Journal on Satisfiability, Boolean Modeling and Computation 3 (2007) 67-90

On Solving Boolean Combinations of UTVPI Constraints

Sanjit A. Seshia sseshia@eecs.berkeley.edu
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley

K. Subramani ksmani@csee.wvu.edu
Lane Department of Computer Science and Electrical Engineering
West Virginia University

Randal E. Bryant randy.bryant@cs.cmu.edu
School of Computer Science
Carnegie Mellon University

Abstract

We consider the satisfiability problem for Boolean combinations of unit two variable per inequality (UTVPI) constraints. A UTVPI constraint is linear constraint containing at most two variables with non-zero coefficients, where furthermore those coefficients must be either -1 or 1. We prove that if a satisfying solution exists, then there is a solution with each variable taking values in $[-n \cdot (b_{\max}+1),\ n \cdot (b_{\max}+1)]$, where n is the number of variables, and $b_{\max}$ is the maximum over the absolute values of constants appearing in the constraints. This solution bound improves over previously obtained bounds by an exponential factor. Our result can be used in a finite instantiation-based approach to deciding satisfiability of UTVPI formulas. An experimental evaluation demonstrates the efficiency of such an approach. One of our key results is to show that an integer point inside a UTVPI polyhedron, if one exists, can be obtained by rounding a vertex. As a corollary of this result, we also obtain a polynomial-time algorithm for approximating optima of UTVPI integer programs to within an additive factor.

KEYWORDS: *unit two variable per inequality constraints, Boolean satisfiability, automated theorem proving, integer linear programming, decision procedures, constraint satisfaction, verification, optimization*

Submitted November 2006; revised March 2007; published July 2007

1. Introduction

A *unit two variable per inequality* (UTVPI) constraint is a special kind of linear constraint having at most two variables with non-zero coefficients, where furthermore those coefficients must be either -1 or 1. For this paper, we restrict our focus to UTVPI constraints over integer variables, also called *generalized 2SAT constraints*. The variables are not required to have finite upper or lower bounds. Useful optimization problems, such as the minimum vertex cover and the maximum independent set problems, can be modeled using UTVPI constraints, and some applications of constraint logic programming and automated theorem proving also generate UTVPI constraints (e.g., see [18, 1]).

A *UTVPI formula* is a Boolean combination of UTVPI constraints. In this paper, we consider the problem of checking the satisfiability of UTVPI formulas. It is easily

seen that this problem is NP-hard. However, the special case of checking satisfiability of a conjunction of UTVPI constraints (i.e., finding a feasible integer point in a UTVPI polyhedron) can be solved in polynomial time; for example, a modified version of Fourier-Motzkin elimination [9, 33] (reviewed in Section 2) runs in $O(n^3)$ time.

A simple approach to checking the satisfiability of a UTVPI formula employs a combination of Boolean satisfiability solving and linear constraint solving. Truth values are assigned to linear constraints so that the UTVPI formula is satisfied. Each such truth assignment corresponds to a UTVPI polyhedron. If this polyhedron has a feasible integer point, that point satisfies the original UTVPI formula as well. If not, another truth assignment must be found. Given a UTVPI formula ϕ with m constraints and n variables, and assuming that integer feasibility is checked using the afore-mentioned modified Fourier-Motzkin elimination algorithm, this simple approach has a worst-case running time of $O(2^m \cdot n^3)$.[1.]

In this paper, we prove that a satisfying solution exists for a UTVPI formula ϕ if and only if there is a solution to ϕ with each variable taking values in the finite range $[-n\cdot(b_{\max}+1), n\cdot(b_{\max}+1)]$, where n is the number of variables in ϕ, and $b_{\max}$ is the maximum over the absolute values of constant terms in the constraints. That such a bounded solution exists is not surprising, since satisfiability solving of UTVPI formulas is in NP, as follows from integer linear programming being in NP [3, 34, 19, 27]. However, the previously best known solution bounds (in the afore-referenced papers) are $\Omega(n^2 \cdot (b_{\max}+1) \cdot 2^n)$. In particular, our result eliminates the 2^n term, thereby exponentially reducing the solution bound.

Our result can be used to check satisfiability of UTVPI formulas in worst-case time $2^{O(n \log d)}$ where $d = 2n(b_{\max}+1)+1$, by encoding each integer variable with $\log d$ Boolean variables. On the theoretical side, this yields a more efficient satisfiability checker for highly over-constrained formulas, where $m = \Omega(n \cdot \log d)$.[2.] Such formulas are often generated in program analysis and hardware verification.

A key step in our proof is to show that for a UTVPI polyhedron, if a feasible integer point exists, then one exists *within a unit hypercube centered at any minimal face solution* (vertex). Apart from enabling us to prove the above solution bound, this *rounding* result also throws light on why a Simplex-based algorithm might find it easier to find an integer solution to UTVPI constraints as compared to arbitrary linear constraints. Simplex-based algorithms usually start with a basic feasible solution to a relaxation (a minimal face solution) and then search for an integer solution using branch-and-bound, cutting planes, or a combination (branch-and-cut). Many of these search techniques are effective for 0-1 integer programs. Our rounding result shows that the search for an integer solution from a minimal face solution of a UTVPI polyhedron is similar to that for finding an integer solution to a 0-1 program, and indicates why common search techniques might be effective.

As a corollary of this rounding result, we obtain a polynomial-time algorithm for approximating optima to an additive factor in UTVPI integer programs.

Our theoretical results are validated by an experimental evaluation on UTVPI formulas from the SMT-LIB repository as well as randomly generated UTVPI formulas, which

1. Assuming the trivial worst-case bound of $O(2^N)$ for checking satisfiability of a Boolean formula in N variables.
2. For a conjunction of UTVPI constraints, m is $O(n^2)$, since one can eliminate redundant constraints. However, for an arbitrary Boolean combination, this is not the case.

shows that a finite instantiation-based decision procedure based on our approach can be competitive with state-of-the-art decision procedures.

1.1 Related Work

There has been much previous work on integer programming with two variables per inequality (see, e.g., the work by Hochbaum et al. [17, 16, 15]). The main differences between this work (applied to UTVPI constraints) and ours are threefold. First, our focus is on satisfiability solving of arbitrary UTVPI formulas and not linear optimization over UTVPI polyhedra. Second, we do not require variables to be bounded. Finally, for our approximation result, the objective function can be an arbitrary linear function, without the restriction on the form of cost coefficients imposed in previous work.

Previous results on bounding solutions have been derived in the context of showing that integer linear programming is in NP [3, 34, 19, 27]. Even when specialized for UTVPI integer programs, these bounds are $\Omega(n^2 \cdot (b_{\max} + 1) \cdot 2^n)$. Our result is therefore an exponential reduction in the solution bound for UTVPI integer programs, and, to the best of our knowledge, has not been obtained before.

Our results rely on the modified version of Fourier-Motzkin elimination for checking integer feasibility of a UTVPI polyhedron; this algorithm is described by Subramani [33]. An incremental transitive closure based algorithm has been given by Harvey and Stuckey [14]. Miné [22] describes techniques to manipulate UTVPI (*octagon*) constraints for abstract interpretation, using the Harvey and Stuckey algorithm to deal with integer variables.

Lahiri and Musuvathi [20] also describe an algorithm to solve UTVPI constraints based on negative cycle detection; their algorithm is suitable for integration into a Nelson-Oppen style framework for cooperating decision procedures [23] as it generates equalities, and it is additionally proof and model generating.

Several *satisfiability modulo theories* (SMT) solvers have been proposed for solving formulas in integer linear arithmetic. The solvers participating in the recent 2006 SMT-COMP competition include Ario [31], CVC3 [7], ExtSAT [21], HTP [28], MathSAT [5], and Yices [11]. Other decision procedures include CVC-Lite [2] and Zapato [1]. These procedures are a combination of a SAT solver and a solver for a system of linear constraints; they differ in the many optimizations to the solvers themselves and the communication between the SAT engine and the integer linear solver such as layered solvers, theory propagation, pre-simplification of constraints, and fast backtracking. The linear solver is typically either Simplex, a transitive closure based algorithm, or the modified Fourier-Motzkin elimination algorithm referenced above (when specialized to UTVPI constraints). The BarcelogicTools SMT solver [25] pioneered the DPLL(T) approach [13, 26], which forms the basis for the Yices solver as well [10]; however, the BarcelogicTools solver currently handles only the difference logic fragment of integer linear arithmetic. We have performed experimental results to compare our approach against Yices, MathSAT, and Ario; a more detailed discussion of the techniques underlying those tools is deferred to Section 4.

1.2 Outline

The rest of the paper is organized as follows. We begin in Section 2 with useful background definitions and results. Section 3 contains the main theoretical contributions of this paper. We present experimental results in Section 4 and conclude in Section 5.

2. Background

We state here, in brief, some definitions and theorems used in the remainder of the paper. Further details can be found in standard textbooks on polyhedral theory and integer linear programming (e.g., [24, 29]).

Following standard linear programming notation, we denote the number of variables by n and number of constraints by m. We assume that a linear constraint is specified in the form $\mathbf{a} \cdot \mathbf{x} \geq b$, where $\mathbf{a}$ is an n-dimensional integer vector $[a_1,\ a_2,\ \ldots, a_n]$, $\mathbf{x}$ is an n-dimensional vector of integer-valued variables $[x_1, x_2, \ldots, x_n]$, and b is an integer. A system of constraints is specified as $\mathbf{A} \cdot \mathbf{x} \geq \mathbf{b}$, where $\mathbf{A}$ is a $m \times n$ matrix with integral entries, $\mathbf{b}$ is a $m \times 1$ integer vector $[b_1, b_2, \ldots, b_m]^{\mathrm{T}}$, and $\mathbf{x}$ is a $n \times 1$ vector of integer-valued variables. We use $b_{\max}$ to denote the L_∞ norm of $\mathbf{b}$; i.e., $b_{\max} = \max_i |b_i|$.

