# Tight Bounds and 2-Approximation Algorithms for Integer Programs with Two Variables per Inequality

Dorit S. Hochbaum* Nimrod Megiddo† Joseph Naor‡ Arie Tamir§

December 1991; revised October 1992

**Abstract.** The problem of integer programming in bounded variables, over constraints with no more than two variables in each constraint is NP-complete, even when all variables are binary. This paper deals with integer linear minimization problems in $n$ variables subject to $m$ linear constraints with at most two variables per inequality, and with all variables bounded between 0 and $U$. For such systems, a 2−approximation algorithm is presented that runs in time $O(mnU^2 \log(Un^2/m))$, so it is polynomial in the input size if the upper bound $U$ is polynomially bounded. The algorithm works by finding first a super-optimal feasible solution that consists of integer multiples of $\frac{1}{2}$. That solution gives a tight bound on the value of the minimum. It further more has an identifiable subset of integer components that retain their value in an integer optimal solution of the problem. These properties are a generalization of the properties of the vertex cover problem. The algorithm described is, in particular, a 2-approximation algorithm for the problem of minimizing the total weight of true variables, among all truth assignments to the 2-satisfiability problem.

*Department of Industrial Engineering and Operations Research and Walter A. Haas School of Business, 350 Barrows Hall, University of California, Berkeley, CA 94720 USA. Research supported in part by ONR contracts N00014-88-K-0377 and N00014-91-J-1241.

†IBM Almaden Research Center, 650 Harry Road, San Jose, California 95120-6099, and School of Mathematical Sciences, Tel Aviv University, Tel Aviv, Israel. Research supported in part by ONR contract N00014-91-C-0026.

‡Faculty of Computer Science, Technion, Israel Institute of Technology, Haifa 32000, Israel. Part of this work was done while the author was visiting the International Computer Science Institute in Berkeley, CA and DIMACS, Rutgers University, New Brunswick, NJ.

§School of Mathematical Sciences, Tel Aviv University, Tel Aviv, Israel, and IBM Almaden Research Center, San Jose, California.

## 1. Introduction

The formulation of the integer programming problem discussed hereafter is as follows:

$$\begin{aligned} &\text{Minimize} \quad \sum_{j=1}^{n} w_j x_j \\ &\text{subject to} \quad a_i x_{j_i} + b_i x_{k_i} \geq c_i \quad (i = 1, \ldots, m) \\ &\qquad\qquad\quad 0 \leq x_j \leq u_j \qquad (j = 1, \ldots, n) \\ &\qquad\qquad\quad x_j \text{ integer} \qquad (j = 1, \ldots, n) \end{aligned} \tag{IP}$$

where $1 \leq j_i, k_i \leq n$, $w_i \geq 0$ $(i = 1, \ldots, n)$, and all the coefficients are integer.

Whereas an integer programming problem with two variables is in the class of polynomial problems [7, 10], the problem (IP) is NP-complete – the vertex cover problem ([8]) is a special case with $a_i = b_i = c_i = u_i = 1$. Although the vertex cover problem is a limited special case of the problem (IP), many of its known properties extend also to the problem (IP). The vertex cover problem, and its complement – the independent set problem, have several structural properties discovered by Nemhauser and Trotter [12]: Consider the linear programming relaxation of the vertex cover problem, (VCR), namely,

$$\begin{aligned} \sum_{j=1}^{n} w_j x_j^* = \ &\text{Min} \quad \sum_{j=1}^{n} w_j x_j \\ &\text{subject to} \quad x_i + x_j \geq 1 \quad (\text{for every edge } (i, j) \text{ in the graph}) \\ &\qquad\qquad\quad 0 \leq x_j \leq 1 \quad (j = 1, \ldots, n). \end{aligned} \tag{VCR}$$

Then, there exists an optimal solution $\boldsymbol{x}^*$ such that $x_j^* \in \{0, 1, \frac{1}{2}\}$ (this was previously observed by Balinski). In addition, there exists an optimal integer solution that is equal to $\boldsymbol{x}^*$ in its integer components. In particular, it means that an optimal integer solution may be obtained by rounding the components of $\boldsymbol{x}^*$ that are equal to $\frac{1}{2}$. The rounding could be up or down to 1 or 0 respectively. Since there are $2^n$ possible rounding schemes (some of which may not lead to a feasible solution) this fact in itself does not aid in speeding up the search for an optimal solution.

