Column Transformation Certificates)**.** *Let $Ax \leq b$ be a constraint system. Let $V \in \mathbb{Q}^{n\times n}$ be a mixed column transformation matrix. Then $y$ is a certificate of unsatisfiability for $Ax \leq b$ iff it is one for $AVz \leq b$.*

Now we only need a mixed column transformation matrix $V$ for every coefficient matrix $A$ such that $H = AV$ is lower triangular with gaps. One such matrix $V$ is the one that transforms $A$ into *Mixed-Echelon-Hermite normal form*:

**Definition 6** (Mixed-Echelon-Hermite Normal Form [12])**.** *A matrix $H \in \mathbb{Q}^{m\times n}$ is in* Mixed-Echelon-Hermite normal form *if*

$$H = \begin{pmatrix} E & 0^{r\times(n_1-r)} & 0^{r\times n_2} \\ E' & 0^{(m-r)\times(n_1-r)} & H' \end{pmatrix},$$

*where $E$ is an $r \times r$ identity matrix (with $r \leq n_1$), $E' \in \mathbb{Q}^{(m-r)\times r}$, and $H' \in \mathbb{Q}^{(m-r)\times n_2}$ is a matrix in hermite normal form, i.e., a lower triangular matrix without gaps, where each entry $h'_{piv(H',j)k}$ in the row $piv(H', j)$ is non-negative and smaller than $h'_{piv(H',j)j}$).*

The following proof for the existence of the Mixed-Echelon-Hermite normal form is constructive and presents the Mixed-Echelon-Hermite transformation.

**Lemma 8** (Mixed-Echelon-Hermite Transformation)**.** *Let $A \in \mathbb{Q}^{m\times n}$ be a matrix, where the upper left $r \times n_1$ submatrix has the same rank $r$ as the complete left $m \times n_1$ submatrix. Then there exists a mixed transformation matrix $V \in \mathbb{Q}^{n\times n}$ such that $H = AV$ is in* Mixed-Echelon-Hermite normal form*.*

*Proof.* Proof from [12] with slight modifications so it also works for singular matrices. We subdivide $A$ into

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$$

such that $A_{11} \in \mathbb{Q}^{r\times n_1}$, $A_{12} \in \mathbb{Q}^{r\times n_2}$, $A_{21} \in \mathbb{Q}^{m-r\times n_1}$, and $A_{21} \in \mathbb{Q}^{m-r\times n_2}$. Then we bring $A_{11}$ with an invertible matrix $V_{11} \in \mathbb{Q}^{n_1\times n_1}$ into reduced echelon column form $H_{11} = (E\ 0^{r\times(n_1-r)}) = A_{11}V_{11}$, where $E$ is an $r \times r$ identity matrix. We get $V_{11}$ and $H_{11}$ by using Bareiss algorithm instead of the better known Gaussian elimination as it is polynomial in time [2].[3] Note that the last $n_1 - r$ columns of $H_{21} = (H'_{21}\ 0^{(m-r)\times(n_1-r)}) = A_{21}V_{11}$ are also zero because all rows in $A_{21}$ are linear dependent of $A_{11}$ (due to the rank). Next we notice that

$$A_{12} - A_{11}V_{11} \begin{pmatrix} A_{12} \\ 0^{(n_1-r)\times n_2} \end{pmatrix} = A_{12} - (E\ 0^{r\times(n_1-r)}) \begin{pmatrix} A_{12} \\ 0^{(n_1-r)\times n_2} \end{pmatrix} = 0^{r\times n_2}$$

so we can reduce the upper right submatrix $A_{12}$ to zero by adding multiples of the $n_1$ columns with rational variables to the $n_2$ columns with integer variables. However, this also transforms the lower right submatrix $A_{22}$ into

$$H'_{22} = A_{22} - A_{21}V_{11} \begin{pmatrix} A_{12} \\ 0^{(n_1-r)\times n_2} \end{pmatrix}.$$

Finally, we transform this new submatrix $H'_{22}$ into hermite normal form $H_{22}$ via the algorithm of Kannan and Bachem (or a similar polynomial time algorithm).[3] This algorithm also returns a unimodular matrix $V_{22} \in \mathbb{Z}^{n_2\times n_2}$ such that $H_{22} = H'_{22}V_{22}$. To summarize: our total mixed transformation matrix is

$$V = \begin{pmatrix} V_{11} & -V_{11} \cdot \begin{pmatrix} A_{12} \\ 0^{(n_1-r)\times n_2} \end{pmatrix} \cdot V_{22} \\ 0^{n_2\times n_1} & V_{22} \end{pmatrix} \text{ and } H = AV = \begin{pmatrix} H_{11} & 0^{r\times n_2} \\ H_{21} & H_{22} \end{pmatrix}.$$

□

[3]We do actually use less efficient, Gaussian-elimination-based transformations in our own implementation [7]. The reason is that these transformations are incrementally efficient. Our experiments show that the transformation cost still remains negligible in practice.