The system $\mathbf{A} \cdot \mathbf{x} \geq \mathbf{b}$ can be viewed as a polyhedron in $\mathbb{R}^n$ with the constraints defining half spaces. If $\delta = \max_{\mathbf{x} \in \mathbf{P}} \mathbf{a} \cdot \mathbf{x}$ is finite, $\mathbf{a} \cdot \mathbf{x} = \delta$ is called a *supporting hyperplane*. A *face* of a polyhedron $\mathbf{P}$ is either $\mathbf{P}$ itself or the intersection of $\mathbf{P}$ with a supporting hyperplane of $\mathbf{P}$.

An *orthant* of $\mathbb{R}^n$ is one of the 2^n sub-spaces of $\mathbb{R}^n$ defined by constraining each Cartesian coordinate axis to be positive or negative.

A *feasible* system of constraints is one that has a solution in $\mathbb{R}^n$. We say that a system is *lattice point feasible* if it has a solution in $\mathbb{Z}^n$. A *half-integral* solution to the system is a vector in $\mathbb{R}^n$ whose every component is an integer multiple of $\frac{1}{2}$. The terms *feasible* and *satisfiable* are used interchangeably, as also are *lattice point* and *integer point*.

2.1 UTVPI Formulas

***Definition* 2.1.** *A constraint* $\mathbf{a} \cdot \mathbf{x} \geq b$ *is said to be an* absolute *constraint if exactly one of the* a_i*s is non-zero, a* difference *constraint if exactly two of the* a_i*s are non-zero with one being* $+1$ *and the other* -1*, and a* sum *constraint if exactly two of the* a_i*s are non-zero with both* $+1$ *or both* -1*.*

$\mathbf{a} \cdot \mathbf{x} \geq b$ *is said to be a* UTVPI constraint *if it is either an absolute, a difference, or a sum constraint.*

A UTVPI formula is generated by the following grammar:

$$\phi \ ::= \ \mathbf{true} \mid \mathbf{false} \mid x_i + x_j \geq b \mid -x_i - x_j \geq b \mid x_i - x_j \geq b \mid x_i \geq b \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2$$

Notice that a logical negation on a UTVPI constraint can be eliminated by inverting the sense of the inequality, interchanging left and right hand sides, and adjusting the constant term. A UTVPI constraint remains UTVPI under such rewriting. The only change is to the sign of variable coefficients, and to the constant term, which can increase in absolute value by at most 1.

***Example* 2.1.** *Consider the following UTVPI formula*

$$(\neg\, x_1 + x_2 \geq -1) \;\wedge\; \big(x_2 - x_3 \geq 0 \;\vee\; x_4 \geq 1\big)$$

The constraint $x_1 + x_2 \geq -1$ is a sum constraint, $x_2 - x_3 \geq 0$ is a difference constraint, and $x_4 \geq 1$ is an absolute constraint. The negation can be eliminated to obtain an equivalent UTVPI formula

$$-x_1 - x_2 \geq 2 \;\wedge\; \big(x_2 - x_3 \geq 0 \;\vee\; x_4 \geq 1\big)$$

Note that the value of $b_{\max}$ has increased from 1 to 2 after eliminating the negation. □

***Definition* 2.2.** *Given a UTVPI formula ϕ, an* enumeration bound *is an integer d such that ϕ has an integer solution if and only if it contains an integer point in the n-dimensional hypercube $\prod_{i=1}^{n}[-d, d]$. The interval $[-d, d]$ is termed as an* enumeration domain.

2.2 Polyhedral Theory

***Definition* 2.3.** *A* minimal face *of a polyhedron is a face that does not contain any other face of the polyhedron. A point lying on a minimal face is called a* minimal face solution (MFS).

When the minimal face is an extreme point (vertex), a MFS is called a *basic feasible solution.*

***Example* 2.2.** *Consider the unbounded UTVPI polyhedron in $\mathbb{R}^2$ defined only by the sum constraint $x_1 + x_2 \geq 1$. The only face of this polyhedron is the line $x_1 + x_2 = 1$. Thus, this is also a minimal face of the polyhedron, and any point lying on this line is a MFS.* □

We write $(\mathbf{A}',\ \mathbf{b}') \subseteq (\mathbf{A},\ \mathbf{b})$ to indicate that the polyhedral system $\mathbf{A}' \cdot \mathbf{x} \geq \mathbf{b}'$ is a subsystem of the polyhedral system $\mathbf{A} \cdot \mathbf{x} \geq \mathbf{b}$. Also, for a matrix $\mathbf{A}$, let $r(\mathbf{A})$ denote the rank of $\mathbf{A}$. We have the following characterization of a minimal face.

***Theorem* 2.1** ([29])**.** *Let $\mathbf{P} = \{\mathbf{x} : \mathbf{A} \cdot \mathbf{x} \geq \mathbf{b}\}$ denote a polyhedron. A non-empty subset $\mathbf{F} \subseteq \mathbf{P}$ is a minimal face of $\mathbf{P}$, if and only if $\mathbf{F} = \{\mathbf{x} : \mathbf{A}' \cdot \mathbf{x} = \mathbf{b}'\}$, for some system $\mathbf{A}' \cdot \mathbf{x} \geq \mathbf{b}'$, where $(\mathbf{A}',\ \mathbf{b}') \subseteq (\mathbf{A},\ \mathbf{b})$, and $r(\mathbf{A}',\ \mathbf{b}') = r(\mathbf{A},\ \mathbf{b})$.*

Fourier-Motzkin (FM) elimination [9] is a well-known projection technique for polyhedra. Starting with a polyhedron $\mathbf{P} : \mathbf{A} \cdot \mathbf{x} \geq \mathbf{b}$, a variable x_j is projected using FM elimination in the following steps:

1. Partition the system of constraints into three sets P_j, N_j, Z_j as follows. For each constraint i, $1 \leq i \leq m$, we add it to:
$$\begin{array}{ll} P_j, & \text{if } a_{i,j} > 0; \\ N_j, & \text{if } a_{i,j} < 0; \\ Z_j, & \text{otherwise.} \end{array}$$

2. Initialize the set of new constraints, Φ, to Z_j.

3. For every pair of constraints (i_P, i_N), where $i_P \in P_j$ and $i_N \in N_j$, add the following constraint to Φ:

$$\sum_{k=1}^{n} \left(a_{i_P,j} \cdot a_{i_N,k} - a_{i_P,k} \cdot a_{i_N,j}\right) \cdot x_k \geq \tilde{b_i}$$

where $\tilde{b_i} = a_{i_P,j} b_{i_N} - a_{i_N,j} b_{i_P}$.

Clearly, the coefficient of x_j in every constraint in Φ is 0. Denote the polyhedron defined by the set Φ as $\tilde{\mathbf{P}} : \tilde{\mathbf{A}} \cdot \tilde{\mathbf{x}} \geq \tilde{\mathbf{b}}$. It is easy to see that $\mathbf{P}$ has a solution in $\mathbb{R}^n$ if and only if $\tilde{\mathbf{P}}$ has a solution in $\mathbb{R}^{n-1}$.

Note that if $\mathbf{P}$ is UTVPI, $\tilde{\mathbf{P}}$ need not be UTVPI. However, it is possible to modify the basic FM elimination procedure by adding a *coefficient normalization* step, so that the resulting polyhedron remains UTVPI, and moreover, is lattice point feasible iff $\mathbf{P}$ is. Notice that the only non-UTVPI constraints in $\tilde{\mathbf{P}}$ are of the form $2x_i \geq b$ or $-2x_i \geq b$, obtained by adding a sum constraint involving x_i and x_j with a difference constraint involving those variables. By dividing both sides of a newly created non-UTVPI constraint by 2, and rounding up the RHS if it is an odd multiple of $\frac{1}{2}$, we obtain a UTVPI constraint with the same integral solutions as the original. In this way, we replace each non-UTVPI constraint in $\tilde{\mathbf{P}}$ with a corresponding UTVPI constraint to obtain a UTVPI polyhedron $\mathbf{P'} : \mathbf{A'} \cdot \mathbf{x'} \geq \mathbf{b'}$.

We will refer to the modified FM elimination procedure as *Fourier-Motzkin elimination with coefficient normalization* (FM-CN). It is easy to see that FM-CN preserves integral solutions, i.e., $\mathbf{P}$ is lattice point feasible iff $\mathbf{P'}$ is. One can use FM-CN to check the feasibility of UTVPI polyhedra in time $O(n^3)$, by successively eliminating variables, checking at each step that we do not generate a trivially false constraint. At any step, we are guaranteed to have a system of no more than $O(n^2)$ constraints, since there are at most $2n + 4 \cdot \binom{n}{2} = 2n^2$ unique UTVPI linear forms on n variables.

3. Theoretical Results

Our theoretical results are organized as follows. We begin, in Section 3.1, by showing that if a UTVPI polyhedron has a minimal face solution (MFS), then there exists a MFS with each component half-integral and in $[-n \cdot b_{\max},\ n \cdot b_{\max}]$. The main theorem, presented in Section 3.3, enables us to go from bounding a MFS to bounding integer solutions. This theorem states that if a UTVPI polyhedron is integer feasible, then it is possible to find a integral solution within a unit box centered at any MFS; i.e., by "rounding" a MFS. In this section, we also describe how to extend results for UTVPI polyhedra to arbitrary UTVPI formulas. Section 3.2 presents auxiliary results on rounding that are used to prove the main theorem. Finally, in Section 3.4, we show that the main theorem can be used to obtain an additive approximation result for optimizing an arbitrary linear constraint over a UTVPI polyhedron.

3.1 Minimal Face Solutions of UTVPI Polyhedra

We begin with the following lemma.

***Lemma* 3.1.** *Let* $\mathbf{P} : \mathbf{A} \cdot \mathbf{x} \geq \mathbf{b}$ *represent a system of* m *difference constraints on* n *variables. Then,* $\mathbf{P}$ *has a feasible integer solution if and only if it has an integer solution in the hypercube* $\prod_{i=1}^{n}[0,\ (n-1) \cdot b_{\max}]$.

Proof. Consider the dual constraint graph as outlined by Cormen et al. [8]. A solution to the system is obtained by assigning to each variable, the shortest path from the source. The length of any shortest path is bounded by $(n-1) \cdot b_{\max}$. The result follows. □

***Example* 3.1.** *We illustrate Lemma 3.1 by the system comprising the two constraints:* $x_1 - x_2 \geq 3, x_2 - x_3 \geq 2$. *Here* $n = 3$ *and* $b_{\max} = 3$, *so* $(n-1)b_{\max} = 6$. *The solution* $x_1 = 5, x_2 = 2, x_3 = 0$ *is one where each* x_i *is in* $[0, 6]$. □

The following lemma considers bounding a MFS of a UTVPI polyhedron in the non-negative orthant.