The fact that some solutions to (VCR) consist of integer multiples of $\frac{1}{2}$ is, however, useful in developing a 2-approximation algorithm for the vertex cover problem, by rounding *up* the linear programming relaxation solution. The results described here extend many of the properties of the vertex cover problem to any integer programming problem with two variables per inequality. Specifically, we show how to find in polynomial time a feasible solution all the components of which are integer multiples of $\frac{1}{2}$. Moreover this solution is optimal among a set of solutions to be defined later. This set contains all integer solutions and is contained in the set of all feasible solutions that are integer multiples of $\frac{1}{2}$. (As demonstrated later, finding an optimal solution over the set of feasible solutions that are integer multiples of $\frac{1}{2}$ is an NP-hard problem). In contrast to the vertex cover problem, an optimal solution to the linear programming relaxation of (IP), called (IPR), is a fractional vector that is often *not* an integer multiple of $\frac{1}{2}$. Whereas for the vertex cover problem, the absolute value of all nonseparable subdeterminants of the constraint matrix is bounded by 2, (this explains why a linear programming relaxation solution that is basic, is an integer multiple of $\frac{1}{2}$), this is not the case for the problem (IP), where the subdeterminants of the constraint matrix may assume arbitrarily large values. It is therefore somewhat surprising that relaxed solutions that are integer multiples of $\frac{1}{2}$ can be obtained for (IPR) in what amounts to essentially the same computational effort as that of solving (VCR).

In order to find solutions that are integer multiples of $\frac{1}{2}$, we use the polynomial time algorithm for solving integer programs in bounded variables over *monotone* inequalities proposed in [6]. An inequality in two variables is called *monotone* if it is of the form

$$ax_{j_i} - bx_{k_i} \geq c$$

where $a$ and $b$ are both nonnegative. Although, as proved by Lagarias [9], the problem of finding a feasible solution of a system of monotone inequalities in integers is NP-complete, the algorithm of Hochbaum and Naor [6] finds an optimal solution in time $O(mnU^2 \log(Un^2/\ m))$. Since our (IP) is not necessarily defined on monotone inequalities, we use a transformation of nonmonotone inequalities to monotone inequalities proposed by Edelsbrunner, Rote and Welzl [2]. The transformation does not preserve integrality, yet each solution to the transformed problem corresponds to a feasible solution of the original problem, and in addition it consists of integer multiples of $\frac{1}{2}$.

The problem of finding a feasible solution of a system of inequalities in integers is NP-complete. The problem (IP) with two variables per inequality has, however, among its many remarkable properties also the property that a feasible solution can be identified in $O((n+m)U)$ time. Such an algorithm, based on ideas of T. Feder, is presented in Section 4. This algorithm is used for obtaining a feasible solution from the solution consisting of integer multiples of $\frac{1}{2}$.

A common method for deriving (lower) bounds for integer programming problems is by solving a linear programming relaxation. A notable feature of the linear programming relaxation of (IPR) of (IP) is that *feasible* solutions can be derived in strongly polynomial time. Megiddo [11] was the first to discover such an algorithm. The fastest algorithms currently known for the problem are by Cohen and Megiddo [1] and by Hochbaum and Naor [6], with running times of $O(mn^2(\log^2 n + \log m))$ and $O(mn^2 \log m)$, respectively. The algorithm presented here is a relaxation which is tighter than the one obtained by the LP-relaxation. In case the bounds $U$ are fixed, the relaxation here is also obtained in faster running time.

As noted in [12], the LP-relaxation of the vertex cover problem (VCR) is solved by finding an optimal cover in a bipartite graph with two vertices for each vertex in the original graph, and two edges for each edge in the original graph. In a bipartite graph a vertex cover may be identified from the solution of a corresponding minimum cut problem. Our algorithm is also a minimum cut algorithm applied to a graph with $nU$, (or rather, $\sum_{j=1}^{n} u_j$) nodes and $mU$ arcs.

The other property of the vertex cover problem, namely, that for any optimal solution to (VCR) there exists an optimal solution $\boldsymbol{z}^*$ of the vertex cover problem that coincides with $\boldsymbol{x}^*$ in every integer component ([12]), is useful in reducing the size of the problem. This property can be used for instance in branch-and-bound procedures. This property has been useful in deriving a whole class of approximation algorithms for the vertex cover problem, all of which are twice the optimum or less, ([5]). The same approach, of fixing those variables that retain their value in an optimal solution, may be used for (IP) with the potential for developing tighter worst-case error bounds for special classes of instances.