It is not possible to transform every matrix $A \in \mathbb{Q}^{m \times n}$ into Mixed-Echelon-Hermite normal form. We have to restrict ourselves to matrices, where the upper left $r \times n_1$ submatrix has the same rank $r$ as the complete left $m \times n_1$ submatrix. However, this is very easy to accomplish for a system of linear mixed arithmetic constraints $l \leq Ax \leq u$. The reason is that the order of inequalities does not change the set of satisfiable solutions. Hence, we can swap the inequalities and, thereby, the rows of $A$ until its upper left $r \times n_1$ submatrix has the desired form. This also means that there are usually multiple possible inequality orderings that each have their own Mixed-Echelon-Hermite normal form $H$.

To conclude this section: whenever we have a double-bounded constraint system $l \leq Dx \leq u$, we can transform it (after some row swapping) into an equisatisfiable system $l \leq Hy \leq u$ where $H = DV$ is in Mixed-Echelon-Hermite normal form and $Vy = x$. Since $H$ is also a lower triangular matrix with gaps, branch-and-bound terminates on $l \leq Hy \leq u$ with a mixed solution $t$ or it will return unsatisfiable (Corollary 1). Moreover, we can convert any mixed solution $t$ for $l \leq Hy \leq u$ into a mixed solution $s$ for $l \leq Dx \leq u$ by setting $s := Vt$. Hence, we have a complete algorithm for double-bounded constraint systems.

# 4 Double-Bounded Reduction

In the previous Section, we have shown how to solve a double-bounded constraint system. Now we show how to reduce any constraint system $A'x \leq b'$ to an equisatisfiable double-bounded system $l \leq Dx \leq u$. Moreover, we explain how to take any solution of $l \leq Dx \leq u$ and turn it into a solution for $A'x \leq b'$.

As the first step of our reduction, we reformulate the constraint system into a so called *split system*:

**Definition 7** (Split System)**.** $(Ax \leq b) \cup (l \leq Dx \leq u)$ *is a* split system *if: (i) all directions are unbounded in* $Ax \leq b$*; (ii) all row vectors* $a_i$ *from* $A$ *are also unbounded in* $(Ax \leq b) \cup (l \leq Dx \leq u)$*. Moreover, we call* $Ax \leq b$ *the unbounded part and* $l \leq Dx \leq u$ *the bounded part of the split system.*

A split system consists of an unbounded part $Ax \leq b$ that is guaranteed to have (infinitely many) integer solutions (see Lemma 3) and a double-bounded part $l \leq Dx \leq u$. Any constraint system can be brought into the above form. We just have to move all unbounded inequalities into the unbounded part and all bounded inequalities into the bounded part.

**Lemma 9** (Split Equivalence). *Let $A'x \leq b'$ be a constraint system with $A' \in \mathbb{Q}^{m \times n}$. Then there exists an equivalent split system $(Ax \leq b) \cup (l \leq Dx \leq u)$ where: (i) $A \in \mathbb{Q}^{m_1 \times n}$ and $D \in \mathbb{Q}^{m_2 \times n}$ so that $m_1 + m_2 = m$; (ii) all rows $d_i^T$ of $D$ and $a_k^T$ of $A$ appear as rows in $A'$; and (iii) $Dx \leq u$ implies $l \leq Dx$.*

*Proof.* For (i), (ii), and the equivalence, it is enough to move all bounded inequalities $a_i'^T x \leq b_i'$ of $A'x \leq b'$ into $Dx \leq u$ and all unbounded inequalities into $Ax \leq b$. For (iii), we assume for a contradiction that $Dx \leq u$ does not imply $l_i \leq d_i^T x$ but $(Dx \leq u) \cup (Ax \leq b)$ does. By Lemma 2, this means that there exists a $y \in \mathbb{Q}^{m_2}$ with $y \geq 0^{m_2}$ and a $z \in \mathbb{Q}^{m_1}$ with $z \geq 0^{m_1}$ so that $y^T D + z^T A = -d_i^T$ and $y^T u + z^T b \leq -l_i$. We also know that there exists a $z_k > 0$ because $Dx \leq u$ alone does not imply $l_i \leq d_i^T x$. We use this fact to reformulate $y^T D + z^T A = -d_i^T$ into $-a_k^T = \frac{1}{z_k}\left[y^T D + d_i^T + \sum_{j=1, j\neq k}^{m_1} z_j a_j^T\right]$, and use the bounds of the inequalities in $Dx \leq u$ and $Ax \leq b$ to derive a lower bound for $a_k^T x$: $-a_k^T x \leq \frac{1}{z_k}\left[y^T u + u_i + \sum_{j=1, j\neq k}^{m_1} z_j b_j\right]$. Hence, $a_k^T$ is bounded in $A'x \leq b'$ and we have our contradiction. $\square$