***Lemma* 3.2.** *Let* $\mathbf{P} : \mathbf{A} \cdot \mathbf{x} \geq \mathbf{b},\ \mathbf{x} \geq \mathbf{0}$ *denote an arbitrary UTVPI polyhedron in the non-negative orthant with* m *constraints and* n *variables. Then, if a MFS exists, then there is a MFS with each component half-integral and at most* $n \cdot b_{\max}$.

Proof. Suppose polyhedron $\mathbf{P}$ has a minimal face solution. Hochbaum et al. [17] have shown that this MFS must be half-integral. We focus here on showing the $n \cdot b_{\max}$ bound.

By definition, the minimal face corresponding to this MFS satisfies a system $\mathbf{A'} \cdot \mathbf{x} = \mathbf{b'}$, where $(\mathbf{A'}\ \mathbf{b'}) \subseteq (\mathbf{A}\ \mathbf{b})$, and $r(\mathbf{A'}) = r(\mathbf{A}) = k$ for some $1 \leq k \leq n$. Accordingly, there are k independent variables and $n - k$ dependent variables in the system; without loss of generality, we assume that the first k variables are independent and set the dependent variables to 0. This results in a system $\mathbf{P_1} : \mathbf{A''} \cdot \mathbf{x''} = \mathbf{b''},\ \mathbf{x''} \geq \mathbf{0}$, where the components of $\mathbf{b''}$ are also components of $\mathbf{b}$, and $\mathbf{x''} = [x_1, x_2, \ldots, x_k]^T$.

The system $\mathbf{P_1}$ contains 3 types of constraints (equations), viz., absolute, difference and sum. We consider each of these types in turn:

1. An absolute constraint is of the form $x_i = b$. Since $\mathbf{x''} \geq \mathbf{0}$, the value of x_i must be in $[0, b_{\max}]$.

2. A sum constraint can be written in the form $x_i + x_j = b$, where $b \geq 0$. Since $\mathbf{x''} \geq \mathbf{0}$, it follows that $0 \leq x_i, x_j \leq b \leq b_{\max}$.

3. From the two cases above, we conclude that the value of any variable appearing in an absolute or sum constraint must lie in $[0, b_{\max}]$ (and moreover, there exists such a half-integral value).

   W.l.o.g, let $x_1, x_2, \ldots, x_l$, $l \leq k$, be variables appearing in the absolute and sum constraints, and let $x_1^*, x_2^*, \ldots, x_l^*$ be the corresponding half-integral values in $[0, b_{\max}]$ satisfying these constraints. Substituting these values into the difference constraints might create new absolute constraints, but no new difference or sum constraints. The constant term in new absolute constraints generated thus is half-integral and of absolute value at most $2b_{\max}$. The substitution process can be iterated at most $k - 1$ times leading to absolute constraints with half-integral constant terms at most $k \cdot b_{\max}$. Thus, a variable appearing in any of the absolute constraints generated in this iterative process takes half-integral values in $[0, k \cdot b_{\max}]$.

When the above iterative substitution process terminates, the only constraints possibly left are some of the original difference constraints, each with an integral constant term of absolute value at most $b_{\max}$. Since these constraints are satisfiable, we can apply Lemma 3.1 to conclude that there exists a solution to these constraints with each variable taking integral values in $[0, (k-1) \cdot b_{\max}]$ (since at most k variables appear in these constraints).

Since $k \leq n$, we conclude that there exists a solution to $\mathbf{P_1}$ with each component at most $n \cdot b_{\max}$. □

We now generalize the result to an arbitrary UTVPI polyhedron.

***Theorem* 3.1.** *Let $\mathbf{P} : \mathbf{A} \cdot \mathbf{x} \geq \mathbf{b}$ denote an arbitrary UTVPI polyhedron with m constraints and n variables. If a MFS exists, there exists a MFS with each component half-integral and in the interval $[-n \cdot b_{\max},\ n \cdot b_{\max}]$.*

We first illustrate the above theorem with an example.

***Example* 3.2.** *Recall Example 2.2, in which we had the unbounded UTVPI polyhedron in $\mathbb{R}^2$ defined only by the sum constraint $x_1 + x_2 \geq 1$, with the single (minimal) face $x_1 + x_2 = 1$. For this example, $n = 2$ and $b_{\max} = 1$.*

The point $(100, -99)$ lies on this face and hence is an MFS. However, we also have the half-integral MFS $(\frac{1}{2}, \frac{1}{2})$, each of whose components lies in the interval $[-2, 2]$. □

The proof of the theorem follows.

Proof. (Theorem 3.1) Suppose $\mathbf{x}^*$ is a MFS of $\mathbf{P}$. Let $j_1, j_2, \ldots, j_k$ be the set of all column indices, $1 \leq j_1, j_2, \ldots, j_k \leq n$, such that $x^*_{j_l} < 0$ for all l, $1 \leq l \leq k$. Construct a matrix $\mathbf{A'}$ by multiplying the j_lth column of $\mathbf{A}$ by -1 for all l, leaving other columns unchanged. We observe that:

1. The polyhedron $\mathbf{P'} : \mathbf{A'} \cdot \mathbf{x} \geq \mathbf{b},\ \mathbf{x} \geq \mathbf{0}$ is also UTVPI.
2. If we construct $\mathbf{x'}^*$ from $\mathbf{x}^*$ by negating $x^*_{j_l}$ for all l, $1 \leq l \leq k$, $\mathbf{x'}^*$ satisfies $\mathbf{P'}$. Moreover, we argue that it is a MFS of $\mathbf{P'}$ as follows:
   Let $(\tilde{\mathbf{A}}, \tilde{\mathbf{b}}) \subseteq (\mathbf{A}, \mathbf{b})$ be the constraints satisfied with equality at $\mathbf{x}^*$, and $(\tilde{\mathbf{A}'}, \tilde{\mathbf{b}'}) \subseteq (\mathbf{A'}, \mathbf{b'})$ be the constraints satisfied with equality at $\mathbf{x'}^*$. Then, $r(\tilde{\mathbf{A}}) = r(\tilde{\mathbf{A}'})$, since $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{A}'}$ correspond to the same rows (of $\mathbf{A}$ and $\mathbf{A'}$ respectively). Also, note that $r(\mathbf{A}) = r(\mathbf{A'})$. Finally, since $(\tilde{\mathbf{A}}, \tilde{\mathbf{b}})$ define a minimal face of $\mathbf{P}$, $r(\tilde{\mathbf{A}}) = r(\mathbf{A})$ [29].
   Thus, $r(\tilde{\mathbf{A}'}) = r(\mathbf{A'})$, and so $\mathbf{x'}^*$ is a MFS of $\mathbf{P'}$.

Using an identical argument, we conclude that, from a MFS of $\mathbf{P'}$, we can construct a MFS of $\mathbf{P}$ by negating values to $x_{j_1}, x_{j_2}, \ldots, x_{j_k}$.

Since $\mathbf{P'}$ has a MFS, by Lemma (3.2) it must have a MFS with each component half-integral and in $[0, n \cdot b_{\max}]$. It follows that $\mathbf{P}$ has a MFS with each component half-integral and in $[-n \cdot b_{\max},\ n \cdot b_{\max}]$. □

One can construct examples where an end point of the enumeration domain stated in Theorem 3.1 is attained. For instance, if the system of constraints comprises the equalities $x_1 = b_{\max}$, $x_i - x_{i-1} = b_{\max}$ for $2 \leq i \leq n-1$, and $x_n + x_{n-1} = -b_{\max}$, the only MFS has $x_j = j \cdot b_{\max}$ for $1 \leq j \leq n-1$ and $x_n = -n \cdot b_{\max}$.

3.2 Rounding and Semi-Rounding

***Definition* 3.1.** *A rational number x is said to be* odd half-integral *if it is an odd multiple of $\frac{1}{2}$.*

***Definition* 3.2.** *A vector $\mathbf{z}$ is said to be a* rounding *of a vector $\mathbf{x}$ if $\mathbf{z}$ is integral and $||\mathbf{z} - \mathbf{x}||_\infty \leq \frac{1}{2}$.*

***Definition* 3.3.** *A vector $\mathbf{z}$ is said to be a* semi-rounding *of a vector $\mathbf{x}$ if all of the following conditions hold: (1) $||\mathbf{z} - \mathbf{x}||_\infty \leq \frac{1}{2}$; (2) all components of $\mathbf{z}$ are half-integral; and (3) if a component of $\mathbf{x}$ is integral, so is the corresponding component of $\mathbf{z}$.*

For example, the two-dimensional vectors $(1, \frac{1}{2})$ and $(1, 1)$ are both semi-roundings of the vector $(1, \frac{1}{3})$, but only $(1, 1)$ is a rounding of $(1, \frac{1}{3})$,

***Lemma* 3.3.** *Let $\mathbf{a} \cdot \mathbf{x} \geq b$ be a UTVPI constraint. Let $\mathbf{x}^*$ be a half-integral vector such that $\mathbf{a} \cdot \mathbf{x}^* > b$, and let $\mathbf{w}^*$ be an arbitrary semi-rounding of $\mathbf{x}^*$. Then, $\mathbf{a} \cdot \mathbf{w}^* \geq b$.*

Proof. The proof proceeds by case splitting on the number of variables in the constraint.

1. Suppose the constraint involves only one variable. Then, it is either of the form $x_i \geq b$ or $-x_i \geq b$. Correspondingly, we either have $x_i^* > b$ or $-x_i^* > b$. Since x_i^* is half-integral, in both cases the LHS exceeds b by at least $\frac{1}{2}$. Thus, any semi-rounding w_i^* of x_i^* satisfies the constraint.

2. Suppose the constraint has two variables, x_i and x_j. Then, since x_i^* and x_j^* are both half-integral, one of the following two cases must hold:

   (a) The LHS is integral, and exceeds b by at least 1. But any semi-rounding of x_i^* and x_j^* can decrease the LHS by at most 1, and hence satisfies the constraint.