The feasibility problem of systems of linear inequalities in *binary* variables with at most two variables per inequality is closely related to the 2-satisfiability problem (2-SAT). An instance of the latter is a conjunction of $q$ disjunctions in $p$ boolean variables, each disjunction having one of the forms: (i) $x_i \vee x_j$, (ii) $x_i \vee \bar{x}_j$, and (iii) $\bar{x}_i \vee \bar{x}_j$. A feasible solution for this problem can be found in linear time [3]. The corresponding system of linear inequalities in binary variables consists of constraints of one of the following types: (i) $x_i + x_j \geq 1$, (ii) $x_i \geq x_j$, and (iii) $x_i + x_j \leq 1$. The problem of minimizing a linear function in binary variables subject to such a system of linear inequalities will be referred to as the 2-SAT integer programming problem.

Recently Gusfield and Pitt [4] described a 2-approximation algorithm for the 2-SAT problem. Their approach is not related to ours and yields neither a lower bound nor the option of fixing some of the variables, as our algorithm does.

In Section 2 we review the procedure of reducing (IP) to a monotone system. In Section 3 we describe the algorithm for optimizing over a monotone system. In Section 4 we present a polynomial algorithm for finding a feasible solution to (IP) or verifying that none exists. Section 5 explains how to generate a 2-approximation solution to (IP) from the optimal solution to the monotone system and a feasible solution to (IP). Finally, in Sections 6 and 7 we discuss properties of 2-SAT systems and their generalizations.

## 2. The reduction to a monotone system and its properties

Consider a generic nonmonotone inequality of the form $ax + by \geq c$ where $a$ and $b$ are positive. (Any nonmonotone inequality can be written in this form, perhaps with a reversed inequality). The procedure in [2] replaces each variable $x$ by two variables, $x^+$

and $x^-$, and each inequality by two inequalities as follows:

$$ax^+ - by^- \geq c$$
$$-ax^- + by^+ \geq c\ .$$

The two resulting inequalities are monotone. Note that upper and lower bounds constraints $\ell_j \leq x_j \leq u_j$ are transformed to

$$\ell_j \leq x_j^+ \leq u_j$$
$$-u_j \leq x_j^- \leq -\ell_j\ .$$

In the objective function, the variable $x$ is substituted by $\frac{1}{2}(x^+ - x^-)$.

Monotone inequalities remain so by replacing the variables $x$ and $y$ in one inequality by $x^+$ and $y^+$, and in the second, by $x^-$ and $y^-$, respectively.

Let $\boldsymbol{A}$ be the matrix of the constraints in the original system and let $\boldsymbol{A}^{(2)}$ be the matrix of the monotone system resulting from the above transformation. The matrix $\boldsymbol{A}^{(2)}$ consists of $2m$ inequalities with two variables per inequality, and $2n$ upper and lower bound constraints. The order of this matrix is therefore $(2m + 4n) \times 2n$.

Given a system of inequalities with two variables per inequality, let the set of feasible solutions for this system be

$$S = \{\boldsymbol{x} \in \Re^n \mid \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{c}\}\ ,$$

and the feasible solutions to the monotone system resulting from the transformation above,

$$S^{(2)} = \{(\boldsymbol{x}^+, \boldsymbol{x}^-) \mid \boldsymbol{A}^{(2)}(\boldsymbol{x}^+, \boldsymbol{x}^-) \leq \boldsymbol{c}^{(2)}\ ,\ \boldsymbol{x}^+, \boldsymbol{x}^- \in \Re^n\}\ .$$

If $\boldsymbol{x} \in S$, $\boldsymbol{x}^+ = \boldsymbol{x}$, and $\boldsymbol{x}^- = -\boldsymbol{x}$, then $(\boldsymbol{x}^+, \boldsymbol{x}^-) \in S^{(2)}$. So, for every feasible solution in $S$, there exists a feasible solution in $S^{(2)}$. Conversely, if $(\boldsymbol{x}^+, \boldsymbol{x}^-) \in S^{(2)}$, then $\boldsymbol{x}^{(2)} = \frac{1}{2}(\boldsymbol{x}^+ - \boldsymbol{x}^-) \in S$. Hence, for every feasible solution in $S^{(2)}$, there is a feasible solution in $S$.