The above Lemma also shows that the bounded part of a constraint system is self-contained, i.e., a constraint system implies that a direction is bounded if and only if its bounded part does. The actual difficulty of reformulating a system into a split system is not the transformation per se, but finding out which inequalities are bounded or not. There are many ways to detect whether an inequality is bounded by a constraint system. Most work even in polynomial time. For instance, solving the linear rational optimization problem "minimize $a_i^T x$ such that $Ax \leq b$" returns $-\infty$ if $a_i$ is unbounded, $\infty$ if $Ax \leq b$ has no rational solution, and the optimal lower bound $l_i$ for $a_i^T x$ otherwise. However, it still requires us to solve $m$ linear optimization problems.

A, in our opinion, more efficient alternative is based on our previously presented algorithm for finding equality bases [9]. This is due to the following relationship between bounded directions and equalities:

**Lemma 10** (Bounds and Equalities). *Let $\mathcal{Q}(Ax \leq b) \neq \emptyset$. Then $h$ is bounded in $Ax \leq b$ iff $Ax \leq 0^m$ implies that $h^T x = 0$.*

*Proof.* By Definition 1, $h$ is bounded in $Ax \leq b$ means that there exists $l, u \in \mathbb{Q}$ such that $Ax \leq b$ implies $h^T x \leq u$ and $-h^T x \leq -l$. By Lemma 2, this is equivalent to: there exist $l, u \in \mathbb{Q}$, $y, z \in \mathbb{Q}^m$ with $y, z \geq 0^m$, and $y^T A = h^T = -z^T A$ so that $y^T b \leq u$ and $z^T b \leq -l$. Symmetrically, $Ax \leq 0$ implies that $h^T x = 0$ is equivalent to: there exist a $y, z \in \mathbb{Q}^m$ with $y, z \geq 0^m$

and $y^T A = h^T = -z^T A$ so that $y^T 0^m \leq 0$ and $z^T 0^m \leq 0$. Since $u$ and $l$ only have to exists, we can trivially choose them as $u := y^T b$ and $l := -z^T b$. This means that $y^T b \leq u$, $z^T b \leq -l$, $y^T 0^m \leq 0$, and $z^T 0^m \leq 0$ are all trivially satisfied by any pair of linear combinations $y, z \in \mathbb{Q}^m$ with $y, z \geq 0^m$ such that $y^T A = h^T = -z^T A$. Hence, the two definitions are equivalent and our lemma holds. □

It is easy and efficient to compute an equality basis for $Ax \leq 0^m$ and to determine with it the inequalities in $Ax \leq b$ that are bounded [9]. The only disadvantage towards the optimization approach is that we do not derive an optimal lower bound $l$ for the inequalities. This is no problem because only the existence of lower bounds is relevant and not the actual bound values.

In a split system $(Ax \leq b) \cup (l \leq Dx \leq u)$, the unbounded part is actually inconsequential to the rational/mixed satisfiability of the system. It may reduce the number of rational/mixed solutions, but it never removes them all. Hence, $(Ax \leq b) \cup (l \leq Dx \leq u)$ is equisatisfiable to just $l \leq Dx \leq u$. We first show this equisatisfiability for the rational case:

**Lemma 11** (Rational Extension). *Let $(Ax \leq b) \cup (l \leq Dx \leq u)$ be a split system. Let $s \in \mathbb{Q}^n$ be a rational solution to the bounded part $l \leq Dx \leq u$ such that $Ds = g$, where $g \in \mathbb{Q}^{m_2}$. Then $(Ax \leq b) \cup (Dx = g)$ has a solution $s'$.*