   (b) The LHS is odd half-integral, i.e., one of x_i^* and x_j^* is integral and the other odd half-integral. Thus, the LHS exceeds b by at least $\frac{1}{2}$. In this case, any semi-rounding of x_i^* and x_j^* can decrease the LHS by at most $\frac{1}{2}$, and will satisfy the constraint.

□

Since every rounding $\mathbf{z}$ of $\mathbf{x}^*$ is also a semi-rounding of $\mathbf{x}^*$, we obtain the following corollary:

***Corollary* 3.1.** *Let $\mathbf{a} \cdot \mathbf{x} \geq b$ be a UTVPI constraint. Let $\mathbf{x}^*$ be a half-integral vector such that $\mathbf{a} \cdot \mathbf{x}^* > b$, and let $\mathbf{z}$ be an arbitrary rounding of $\mathbf{x}^*$. Then, $\mathbf{a} \cdot \mathbf{z} \geq b$.*

We now state a useful property of Fourier-Motzkin elimination with coefficient normalization.

***Proposition* 3.1.**
Let $\mathbf{P} : \mathbf{A} \cdot \mathbf{x} \geq \mathbf{b}$ denote a UTVPI polyhedron in $\mathbb{R}^{n+1}$ and $\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_{n+1}^*)$ denote a half-integral feasible solution to $\mathbf{P}$. Further, suppose that $\mathbf{P}$ is lattice point feasible.

Let $\mathbf{P}' : \mathbf{A}' \cdot \mathbf{x}' \geq \mathbf{b}'$ be obtained from $\mathbf{P}$ by projecting out variable x_{n+1} using Fourier-Motzkin elimination with coefficient normalization and denote $(x_1^*, x_2^*, \ldots, x_n^*)$ by $\mathbf{x}'^*$. Then, there exists a semi-rounding $\mathbf{w}'^*$ of $\mathbf{x}'^*$ such that $\mathbf{w}'^*$ is a solution to $\mathbf{P}'$.

The statement of this lemma is rather technical; to clarify it, we first give the following example, and then continue with the proof.

***Example* 3.3.** *Let* $\mathbf{P}$ *be the UTVPI polyhedron defined by the following four constraints:*

$$\begin{array}{ll} x_1 + x_2 \geq 2 & -x_1 - x_2 \geq -3 \\ x_1 - x_2 \geq -1 & -x_1 + x_2 \geq 0 \end{array}$$

Pictorially, $\mathbf{P}$ *is the square in the positive quadrant of* $\mathbb{R}^2$ *with vertices* $(\frac{1}{2},\frac{3}{2})$,(1,1),(1,2),$(\frac{3}{2},\frac{3}{2})$. *Two of the vertices of* $\mathbf{P}$ *are lattice points.*

If we project out x_2 *using FM-CN, we obtain the system defined by the two inequalities:*

$$\begin{aligned} x_1 &\geq \left\lceil \frac{1}{2} \right\rceil = 1 \\ -x_1 &\geq \left\lceil \frac{-3}{2} \right\rceil = -1 \end{aligned}$$

This corresponds to the single-point polyhedron $\mathbf{P'}$ *defined by* $x_1 = 1$.

Let $\mathbf{x}^* = (\frac{1}{2}, \frac{3}{2})$. *Then,* $\mathbf{x'}^* = \frac{1}{2}$. *Clearly,* $\mathbf{x'}^*$ *is not in* $\mathbf{P'}$. *However, we can obtain a semi-rounding* $\mathbf{w'}^* = 1$ *of* $\mathbf{x'}^*$ *that does lie in* $\mathbf{P'}$.

□

Proof. (Proposition 3.1)
First, note that since $\mathbf{P}$ is lattice point feasible, so is $\mathbf{P'}$.

If $\mathbf{x'}^*$ is already a solution to $\mathbf{P'}$ then the theorem holds trivially.

So suppose that $\mathbf{x'}^*$ does not satisfy $\mathbf{P'}$. The only reason this occurs is because $\mathbf{x'}^*$ is cut off by coefficient normalization, i.e., due to the presence of one or both of the following situations:

1. There exists at least one variable x_i such that $\mathbf{P}$ has constraints of the form:

$$x_i - x_{n+1} \geq b_i^+ \quad (1)$$
$$x_i + x_{n+1} \geq b_i^{+\prime} \quad (2)$$

which result in the following constraint in $\mathbf{P'}$:

$$x_i \geq \left\lceil \frac{b_i^+ + b_i^{+\prime}}{2} \right\rceil \quad (3)$$

where, $b_i^+ + b_i^{+\prime}$ is odd.

Since $\mathbf{x'}^*$ does not satisfy $\mathbf{P'}$, the following equality also holds:

$$x_i^* = \frac{b_i^+ + b_i^{+\prime}}{2} \quad (4)$$

Let I be the index set of all such variables x_i.

2. There exists at least one variable x_j such that $\mathbf{P}$ has constraints of the form:

$$-x_j + x_{n+1} \geq b_j^- \tag{5}$$
$$-x_j - x_{n+1} \geq b_j^{-\prime} \tag{6}$$

which result in the following constraint in $\mathbf{P'}$:

$$x_j \leq \left\lfloor \frac{-b_j^- - b_j^{-\prime}}{2} \right\rfloor \tag{7}$$

where, $b_j^- + b_j^{-\prime}$ is odd.

Since $\mathbf{x'}^*$ does not satisfy $\mathbf{P'}$, the following equality also holds:

$$x_j^* = \frac{-b_j^- - b_j^{-\prime}}{2} \tag{8}$$

Let J be the index set of all such variables x_j.

Note that for some $i \in I$, and $j \in J$, if $i = j$, then we must have $\frac{b_i^+ + b_i^{+\prime}}{2} = \frac{-b_j^- - b_j^{-\prime}}{2}$. But that would mean that $\mathbf{P'}$ is infeasible, since constraints (3) and (7) would contradict each other. Hence, we can assume hereafter that the two index sets I and J are disjoint.

We now give a rounding algorithm that generates a semi-rounding $\mathbf{w'}^*$ of $\mathbf{x'}^*$ that satisfies $\mathbf{P'}$. The rounding algorithm is as follows:

1. Initialize the set of variables to be rounded up, $\mathcal{U}$, to be $\{x_i | i \in I\}$. Similarly, initialize the set of variables to be rounded down, $\mathcal{D}$ as $\{x_j | j \in J\}$.

2. $\mathcal{U}_0 := \mathcal{U}$, $\mathcal{D}_0 := \mathcal{D}$, $t := 0$.

3. Compute $\mathcal{U}_{t+1}$ and $\mathcal{D}_{t+1}$ as follows. For every $x_i \in \mathcal{U}_t$ and $x_j \in \mathcal{D}_t$,

   (a) Include in $\mathcal{U}_{t+1}$ any variable x_k such that the following constraints in $\mathbf{P'}$, which are valid for $\mathbf{P}$, hold with equality at $\mathbf{x'}^*$:

$$x_k - x_i \geq b_{ki} \tag{9}$$
$$x_j + x_k \geq b_{jk} \tag{10}$$

   (b) Include in $\mathcal{D}_{t+1}$ any variable x_k such that the following constraints in $\mathbf{P'}$, which are valid for $\mathbf{P}$, hold with equality at $\mathbf{x'}^*$:

$$-x_k - x_i \geq b'_{ki} \tag{11}$$
$$x_j - x_k \geq b'_{jk} \tag{12}$$

4. If $\mathcal{U}_{t+1} \subseteq \mathcal{U}$ and $\mathcal{D}_{t+1} \subseteq \mathcal{D}$, stop.

   Otherwise, perform the assignments $\mathcal{U} := \mathcal{U} \cup \mathcal{U}_{t+1}$, $\mathcal{D} := \mathcal{D} \cup \mathcal{D}_{t+1}$, $t := t + 1$, and go to step (3).

It is easy to prove by induction on t, that for any $x_k \in \mathcal{U}$, $k \notin I$, there either exists $i \in I$ and an integer b_{ki} such that

$$x_k^* - x_i^* = b_{ki} \tag{13}$$

or a $j \in J$ and an integer b_{jk} such that

$$x_j^* + x_k^* = b_{jk} \tag{14}$$

Similarly, for each $x_k \in \mathcal{D}$, $k \notin J$, there either exists $i \in I$ and an integer b'_{ki} such that

$$-x_k^* - x_i^* = b'_{ki} \tag{15}$$

or a $j \in J$ and an integer b'_{jk} such that

$$x_j^* - x_k^* = b'_{jk} \tag{16}$$

Suppose the two sets $\mathcal{U}$ and $\mathcal{D}$ are disjoint. Then, to obtain a semi-rounding $\mathbf{w'}^*$ of $\mathbf{x'}^*$, we round up every variable in $\mathcal{U}$ and round down every variable in $\mathcal{D}$.

To complete the proof, the following two sub-goals remain to be established:

1. $\mathcal{U} \cap \mathcal{D} = \emptyset$.
2. $\mathbf{w'}^*$ satisfies $\mathbf{P'}$.

Assuming the first sub-goal, consider the second sub-goal first. We observe that:

- By Lemma 3.3, any constraints in $\mathbf{P'}$ that are not satisfied with equality at $\mathbf{x'}^*$ will continue to be satisfied by $\mathbf{w'}^*$.
- From Equations (13)–(16), we note that for all $x_k \in \mathcal{U} \cup \mathcal{D}$, x_k^* is odd half-integral, since it is an integral offset from x_i^* or x_j^* for some $i \in I$ or $j \in J$.

  Thus, for all $x_k \in \mathcal{U} \cup \mathcal{D}$, there cannot be any absolute constraint involving x_k in $\mathbf{P'}$ that holds with equality at $\mathbf{x'}^*$. Thus, by Lemma 3.3, the semi-rounding produced by the above algorithm satisfies these absolute constraints.
- Steps 3(a) and 3(b) of the rounding algorithm ensure that all two-variable constraints of $\mathbf{P'}$ satisfied with equality at $\mathbf{x'}^*$ continue to be satisfied by the generated semi-rounding. For example, if $x_k - x_i \geq b_{ki}$ is satisfied with equality at $\mathbf{x'}^*$, and x_i^* is rounded up, so is x_k^*, so the constraint continues to be satisfied.