*Proof.* Assume for a contradiction that $(Ax \leq b) \cup (Dx = g)$ has no solution. By Lemma 1, this means that there exist a $y \in \mathbb{Q}^{m_1}$ with $y \geq 0^{m_1}$ and $z, z' \in \mathbb{Q}^{m_2}$ with $z, z' \geq 0^{m_2}$ such that $y^T A + z^T D - z'^T D = 0^n$ and $y^T b + z^T g - z'^T g < 0$. Since $Dx = g$ is satisfiable by itself, there must exist a $y_i > 0$. Now we use this fact to reformulate the equation $y^T A + z^T D - z'^T D = 0^n$ into

$$-a_i^T = \frac{1}{y_i}\left[\left(\sum_{j=1 j\neq i}^{m_1} y_j a_j^T\right) + z^T D - z'^T D\right],$$

from which we deduce a lower bound for $a_i^T x$ in $(Ax \leq b) \cup (l \leq Dx \leq u)$:

$$-a_i^T x \leq \frac{1}{y_i}\left[\left(\sum_{j=1 j\neq i}^{m_1} y_j b_j\right) + z^T u - z'^T l\right].$$

Therefore, $a_i$ is bounded in $(Ax \leq b) \cup (l \leq Dx \leq u)$, which is a contradiction. □

Note that the bounded part $l \leq Dx \leq u$ of a split system can still have unbounded directions (not inequalities). Some of these unbounded directions in $l \leq Dx \leq u$ are the orthogonal directions to the row vectors $d_i$, i.e., vectors $v_j \in \mathbb{Z}^n$ such that $d_i^T v_j = 0$ for all $i \in \{1, \ldots, m_2\}$. This also means that the existence of one mixed solution $s \in (\mathbb{Q}^{n_1} \times \mathbb{Z}^{n_2})$ and one unbounded direction proves the existence of infinitely many mixed solutions. We just need to follow the orthogonal directions, i.e., for all $\lambda \in \mathbb{Z}$, $s' = \lambda \cdot v_j + s$ is also a mixed solution because $d_i^T s' = \lambda \cdot d_i^T v_j + d_i^T s = d_i^T s$. In the next two steps, we prove that $Ax \leq b$ cannot cut off all of these orthogonal solutions because it is completely unbounded. The first step proves that $Ax \leq b$ remains absolutely unbounded even if we settle on one set of orthogonal solutions, i.e., enforce $Dx = Ds$ for some solution $s$.

**Lemma 12** (Persistence of Unboundedness)**.** *Let $(Ax \leq b) \cup (l \leq Dx \leq u)$ be a split system. Let $s \in \mathbb{Q}^n$ be a rational solution for $l \leq Dx \leq u$ such that $Ds = g$ (with $g \in \mathbb{Q}^{m_2}$). Then all row vectors $a_i$ from $A$ are still unbounded in $(Ax \leq b) \cup (Dx = g)$.*

*Proof.* By Lemma 11, $(Ax \leq b) \cup (Dx = g)$ has at least a rational solution $s^*$. Moreover, $(Ax \leq 0) \cup (Dx = 0)$ does not imply $a_i^T x = 0$ because of Lemma 10 and the assumption that the row vectors $a_i$ from $A$ are unbounded in $(Ax \leq b) \cup (l \leq Dx \leq u)$. In reverse, $(Ax \leq b) \cup (Dx = g)$ having a real solution, $(Ax \leq 0) \cup (Dx = 0)$ does not imply $a_i^T x = 0$, and Lemma 10 prove together that the row vectors $a_i$ from $A$ are also unbounded in $(Ax \leq b) \cup (Dx = g)$. □

The next step proves how to extend the mixed solution from the bounded part to the complete system with the help of the Mixed-Echelon-Hermite normal form and the absolute unboundedness of $Ax \leq b$.

**Lemma 13** (Mixed Extension)**.** *Let $(Ax \leq b) \cup (l \leq Dx \leq u)$ be a split system. Let $s \in (\mathbb{Q}^{n_1} \times \mathbb{Z}^{n_2})$ be a mixed solution for $l \leq Dx \leq u$. Then $(Ax \leq b) \cup (l \leq Dx \leq u)$ has a mixed solution $s'$.*