Thus, if the two sets $\mathcal{U}$ and $\mathcal{D}$ are disjoint, we can conclude that $\mathbf{w'}^*$ satisfies $\mathbf{P'}$. We will now show that the former is indeed the case.

The proof is by contradiction. Suppose $\mathcal{U} \cap \mathcal{D} \neq \emptyset$. Let x_k be a variable present in both sets. As we noted before, for any $i \in I$ and $j \in J$, $i \neq j$, so we can assume that k is neither in I nor in J. We have the following cases, each of which leads to a contradiction:

1. Equations (13) and (16) hold. Then, for some integer b_{ji}, we have

$$x_j^* - x_i^* = b_{ji} \tag{17}$$

The above equation corresponds to the following inequality derived by adding Inequalities (9) and (12), which is valid for both $\mathbf{P}$ and $\mathbf{P}'$:

$$x_j - x_i \geq b_{ji} \tag{18}$$

Further, from Equation (17) and Inequalities (1), (2), (5), and (6), we can conclude that

$$-b_{ji} = x_i^* - x_j^* \geq b_i^+ + b_j^- \tag{19}$$

$$-b_{ji} = x_i^* - x_j^* \geq b_i^{+\prime} + b_j^{-\prime} \tag{20}$$

Also from Equations (4) and (8), we know that

$$-b_{ji} = x_i^* - x_j^* = \frac{b_i^+ + b_j^- + b_i^{+\prime} + b_j^{-\prime}}{2} \tag{21}$$

From (19), (20), and (21) above, we infer that $b_i^+ + b_j^- = b_i^{+\prime} + b_j^{-\prime} = -b_{ji}$.

Thus, the inequalities in (19) and (20) hold with equality. Also, from Inequalities (1) and (5), $x_i - x_j \geq b_i^+ + b_j^-$ is valid for $\mathbf{P}$. Thus, we can conclude that Inequality (18) holds with equality for $\mathbf{P}$. This further implies that Inequalities (1), (2), (5), and (6) hold with equality for $\mathbf{P}$.

Since there is a unique solution to Constraints (1), (2), (5), (6) and (18) that satisfies them with equality, in every feasible solution of $\mathbf{P}$, $x_i = x_i^*$, $x_j = x_j^*$, and $x_{n+1} = x_{n+1}^*$. Since at least one of x_i^* and x_j^* is odd half-integral, this contradicts the premise that $\mathbf{P}$ has a lattice point solution.

2. Equations (14) and (15) hold. This case is identical to Case (1) above.

3. Equations (14) and (16) hold. Then, we have

$$x_j^* = \frac{b_{jk} + b'_{jk}}{2} \tag{22}$$

This implies that $\frac{b_{jk}+b'_{jk}}{2} = \frac{-b_j^- - b_j^{-\prime}}{2}$.

Further, Equation (22) corresponds to the following valid cut for $\mathbf{P}'$ (i.e., it preserves lattice point solutions), obtained by adding (10) and (12):

$$x_j \geq \left\lceil \frac{b_{jk} + b'_{jk}}{2} \right\rceil \tag{23}$$

However, Constraints (7) and (23) contradict each other, implying that $\mathbf{P}'$ is not lattice point feasible, which contradicts the theorem's premise.

4. Equations (13) and (15) hold. This case is identical to Case (3) above.

Thus, $\mathcal{U} \cap \mathcal{D} = \emptyset$ and we obtain a semi-rounding $\mathbf{w}'^*$ of $\mathbf{x}'^*$ as required. This completes the proof.

□

3.3 Main Theorems

We now arrive at the key result of this paper.

***Theorem* 3.2.** *Let* $\mathbf{P} : \mathbf{A} \cdot \mathbf{x} \geq \mathbf{b}$ *denote a UTVPI polyhedron and* $\mathbf{x}^*$ *denote a half-integral MFS. If* $\mathbf{P}$ *is lattice point feasible, then it contains a lattice point* $\mathbf{z}$ *such that* $||\mathbf{z} - \mathbf{x}^*||_\infty \leq \frac{1}{2}$*, i.e.,* $\mathbf{z}$ *is a rounding of* $\mathbf{x}^*$*.*

Proof. We prove the theorem by induction on the length of $\mathbf{x}$.

Base Case: Let $\mathbf{x} = x \in \mathbb{R}$. If x^* is a MFS, there exists a constraint $x \geq b$ that holds with equality for x^*. Thus, the theorem holds trivially for $\mathbf{z} = x^*$.

Induction Step: Let us assume that the theorem holds for lengths up to n; i.e., for all polyhedra $\mathbf{P}$ defined in terms of vectors $\mathbf{x}$ of length up to n.

Consider the case when $\mathbf{x} \in \mathbb{R}^{n+1}$. Let $\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_{n+1}^*)$ be a half-integral MFS of $\mathbf{P}$. If $\mathbf{x}^*$ is integral, we set $\mathbf{z}$ to $\mathbf{x}^*$ and we are done. So, let us assume that $\mathbf{x}^*$ has some odd half-integral entries. Note that if two variables x_i and x_j appear together in a constraint of $\mathbf{P}$ that holds with equality, either both x_i^* and x_j^* are integral or both are odd half-integral.

Project variable x_{n+1} out of $\mathbf{P}$ using Fourier-Motzkin elimination with coefficient normalization (FM-CN). Let $\mathbf{P'} : \mathbf{A'} \cdot \mathbf{x'} \geq \mathbf{b'}$ be the resulting system, where $\mathbf{x'} \in \mathbb{R}^n$.

Suppose there exists a lattice point solution $\mathbf{y} = (y_1, y_2, \ldots, y_{n+1})$ of $\mathbf{P}$. Thus, $\mathbf{y'} = (y_1, y_2, \ldots, y_n)$ is a lattice point solution of $\mathbf{P'}$.

Consider $\mathbf{x'}^* = (x_1^*, x_2^*, \ldots, x_n^*)$. We will show that there exists a rounding $\mathbf{z'} = (z_1, z_2, \ldots, z_n)$ of $\mathbf{x'}^*$ which satisfies $\mathbf{P'}$. We consider the following three cases:

Case 1: $\mathbf{x'}^*$ is in the interior of $\mathbf{P'}$, i.e., none of the constraints in $\mathbf{A'} \cdot \mathbf{x'} \geq \mathbf{b'}$ hold with equality. By Corollary 3.1, any rounding of $\mathbf{x'}^*$ yields a lattice point solution $\mathbf{z'}$ of $\mathbf{P'}$.

Case 2: Suppose that $\mathbf{x'}^*$ is a solution of $\mathbf{P'}$ that satisfies some constraints with equality. Suppose that for some $(\mathbf{A''},\ \mathbf{b''}) \subseteq (\mathbf{A'},\ \mathbf{b'})$, $\mathbf{A''} \cdot \mathbf{x'}^* = \mathbf{b''}$, and the remaining constraints are strict, i.e., not satisfied with equality. Since $\mathbf{x'}^*$ is a MFS of $\mathbf{A''} \cdot \mathbf{x'} \geq \mathbf{b''}$, by the induction hypothesis, we can conclude that there exists a lattice point rounding $\mathbf{z'}$ of $\mathbf{x'}^*$, such that $\mathbf{z'}$ is a solution of $\mathbf{A''} \cdot \mathbf{x'} \geq \mathbf{b''}$. Since, by Corollary 3.1, any rounding of $\mathbf{x'}^*$ satisfies the strict constraints, $\mathbf{z'}$ is also a lattice point solution of $\mathbf{P'}$.

Case 3: It is possible that after coefficient normalization, $\mathbf{x'}^*$ does not satisfy $\mathbf{P'}$. By Proposition 3.1, there exists a semi-rounding $\mathbf{w'}^*$ of $\mathbf{x'}^*$ that satisfies $\mathbf{P'}$. Thus, either Case (1) or Case (2) applies with $\mathbf{x'}^*$ replaced by $\mathbf{w'}^*$, and we can obtain a rounding $\mathbf{z'}$ of $\mathbf{w'}^*$ that is a lattice point solution of $\mathbf{P'}$. Finally, note that a rounding of $\mathbf{w'}^*$ is also a rounding of $\mathbf{x'}^*$, since integral components of $\mathbf{x'}^*$ are preserved in $\mathbf{w'}^*$. This completes Case (3).

Thus, we can obtain a lattice point solution $\mathbf{z'}$ of $\mathbf{P'}$ that is a rounding of $\mathbf{x'}^*$.

Since $\mathbf{P}$ is UTVPI, and $\mathbf{P'}$ is obtained from $\mathbf{P}$ using FM-CN, a lattice point solution of $\mathbf{P'}$ can be extended to one of $\mathbf{P}$. Thus, there exists an integral z_{n+1} such that $\mathbf{z} = (z_1, z_2, \ldots, z_n, z_{n+1})$ is a solution of $\mathbf{P}$.

To complete the proof, we show that there exists such an integral z_{n+1} that is moreover a rounding of x_{n+1}^*. Since $\mathbf{x}^*$ is a MFS of $\mathbf{P}$, there exists a subset of constraints $(\tilde{\mathbf{A}}, \tilde{\mathbf{b}})$ of $(\mathbf{A}, \mathbf{b})$ that hold with equality at $\mathbf{x}^*$. The value of x_{n+1} is constrained only by the values

of other variables x_j such that there exists an equation in $\tilde{\mathbf{A}}\mathbf{x} = \tilde{\mathbf{b}}$ in which x_{n+1} and x_j appear together. Let J be the index set of all such variables x_j. We now show that there exists a rounding z_{n+1} of x^*_{n+1} that satisfies $\mathbf{P_1} : \tilde{\mathbf{A}}\mathbf{x} \geq \tilde{\mathbf{b}}$. There are two cases:

1. If x^*_{n+1} is integral, so is x^*_j for all $j \in J$. Thus, $z_{n+1} = x^*_{n+1}$ satisfies $\mathbf{P_1}$, and we are done.