*Proof.* Let $g = Ds$. Without loss of generality we assume that the upper left $r \times n_1$ submatrix of $D$ has the same rank $r$ as the complete left $m_1 \times n_1$ submatrix of $D$. (Otherwise, we just reorder the rows accordingly.) Therefore, there exists a mixed column transformation matrix $V$ such that $H = DV$ is in mixed-echelon-hermite normal form (see Lemma 8). By Lemma 6, there exists a mixed vector $t \in (\mathbb{Q}^{n_1} \times \mathbb{Z}^{n_2})$ such that $s = Vt$ and $t$ is a mixed-solution to $l \leq Hy \leq u$ as well as $Hy = g$. Let $\mathcal{U}$ be the set of indices with 0 columns in $H$ and $\mathcal{B}$ the column indices with bounded variables. Then the equation system $(Hy = g)$ fixes each variable $y_j$ with $j \in \mathcal{B}$ to the value $t_j$ because $H$ is lower triangular with gaps. Hence, $((AV)y \leq b) \cup (Hy = g)$ is equivalent to

$$A\left[\sum_{j\in\mathcal{U}}\begin{pmatrix} v_{1j} \\ \vdots \\ v_{nj}\end{pmatrix}\cdot y_j\right]\leq b-A\left[\sum_{j\in\mathcal{B}}\begin{pmatrix} v_{1j} \\ \vdots \\ v_{nj}\end{pmatrix}\cdot t_j\right]. \tag{1}$$

Due to Lemma 12 and 7, all directions are unbounded in (1). This means (1) has an integer solution (Lemma 3) assigning each variable $y_j$ with $j \in \mathcal{U}$ to a $t'_j \in \mathbb{Z}$. (Can be computed via the unit cube test [11]). We extend this solution to all variables $y$ by setting $t'_j := t_j$ for $j \in \mathcal{B}$ and we have a mixed solution $t' \in (\mathbb{Q}^{n_1} \times \mathbb{Z}^{n_1})$ for $((AV)y \leq b) \cup (l \leq Hy \leq u)$. Hence, we have via Lemma 6 a mixed solution $s' \in (\mathbb{Q}^{n_1} \times \mathbb{Z}^{n_2})$ for $(Ax \leq b) \cup (l \leq Dx \leq u)$ with $s' = Vt'$. □

**Corollary 3** (Double-Bounded Reduction). *The split system* $(Ax \leq b) \cup (l \leq Dx \leq u)$ *is mixed equisatisfiable to* $(l \leq Dx \leq u)$*.*

## 5 Experiments

We integrated the Double-Bounded reduction and the Mixed-Echelon-Hermite transformation into our own theory solver *SPASS-IQ v0.2*[4] and ran it on four families of newly constructed benchmarks[4]. Once with the transformations turned on (*SPASS-IQ*) and once with the transformations turned off (*SPASS-IQ-Off*). If SPASS-IQ encounters a system $Ax \leq b$ that is not explicitly bounded, i.e., where not all variables have an explicit upper and lower bound, then it computes an equality basis for $Ax \leq 0^m$. This basis is used to determine whether the system is implicitly bounded, absolutely unbounded or partially bounded, as well as which of the inequalities are bounded. Our solver only applies our two transformations if the problem is partially unbounded. The resulting equisatisfiable but bounded problem is then solved via branch-and-bound. The other two cases, absolutely unbounded and implicitly bounded, are solved respectively via the unit cube test [11] and branch-and-bound on the original system. Our solver also converts any mixed solutions from the transformed system into mixed solutions for the original system following the proof of Lemma 13. Rational conflicts are converted between the two systems by using Corollary 2.

We tried to restrict our benchmarks to partially unbounded problems since we only apply our transformations on those problems. We even found some partially unbounded problems in the SMT-LIB benchmarks for QF_LIA (quantifier free linear arithmetic). However, there are not many such bench-

[4]Available on `http://www.spass-prover.org/spass-iq`

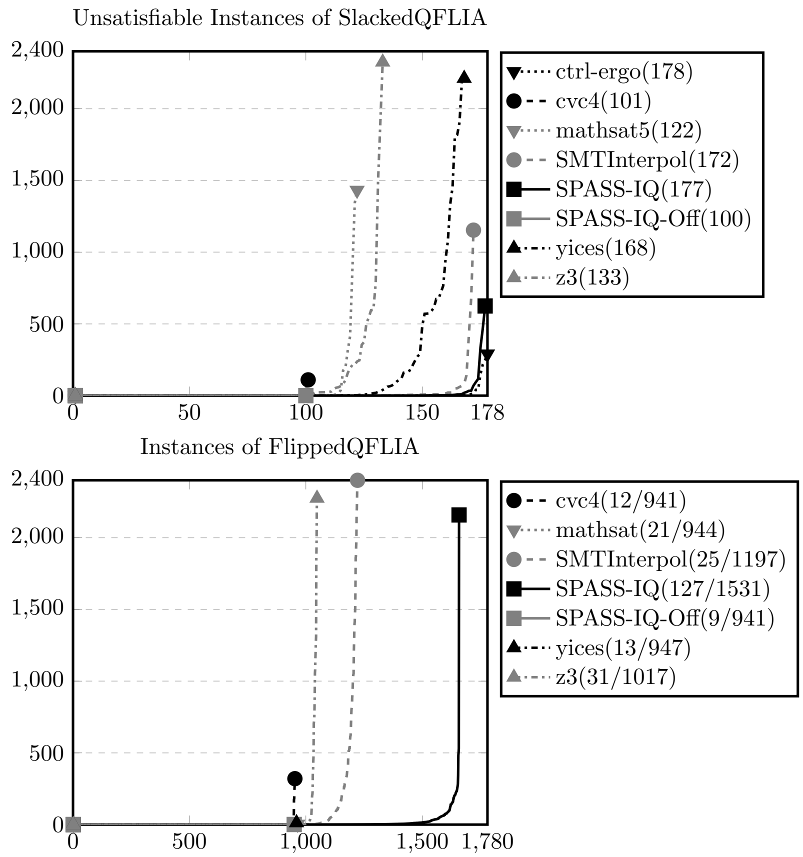


Figure 1: horizontal axis: # of solved instances; vertical axis: time (seconds)

marks: only one in *CAV-2009*, five in *cut_lemmas*, and three in *slacks*. So we created in addition four new benchmark families:

*SlackedQFLIA*: are linear integer benchmarks based on the SMT-LIB classes *CAV-2009* [16], *cut_lemmas* [20], and *dillig* [16]. We simply took all of the unsatisfiable benchmarks and replaced in them all variables $x$ with $x_+ - x_-$ where $x_+$ and $x_-$ are two new variables such that $x_+, x_- \geq 0$. This transformation, called slacking, is equisatisfiable and the slacked version of the *dillig*-benchmarks, called *slacked* [21], is already in the SMT-LIB. Slacking turns any unsatisfiable problem into a partially unbounded one. Hence, all problems in *SlackedQFLIA* are partially unbounded. Slacking is commonly used to integrate absolute values into linear systems or for solvers that require non-negative variables [23].

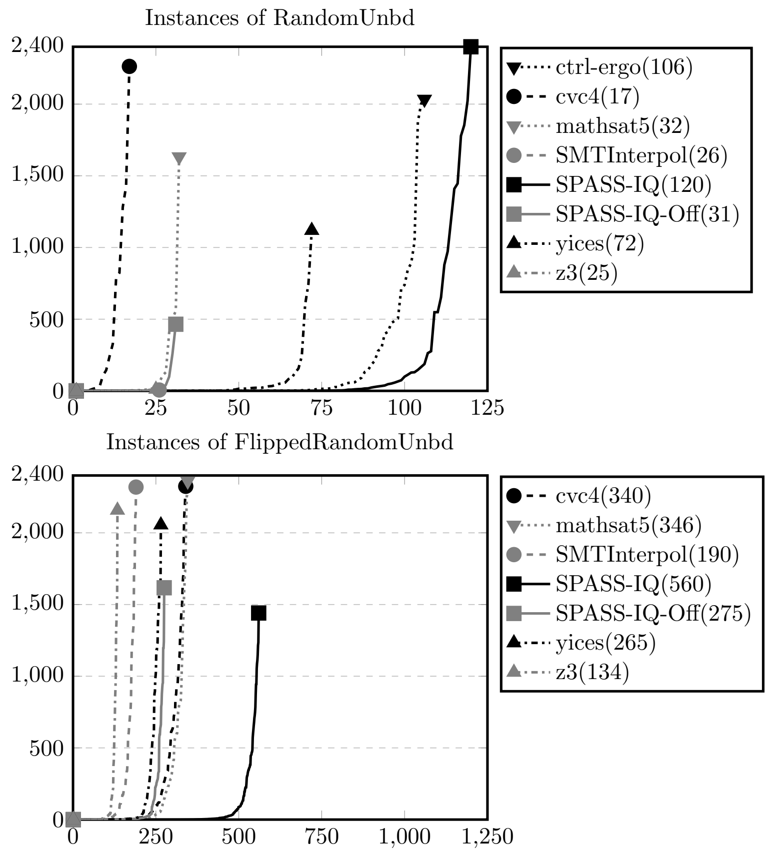


Figure 2: horizontal axis: # of solved instances; vertical axis: time (seconds)

*RandomUnbd*: are linear integer benchmarks that are all partially unbounded and satisfiable with 10, 25, 50, 75, and 100 variables. All problems are randomly created via a sagemath script[4].