2. If x^*_{n+1} is odd half-integral, so is x^*_j for all $j \in J$. In this case, we claim that there exists a consistent way to round x^*_{n+1}, either up or down, so that the result satisfies $\mathbf{P_1}$. Suppose not, i.e., there exist constraints that force x^*_{n+1} to be rounded up as well as down. There are four instances in which this might occur:

   (a) There exist constraints $x_{n+1} - x_i \geq b$ and $x_j - x_{n+1} \geq b'$ in $\mathbf{P}$ that hold with equality at $\mathbf{x}^*$; furthermore, $z_j = \lfloor x^*_j \rfloor$ and $z_i = \lceil x^*_i \rceil$. Thus, we have $x^*_j - x^*_i = b + b'$, but $z_j - z_i < b + b'$. Since $x_j - x_i \geq b + b'$ is a valid inequality for $\mathbf{P}$, this means that $\mathbf{z}$ does not lie in $\mathbf{P}$, a contradiction.

   (b) There exist constraints $-x_{n+1} - x_i \geq b$ and $x_j + x_{n+1} \geq b'$ in $\mathbf{P}$ that hold with equality at $\mathbf{x}^*$; furthermore, $z_j = \lfloor x^*_j \rfloor$ and $z_i = \lceil x^*_i \rceil$. This case is identical to Case (2a) above.

   (c) There exist constraints $x_j - x_{n+1} \geq b$ and $x_j + x_{n+1} \geq b'$ in $\mathbf{P}$ that hold with equality at $\mathbf{x}^*$, with $z_j = \lfloor x^*_j \rfloor$. Thus, $2x^*_j = b + b'$. Since, x^*_j is odd half-integral, $b + b'$ must be an odd integer. Moreover, $2z_j < b + b'$. However, since $2x_j \geq b + b'$ is a valid inequality for $\mathbf{P}$, this means that $\mathbf{z}$ does not lie in $\mathbf{P}$, a contradiction.

   (d) There exist constraints $x_{n+1} - x_i \geq b$ and $-x_{n+1} - x_i \geq b'$ in $\mathbf{P}$ that hold with equality at $\mathbf{x}^*$, with $z_i = \lceil x^*_i \rceil$. This case is identical to Case (2c) above.

   Thus, there exists a consistent way to round x^*_{n+1} either up or down and satisfy every constraint in $\mathbf{P_1}$. Let z_{n+1} be this rounding.

Applying Corollary 3.1, any rounding of x^* satisfies the constraints in $(\mathbf{A}, \mathbf{b}) \setminus (\tilde{\mathbf{A}}, \tilde{\mathbf{b}})$.

Thus, we can obtain a rounding $\mathbf{z}$ of $\mathbf{x}^*$ that is a lattice point solution of $\mathbf{P}$.

□

From Theorem (3.1) and Theorem (3.2), we can conclude the following theorem.

***Theorem* 3.3.** *Let $\mathbf{P} : \mathbf{A} \cdot \mathbf{x} \geq \mathbf{b}$ denote a UTVPI polyhedron with m constraints and n variables. Then, $\mathbf{P}$ has enumeration bound $n \cdot b_{\max}$.*

The above result is easily generalized for arbitrary UTVPI formulas.

***Theorem* 3.4.** *Let ϕ denote a UTVPI formula with m constraints, n variables, and let $b_{\max}$ be the maximum over the absolute values of constant terms appearing in ϕ. Then, ϕ has enumeration bound $n \cdot (b_{\max} + 1)$.*

Proof. If ϕ has a satisfying integer solution, that solution must satisfy one of the terms in the disjunctive normal form (DNF) of ϕ. Each term in the DNF representation of ϕ is a UTVPI polyhedron in which the constant term in any constraint has absolute value at most $b_{\max} + 1$ (we use $b_{\max} + 1$ in place of $b_{\max}$ to account for eliminating negations on constraints). It follows that there is a solution to ϕ in $[-n \cdot (b_{\max} + 1),\ n \cdot (b_{\max} + 1)]$. □

3.4 Approximation Results for Optimization

Consider the problem of optimizing an arbitrary linear function over a UTVPI polyhedron $\mathbf{P}$. This problem is NP-hard (minimum vertex cover is a special case). As a corollary of Theorem (3.2), we obtain the following theorem showing that one can approximate the optimal value to within an additive factor.

***Theorem* 3.5.** *Let* $\mathbf{P} = \{\mathbf{x} : \mathbf{A} \cdot \mathbf{x} \geq \mathbf{b}\}$ *denote a UTVPI polyhedron that contains a lattice point. Let the integer linear program be* $\max \mathbf{c} \cdot \mathbf{x} : \mathbf{x} \in \mathbf{P}$.

If the optimum value is finite, solving the LP-relaxation and rounding the resulting solution yields a feasible lattice point that approximates the optimum to within an additive factor of $\pm\frac{\sum_{j=1}^{n} |c_j|}{2}$. *If the LP-relaxation is unbounded, so is the integer program.*

Proof. If the optimum value v^* of the LP-relaxation is finite, it is attained at a MFS $\mathbf{x}^*$. The MFS $\mathbf{x}^*$ must be half-integral [17]. Since $\mathbf{P}$ is lattice point feasible, by Theorem 3.2, there exists a lattice point $\mathbf{z}$ in $\mathbf{P}$ such that such that $||\mathbf{z} - \mathbf{x}^*||_\infty \leq \frac{1}{2}$. It follows that $\mathbf{c} \cdot \mathbf{z}$ is within $\pm\frac{\sum_{j=1}^{n} |c_j|}{2}$ of v^*, and hence of the integer optimum.

If the LP-relaxation is unbounded, so must be the integer program, since $\mathbf{P}$ is lattice point feasible [24]. □

Moreover, an approximate solution can be obtained in polynomial time in the following three steps:

1. Check whether $\mathbf{P}$ is lattice point feasible using Fourier-Motzkin elimination with coefficient normalization. If $\mathbf{P}$ is lattice point infeasible, stop.
2. If $\mathbf{P}$ is lattice point feasible, solve its LP-relaxation. If it is unbounded, we conclude that the original IP is also unbounded. Otherwise, the optimum is attained at a MFS $\mathbf{x}^*$.
3. Round $\mathbf{x}^*$ to obtain an integer solution that is within $\pm\frac{\sum_{j=1}^{n} |c_j|}{2}$ of the optimum. The rounding is performed as follows. For each variable x_i that has an odd half-integral value x_i^*, we check whether adding the constraint $x_i = \lceil x_i^* \rceil$ to $\mathbf{P}$ preserves lattice point feasibility. If not, we set x_i to $\lfloor x_i^* \rfloor$ and iterate, picking another variable to round, until we have obtained a feasible integer solution.

It is easy to see that each step can be performed in polynomial time. Notice that if lattice point feasibility is preserved by setting x_i either to $\lceil x_i^* \rceil$ or to $\lfloor x_i^* \rfloor$, the direction of rounding can be chosen heuristically to obtain a tighter approximation.

Our approximation theorem is general, in that it applies to any UTVPI integer program, including non 0-1 programs with *arbitrary coefficients* in the objective function. However, the approximation factor is additive, and the result is more likely to be useful for non 0-1 programs. In contrast, the results of Hochbaum et al. [17] guarantee a 2-approximation for UTVPI integer programs expressed as a minimization problem where the objective function is required to have non-negative coefficients.

4. Experimental Evaluation

We now present experimental results demonstrating that a decision procedure based on the solution bound derived in this paper can be competitive with other state-of-the-art procedures.

4.1 Implementation

We implemented a SAT-based decision procedure based on *finite instantiation*. The procedure operates in three steps.

First, given a UTVPI formula ϕ, it computes the enumeration bound $n \cdot (b_{\max} + 1)$. Second, it translates the input UTVPI formula to a Boolean formula by replacing each integer variable by a finite-precision, signed bit-vector that can take any value in the range $[-n \cdot (b_{\max} + 1),\ n \cdot (b_{\max} + 1)]$. Arithmetic and relational operators are then encoded as arithmetic circuits and comparators. Our implementation encodes addition using ripple carry adders and relational operators using the standard comparators for signed integer equality and less-than comparison (e.g., in the latter case, the sign bit is compared first and then the remaining bits); changing the encoding of arithmetic circuits was found not to have any significant impact on run time.

Let ϕ_{bool} denote the resulting Boolean formula. Clearly, ϕ_{bool} is satisfiable if and only if ϕ is satisfiable. Thus, the final step consists of invoking a SAT solver on ϕ_{bool}. Notice that the translation to SAT takes polynomial time and that the size of ϕ_{bool} is polynomial in that of ϕ.

Note that the derived solution bounds can be used in other ways. For instance, the numeric bounds can be imposed on variables and conjoined with the rest of the formula before invoking a theorem prover on it.

Our decision procedure was implemented in UCLID [6], which is written in Moscow ML, a dialect of Standard ML. For our experiments, UCLID invokes the MiniSat SAT solver [12]. Note that any alternative SAT solver can be employed just as easily.

4.2 Setup

Our benchmarks include the "RTCL" family of SMT-LIB benchmarks, 36 in all, most of which are unsatisfiable. Even though all but two of these benchmarks do not contain sum constraints (i.e., all but two are purely difference logic formulas), we retained the full set of benchmarks for completeness. However, as mentioned later in this section, these benchmarks are very easy to solve. Therefore, our experimental evaluation also includes a set of randomly generated UTVPI formulas.

We generated the random UTVPI formulas as "circuits" involving Boolean operators where the inputs to the circuit are UTVPI constraints rather than being Boolean variables. Each formula was generated based on 3 parameters: the maximum number of variables, an upper bound on the size of the constant term, and the maximum depth of the circuit. We varied the maximum number of variables over the set $\{500, 1000\}$, the constant term upper bound over the set $\{16, 256, 4096, 65536\}$, and the maximum circuit depth over $\{14, 15\}$. (The parameter ranges were selected so that the run-times of the solvers varied from a few seconds to several minutes.) For each choice of these three parameters, we generated a formula using one of two different random seeds; the seed was used in generating, at each level in the circuit, either a randomly chosen Boolean operator or a UTVPI constraint. The variables and constant term in each UTVPI constraint were randomly generated as well. This procedure generated a total of 32 formulas of which 12 are unsatisfiable. [3.]