*FlippedQFLIA* and *FlippedRandomUnbd*: are linear mixed benchmarks that are all partially unbounded. They are based on *SlackedQFLIA* and *RandomUnbd*. We constructed them by first copying ten versions of the integer benchmarks and then randomly flipping the type of some of the variables to rational (probability of 20%). Some of the flipped instances of *SlackedQFLIA* became satisfiable.

We compared our solver with some of the state-of-the-art SMT solvers currently available for linear mixed arithmetic: *cvc4-1.5* [3], *mathsat5-5.1* [14], *SMTInterpol 2.1-335-g4c543a5* [13], *yices2.5.4* [17], and *z3-4.6.0* [15]. Most of these solvers employ a branch-and-bound approach with an underlying dual simplex solver [18], which is also the basis for our own solver. As far as we are aware, none of them employ any techniques that guarantee termination.

SMTInterpol extends branch-and-bound via the cuts from proofs approach, which uses the Mixed-Echelon-Hermite transformation to find more versatile branches and cuts [12]. Although the procedure is not complete, the similarities to our own approach make an interesting comparison. Actually, the Double-Bounded reduction alone would be sufficient to make SMTInterpol terminating since it already builds branches via a Mixed-Echelon-Hermite transformation.

We also compared our solver with the *ctrl-ergo* solver [5] although it is restricted to pure integer arithmetic. Ctrl-ergo is complete over linear integer arithmetic and uses the most similar approach to our transformations that we found in the literature. It dynamically eliminates one linear independent bounded direction at a time via transformation. The disadvantages of the dynamic approach are that it is very restrictive and does not leave enough freedom to change strategies or to add complementing techniques. Moreover, ctrl-ergo uses this transformation approach for all problems and not only the partially unbounded ones, which sometimes leads to a massive overhead on bounded problems.

For the experiments, we used a Debian Linux cluster and allotted to each problem and solver combination 2 cores of an Intel Xeon E5620 (2.4 GHz) processor, 4 GB RAM, and 40 minutes. The only solver benefiting from multiple cores is SMTInterpol. The plots in Figures 1 and 2 depict the results of the different solvers. In the legends of the plots, the numbers behind the solver names are the number of solved instances. For *FlippedQFLIA*, there are two numbers to indicate the number of satisfiable/unsatisfiable instances solved. This is only necessary for *FlippedQFLIA* because it is the only tested benchmark family with satisfiable and unsatisfiable instances. (We verified that the results match if two solvers solved the same problem.)

Although our solver could not solve all problems (due to time and memory limits) it was still able to solve more problems than the other solvers. It was also faster on most instances than the other solvers. In some of the unsatisfiable, partially unbounded benchmarks ctrl-ergo is better than SPASS-IQ. This is due to its conflict focused, dynamic approach. For the same reason, ctrl-ergo is slower on the satisfiable, partially unbounded benchmarks. Only SPASS-IQ, ctrl-ergo, and yices solved all of the ten original SMT-LIB benchmarks that are partially unbounded, though the complete methods were still a lot faster (SPASS-IQ took 23s, ctrl-ergo took 42s, and yices took 1273s). On one of these benchmarks, 20-14.slacks.smt2 from *slacks*, all other solvers seem to diverge. Another interesting result of our experiments is that relaxing some integer variables to rational variables seems to make the problems harder instead of easier. We expected this for our transformations because the resulting systems become more complex and less sparse, but it is also true for the other solvers. The reason might be that bound refinement, a technique used in most branch-and-bound implementations, is less effective on mixed problems.

The time SPASS-IQ needs to detect the bounded inequalities and to apply our transformations is negligible. This is even true for the implicitly bounded problems we tested. As mentioned before, we do not have to apply our transformations to terminate on bounded problems. This is also the only advantage we gain from detecting that a problem is implicitly bounded. Since there is no noticeable difference in the run time, we do not further elaborate the results on bounded problems, e.g. with graphs.

An actual disadvantage of our approach is that the Mixed-Echelon-Hermite transformation increases the density of the coefficient matrix as well as the absolute size of the coefficients. Both are important factors for the efficiency of the underlying simplex solver. Moreover, SPASS-IQ reaches more often the memory limit than the time limit because it needs a (too) large number of branches and bound refinements before terminating.

## 6 Conclusion

We have presented the Mixed-Echelon-Hermite transformation (Lemma 8) and the Double-Bounded reduction (Lemma 9 & Corollary 3). We have shown that both transformations together turn any constraint system into an equisatisfiable system that is also bounded (Lemma 4). This is sufficient to make branch-and-bound, and many other linear mixed decision procedures, complete and terminating. We have also shown how to convert certificates of (un)satisfiability efficiently between the transformed and original systems (Corollary 2 & Lemma 13). Moreover, experimental results on partially unbounded benchmarks show that our approach is also efficient in practice.