3. All random benchmarks are available in SMT format at http://www.cs.cmu.edu/~uclid/utvpi-smt.tgz

We compared UCLID against the top three decision procedures in integer linear arithmetic category of the recent SMT-COMP 2006 competition: Yices [11], MathSAT [5], and Ario [31]. Experiments were run on a Linux workstation with a 2.8 GHz Xeon processor and 2 GB of RAM. A timeout of 1800 seconds was imposed on each run.

4.3 Comparison with Yices

We give a detailed comparison with Yices, the winning solver in the 2006 SMT-COMP competition, as statistics on its usage were easily generated for further analysis. Yices uses a Simplex-based solver for integer linear constraints within a DPLL(T) framework [10].

Figure 1 compares UCLID's total time (time for both encoding and SAT solving) to that taken by Yices. In each plot, the y-coordinate of a point is the time taken by UCLID, and the x-coordinate is the time taken by Yices. We also plot the diagonal line $y = x$ in each plot. Thus, points below the diagonal correspond to benchmarks on which UCLID outperforms Yices, while points above it correspond to benchmarks on which UCLID is outperformed. The points corresponding to Random benchmarks are indicated by a star while those corresponding to RTCL SMT-LIB benchmarks are indicated by a circle.

On all but two RTCL benchmarks (these being both difference logic formulas), Yices outperforms UCLID. Note that these benchmarks are extremely easy to solve, with runtimes for both solvers being less than a second on each.

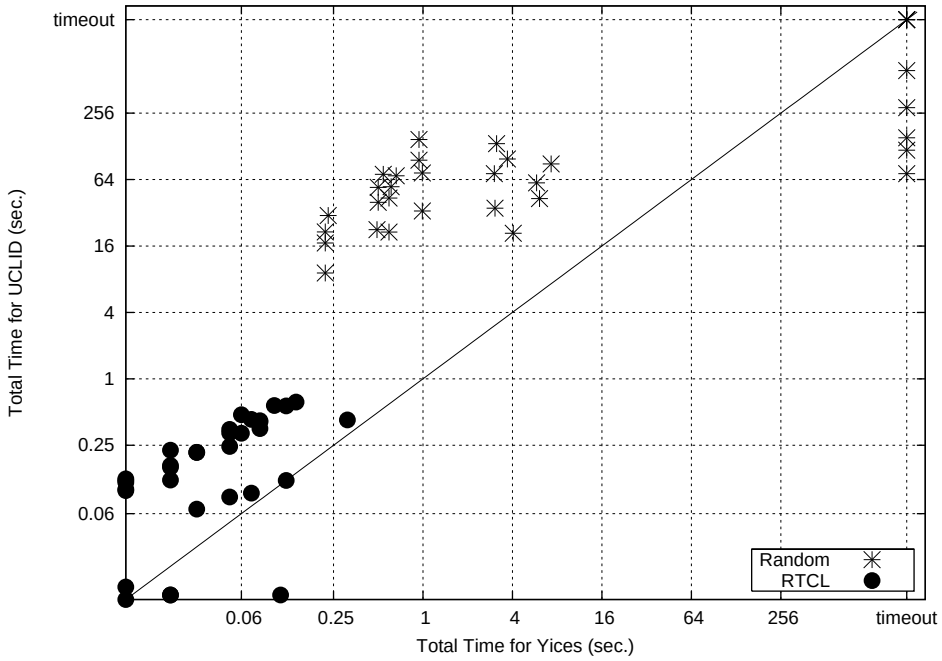

Figure 1. Experimental comparison of UCLID versus Yices for UTVPI formulas. Note the log scale on both axes. The timeout is 1800 seconds.

Next, consider the results on randomly generated benchmarks.

Observe that there are two distinct clusters of benchmarks. The first cluster, on the left, comprises 24 benchmarks on which Yices outperforms UCLID, in many cases by two orders of magnitude. These benchmarks include all 12 unsatisfiable formulas. On re-running Yices again on these examples with the "statistics generation flag" turned on, we observed that the number of conflicts reported for these benchmarks is extremely small, ranging from 1 to a few hundred. UCLID's run-time on these benchmarks is on the order of a few seconds to a few minutes; the encoding times and SAT times are comparable in most cases.

The cluster on the right, comprising the remaining 8 satisfiable formulas, behaves quite differently. Yices times out on each of these formulas. UCLID is able to finish within the timeout on 5 of these, with the largest run-time being 622 seconds of which 480 seconds is spent in SAT solving. As we did for the preceding cluster, we re-ran Yices with statistics generation turned on. This time, we found that the number of conflicts was about $50,000$ while the number of decisions ranged close to 10 million. We also learned that Yices' Simplex-based solver [10] performs a lot of pivoting operations for these benchmarks.[4.]

To summarize, while Yices outperforms UCLID on the majority of Random benchmarks, there are some examples where UCLID's enumerative approach works significantly better. The Yices run-times appear to track the number of decisions and conflicts, indicating that the main difficulty lies in enumerating a polyhedron with a feasible integer point. The large number of pivots on some benchmarks indicates that, for those instances, finding a feasible integer solution once the right polyhedron is enumerated can also be non-trivial. UCLID's SAT-based finite instantiation approach focuses the search on models within the solution bound, and the SAT solver can learn clauses that not only rule out infeasible combinations of UTVPI constraints, but also more complex relations amongst values to variables. It would be interesting to combine the two approaches, mixing a UCLID-style search over models with a Yices-style search for a lattice point feasible polyhedron.

4.4 Comparison with other solvers

We now describe the experimental results comparing UCLID's performance to that of MathSAT [5] and Ario [31] on both RTCL and Random benchmarks.

The MathSAT solver for linear arithmetic uses an incremental and layered approach [4], wherein satisfiability of partial assignments to Boolean-valued predicates (linear constraints) is checked by solvers in theories of increasing expressiveness (in order, equalities, difference logic, linear arithmetic over reals, and finally integer linear arithmetic). The Ario solver integrates a SAT engine and two linear arithmetic solvers: one dedicated to solving UTVPI constraints, while the other handles non-UTVPI linear constraints [32]. The solver for UTVPI constraints uses a transitive-closure algorithm and is tightly integrated with the SAT engine to provide implications and conflict-induced learned constraints.

The comparison of UCLID's run-times with MathSAT and Ario SMT solvers is shown in Figures 2 and 3 respectively, using the same format as in Figure 1. In both cases, UCLID's enumerative approach outperforms the competing solver on most of the Random and RTCL benchmarks. MathSAT's performance on the majority of these benchmarks is significantly better than that of Ario; the latter is the only solver that takes more than a few seconds

4. Personal communication with B. Dutertre and N. Shankar.

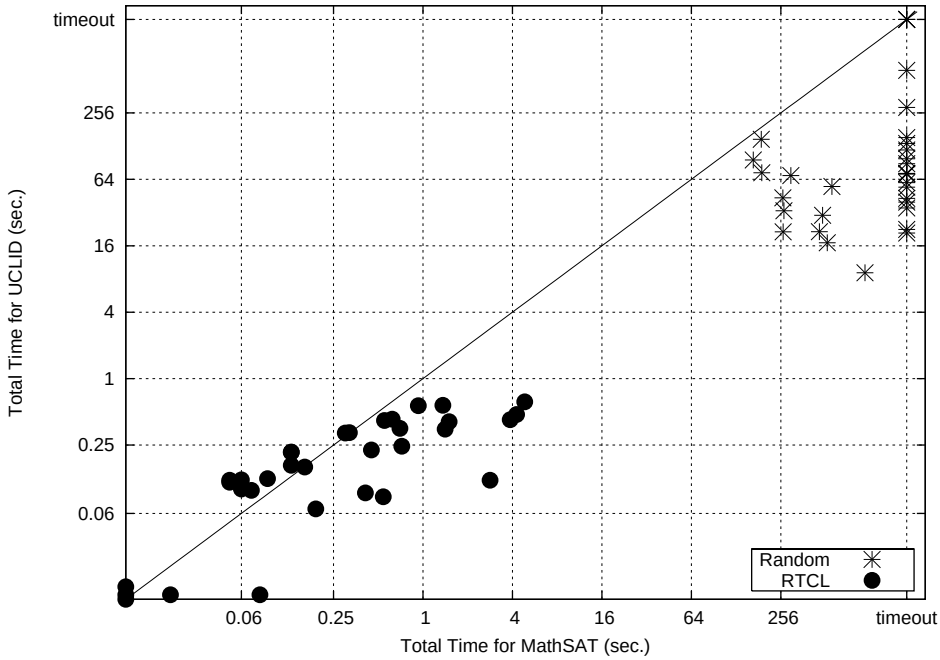

Figure 2. Experimental comparison of UCLID versus MathSAT for UTVPI formulas. Note the log scale on both axes. The timeout is 1800 seconds.

on some of the RTCL benchmarks. Ario did not finish within the timeout on any of the unsatisfiable Random benchmarks, while there was no such pattern in the case of MathSAT.

It is rather surprising that Ario's dedicated UTVPI solver does not help very much; one reason might be the overhead of communication between the UTVPI and SAT engines for unsatisfiable formulas over many UTVPI constraints. We believe that Yices' careful design of the Simplex-based solver is a major reason for its superior performance on these benchmarks compared to MathSAT and Ario. In particular, one innovative idea in Yices that seems effective is the decomposition of the linear arithmetic formula into a homogeneous system of linear equations and a formula only over single-variable inequalities [10]. This simplification technique results in far fewer changes to the Simplex tableau as the search over Boolean (abstract) models proceeds, and also permits a significant reduction in the size of the tableau using Gaussian elimination.

5. Conclusion

We have proposed a new approach to deciding the satisfiability of Boolean combinations of UTVPI constraints. The central insight is that it is sufficient to search for bounded solutions, where each variable is restricted within the finite range $[-n \cdot (b_{\max}+1),\, n \cdot (b_{\max}+1)]$. The solution bound we derive improves over previous results by an exponential factor. The key step in our derivation is a novel result for UTVPI polyhedra on finding integer solutions

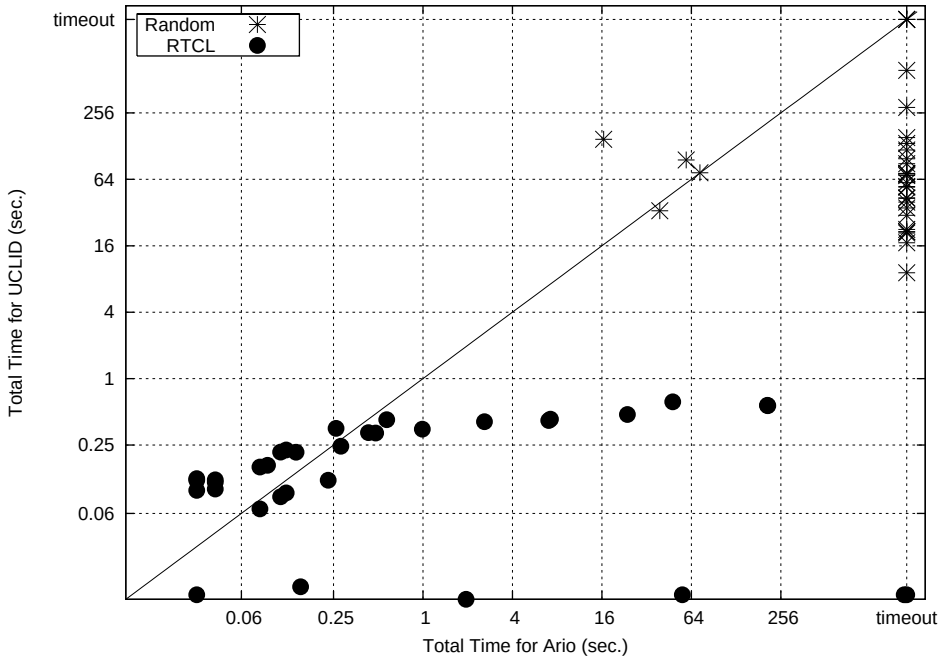

Figure 3. Experimental comparison of UCLID versus Ario for UTVPI formulas. Note the log scale on both axes. The timeout is 1800 seconds.

by rounding minimal face solutions. Experiments demonstrate the efficacy of a SAT-based decision procedure based on our theoretical results.

It would be interesting to extend our results to Boolean combinations of linear constraints that comprise mostly UTVPI constraints. Previous work [30] has shown that, for Boolean combinations of mostly difference constraints, the exponential term in the solution bound depends only on the number and coefficients of the non-difference constraints. It is still open as to whether a similar result can also be obtained for formulas of mostly UTVPI constraints.

Acknowledgments

We thank the anonymous reviewers for their many valuable comments, and Bryan Brady for help with translating benchmarks. We are also grateful to Bruno Dutertre and Natarajan Shankar for providing insights into Yices' performance. This research was supported in part by the U.S. Army under ARO grant DAAD19-01-1-0485, by the Air Force Office of Scientific Research under grant FA9550-06-1-0050, and by SRC contract 1355.001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. Government, or any other entity.

References

[1] T. Ball, B. Cook, S. Lahiri, and L. Zhang. Zapato: Automatic theorem proving for predicate abstraction refinement. In *Proc. Computer-Aided Verification (CAV)*, *LNCS* **3114**, pages 457–461, 2004.

[2] S. Berezin, V. Ganesh, and D. L. Dill. An online proof-producing decision procedure for mixed-integer linear arithmetic. In *Proc. Tools and Algorithms for the Construction and Analysis of Systems (TACAS'03)*, LNCS **2619**, pages 521–536, April 2003.

[3] I. Borosh and L. B. Treybig. Bounds on positive integral solutions of linear Diophantine equations. *Proceedings of the American Mathematical Society*, **55**(2):299–304, March 1976.

[4] Marco Bozzano, Roberto Bruttomesso, Alessandro Cimatti, Tommi A. Junttila, Peter van Rossum, Stephan Schulz, and Roberto Sebastiani. MathSAT: Tight integration of SAT and mathematical decision procedures. *Journal of Automated Reasoning*, **35**(1-3):265–293, 2005.

[5] Roberto Bruttomesso, Alessandro Cimatti, Anders Franzen, Alberto Griggio, and Roberto Sebastiani. System description: MathSAT 3.4. http://www.csl.sri.com/users/demoura/smt-comp/descriptions/mathsat.ps, August 2006.

[6] R. E. Bryant, S. K. Lahiri, and S. A. Seshia. Modeling and verifying systems using a logic of counter arithmetic with lambda expressions and uninterpreted functions. In E. Brinksma and K. G. Larsen, editors, *Proc. Computer-Aided Verification (CAV'02)*, LNCS **2404**, pages 78–92, July 2002.

[7] Clark Barrett, Cesare Tinelli, et al. SMT-COMP 2006 entry: CVC3. http://www.csl.sri.com/users/demoura/smt-comp/descriptions/CVC3-description.pdf, August 2006.

[8] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*, chapter 24, pages 602–604. MIT Press, second edition, 2001.

[9] G. B. Dantzig and B. C. Eaves. Fourier-Motzkin elimination and its dual. *Journal of Combinatorial Theory A*, **14**:288–297, 1973.

[10] Bruno Dutertre and Leonardo de Moura. A Fast Linear-Arithmetic Solver for DPLL(T). In *Proceedings of the 18th Conference on Computer-Aided Verification (CAV)*, *LNCS* **4144**, pages 81–94, 2006.

[11] Bruno Dutertre and Leonardo de Moura. The Yices SMT solver. http://yices.csl.sri.com/tool-paper.pdf, September 2006.

[12] Niklas Eén and Armin Biere. Effective preprocessing in SAT through variable and clause elimination. In *International Conference on Theory and Applications of Satisfiability Testing (SAT)*, *LNCS* **3569**, pages 61–75, June 2005.

[13] Harald Ganzinger, George Hagen, Robert Nieuwenhuis, Albert Oliveras, and Cesare Tinelli. DPLL(T): Fast decision procedures. In *Proceedings of the International Conference on Computer-Aided Verification (CAV)*, pages 175–188, July 2004.

[14] Warwick Harvey and Peter J. Stuckey. A unit two variable per inequality integer constraint solver for constraint logic programming. In *Proceedings of the Twentieth Australasian Computer Science Conference*, pages 102–111, 1997.

[15] D. Hochbaum. *Approximation Algorithms for NP-Hard Problems*, chapter 3. PWS Publishing Company, 1995.

[16] D. Hochbaum and J. Naor. Simple and fast algorithms for linear and integer programs with two variables per inequality. *SIAM Journal on Computing*, **23**(6):1179–1192, 1994.

[17] Dorit Hochbaum, N. Megiddo, J. Naor, and A. Tamir. Tight bounds and 2-approximation algorithms for integer programs with two variables per inequality. *Mathematical Programming*, **62**:63–92, 1993.

[18] Joxan Jaffar, Michael J. Maher, Peter J. Stuckey, and Roland H. C. Yap. Beyond finite domains. In *2nd International Workshop on Principles and Practice of Constraint Programming (PPCP'94)*, *LNCS* **874**, pages 86–94, 1994.

[19] R. Kannan and C. L. Monma. On the computational complexity of integer programming problems. In *Optimisation and Operations Research*, *Lecture Notes in Economics and Mathematical Systems* **157**, pages 161–172, 1978.

[20] Shuvendu K. Lahiri and Madanlal Musuvathi. An efficient decision procedure for UTVPI constraints. In *5th International Workshop on Frontiers of Combining Systems (FroCoS)*, *LNCS* **3717**, pages 168–183, 2005.

[21] Paulo J. Matos. System description: ExtSat 1.1 revisited. http://www.csl.sri.com/users/demoura/smt-comp/descriptions/extsat_description.pdf, August 2006.

[22] Antoine Miné. The octagon abstract domain. *Higher-Order and Symbolic Computation*, **19**:31–100, 2006.

[23] G. Nelson and D. C. Oppen. Simplification by cooperating decision procedures. *ACM Transactions on Programming Languages and Systems*, **1**(2):245–257, 1979.

[24] G. Nemhauser and L. Wolsey. *Integer and Combinatorial Optimization*, chapter I.4: Polyhedral Theory. Wiley-Interscience, New York, 1988.

[25] R. Nieuwenhuis and A. Oliveras. Decision Procedures for SAT, SAT Modulo Theories and Beyond. The BarcelogicTools. In G. Sutcliffe and A. Voronkov, editors, *12h International Conference on Logic for Programming, Artificial Intelligence and Reasoning, LPAR'05*, *LNCS* **3835**, pages 23–46, 2005.

[26] R. Nieuwenhuis and A. Oliveras. DPLL(T) with Exhaustive Theory Propagation and its Application to Difference Logic. In K. Etessami and S. Rajamani, editors, *17th International Conference on Computer Aided Verification, CAV'05*, *LNCS* **3576**, pages 321–334, 2005.

[27] Christos H. Papadimitriou. On the complexity of integer programming. *Journal of the ACM*, **28**(4):765–768, 1981.

[28] Kenneth Roe. The heuristic theorem prover SMT-COMP'06 submission. http://www.csl.sri.com/users/demoura/smt-comp/descriptions/htp.pdf, August 2006.

[29] Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley and Sons, 1986.

[30] Sanjit A. Seshia and Randal E. Bryant. Deciding quantifier-free Presburger formulas using parameterized solution bounds. *Logical Methods in Computer Science*, **1**(2):1–26, December 2005.

[31] Hossein Sheini and Karem Sakallah. Ario: A linear integer arithmetic solver. http://www.csl.sri.com/users/demoura/smt-comp/descriptions/ario.pdf, August 2006.

[32] Hossein M. Sheini and Karem A. Sakallah. A scalable method for solving satisfiability of integer linear arithmetic logic. In *8th International Conference on Theory and Applications of Satisfiability Testing (SAT)*, *LNCS* **3569**, pages 241–256, 2005.

[33] K. Subramani. On deciding the non-emptiness of 2SAT polytopes with respect to first order queries. *Mathematical Logic Quarterly*, **50**(3):281–292, 2004.

[34] J. von zur Gathen and M. Sieveking. A bound on solutions of linear integer equalities and inequalities. *Proceedings of the American Mathematical Society*, **72**(1):155–158, October 1978.