Our approach can be nicely combined with the extensive branch-and-bound framework and its many extensions, where other complete techniques cannot be used in a modular way [5, 8]. For future research, we plan to test our transformations in combination with other algorithms, e.g., cuts from proofs, or as a dynamic version similar to the approach used by ctrl-ergo [5]. We also want to test whether our transformations are useful preprocessing steps for select constraint system classes that are bounded.

## References


[1] E. Althaus, E. Kruglov, and C. Weidenbach. Superposition modulo linear arithmetic SUP(LA). In *FroCoS 2009*, volume 5749 of *LNCS*. Springer, 2009.

[2] E. H. Bareiss. Sylvester's identity and multistep integer-preserving gaussian elimination. *Mathematics of Computation*, 22(103):565–578, 1968.

[3] C. Barrett, C. Conway, M. Deters, L. Hadarean, D. Jovanović, T. King, A. Reynolds, and C. Tinelli. CVC4. In *CAV*, volume 6806 of *LNCS*. 2011.

[4] P. Baumgartner and U. Waldmann. Hierarchic Superposition with Weak Abstraction. In *CADE-24*, volume 7898 of *LNAI*. Springer, 2013.

[5] F. Bobot, S. Conchon, E. Contejean, M. Iguernelala, A. Mahboubi, A. Mebsout, and G. Melquiond. A simplex-based extension of fourier-motzkin for solving linear integer arithmetic. In *IJCAR 2012*, volume 7364 of *LNCS*, 2012.

[6] S. Boyd and L. Vandenberghe. *Convex Optimization*. CUP, 2004.

[7] M. Bromberger. A reduction from unbounded linear mixed arithmetic problems into bounded problems. *ArXiv e-prints*, abs/1804.07703, 2015.

[8] M. Bromberger, T. Sturm, and C. Weidenbach. Linear integer arithmetic revisited. In *CADE-25*, volume 9195 of *LNCS*. 2015.

[9] M. Bromberger and C. Weidenbach. Computing a complete basis for equalities implied by a system of LRA constraints. In *SMT 2016*, 2016.

[10] M. Bromberger and C. Weidenbach. Fast cube tests for LIA constraint solving. In *IJCAR 2016*, volume 9706 of *LNCS*. 2016.

[11] M. Bromberger and C. Weidenbach. New Techniques for Linear Arithmetic: Cubes and Equalities. *Formal Methods in System Design*, 51(3), 2017.

[12] J. Christ and J. Hoenicke. Cutting the mix. In *CAV*, volume 6806 of *LNCS*. Springer, 2015.

[13] J. Christ, J. Hoenicke, and A. Nutz. SMTInterpol: an interpolating SMT solver. In *SPIN 2012*, volume 7385 of *LNCS*. Springer, 2012.

[14] A. Cimatti, A. Griggio, B. Schaafsma, and R. Sebastiani. The MathSAT5 SMT Solver. In *TACAS*, volume 7795 of *LNCS*, 2013.

[15] L. de Moura and N. Bjørner. Z3: An efficient SMT solver. In *Tools and Algorithms for the Construction and Analysis of Systems*, volume 4963 of *LNCS*. 2008.

[16] I. Dillig, T. Dillig, and A. Aiken. Cuts from proofs: A complete and practical technique for solving linear inequalities over integers. In *CAV*, volume 5643 of *LNCS*. 2009.

[17] B. Dutertre. Yices 2.2. In *CAV 2014*, volume 8559 of *LNCS*, 2014.

[18] B. Dutertre and L. de Moura. A fast linear-arithmetic solver for DPLL(T). In *CAV*, volume 4144 of *LNCS*. 2006. Extended version: Integrating simplex with DPLL(T). Tech. rep., CSL, SRI INTERNATIONAL (2006).

[19] A. Fietzke and C. Weidenbach. Superposition as a decision procedure for timed automata. *Mathematics in Computer Science*, 6(4), 2012.

[20] A. Griggio. A practical approach to satisfiability modulo linear integer arithmetic. *JSAT*, 8(1/2), 2012.

[21] D. Jovanović and L. de Moura. Cutting to the chase. *JAR*, 51(1), 2013.

[22] C. H. Papadimitriou. On the complexity of integer programming. *J. ACM*, 28(4), Oct. 1981.

[23] A. Schrijver. *Theory of Linear and Integer Programming.* John Wiley & Sons, Inc., New York, NY, USA, 1986.