Let $S_I = \{\boldsymbol{x} \in S \mid \boldsymbol{x} \text{ integer }\}$, and let

$$S_I^{(2)} = \left\{\tfrac{1}{2}(\boldsymbol{x}^+ - \boldsymbol{x}^-) \mid (\boldsymbol{x}^+, \boldsymbol{x}^-) \in S^{(2)} \text{ and } \boldsymbol{x}^+, \boldsymbol{x}^- \text{ integer }\right\}\ .$$

If $\boldsymbol{x} \in S_I$, then $\boldsymbol{x} \in S_I^{(2)}$. Thus, $S_I \subseteq S_I^{(2)} \subseteq S$.

In fact, the set of solutions $S_I^{(2)}$ is even smaller than the set of feasible solutions that are integer multiples of $\frac{1}{2}$. To see that, let

$$S^{(\frac{1}{2})} = \{\boldsymbol{x} \mid \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{c} \text{ and } \boldsymbol{x} \in \tfrac{1}{2}Z^n\}\ .$$

Then, the claim is that $S_I^{(2)} \subset S^{(\frac{1}{2})}$, yet $S^{(\frac{1}{2})}$ may contain points not in $S_I^{(2)}$. The following example illustrates such a case:

$$\begin{aligned} &5x + 2y \le 6 \\ &0 \le x, y \le 1 \ . \end{aligned}$$

Obviously, $(x = 1,\ y = \frac{1}{2})$ is a feasible solution in $S^{(\frac{1}{2})}$. But there is no corresponding integer solution in $S^{(2)}$ as $x^+ = -x^- = 1$ implies that $y^+ = y^- = 0$. It follows that the bound derived from optimizing over $S_I^{(2)}$ is tighter than a bound derived from optimizing over $S^{(\frac{1}{2})}$. Not only is this latter optimization weaker, but it is also in general NP-hard. To see that this is NP-hard, we use a reduction from the vertex cover problem proposed by O. Goldschmidt. Consider a vertex cover problem with nonnegative weights of nodes,

$$\begin{aligned} \text{Minimize} \quad & \sum_{j=1}^{n} w_j x_j \\ \text{subject to} \quad & x_{j_i} + x_{k_i} \ge 1 \quad (i = 1, \ldots, m) \\ & 0 \le x_j \le 1 \ , \ x_j \text{ integer} \quad (j = 1, \ldots, n) \end{aligned} \tag{VC}$$

(where $1 \le j_i, k_i \le n$). Due to the nonnegativity of $w_j$, the problem does not change if we increase the bounds to 2:

$$\begin{aligned} \text{Minimize} \quad & \sum_{j=1}^{n} w_j x_j \\ \text{subject to} \quad & x_{j_i} + x_{k_i} \ge 1 \quad (i = 1, \ldots, m) \\ & 0 \le x_j \le 2 \ , \ x_j \text{ integer} \quad (j = 1, \ldots, n) \ . \end{aligned}$$

Using the substitution $y_j = \frac{1}{2} x_j$, it is evident that this problem is equivalent to the following problem in which the integrality requirement is replaced by the restriction that the variables are integer multiples of $\frac{1}{2}$:

$$\begin{aligned} \text{Minimize} \quad & \sum_{j=1}^{n} w_j y_j \\ \text{subject to} \quad & 2y_{j_i} + 2y_{k_i} \ge 1 \quad (i = 1, \ldots, m) \\ & 0 \le y_j \le 1 \ , \ y_j \text{ is an integer multiple of } \tfrac{1}{2} \quad (j = 1, \ldots, n) \ . \end{aligned}$$

Hence, to solve in integer multiples of $\frac{1}{2}$ is at least as difficult as to solve the vertex cover problem.

An intuitive explanation of the difficulty of these problems and of the relative weakness of the LP-relaxation bound is that the coefficients of the variables in the constraints are "unnecessarily" large. The reduction we use (of [6]) effectively eliminates such large coefficients and substitutes them by ones of absolute value 1, at the expense of increasing the number of inequalities and variables by a factor of $U$.

## 3. Integer optimization over monotone inequalities

Hochbaum and Naor [6] describe an algorithm for optimization in integers over a system of *monotone* inequalities,

$$\begin{aligned} &\text{Minimize} && \sum_{j=1}^{n} d_j x_j \\ &\text{subject to} && a_i x_{j_i} - b_i x_{k_i} \geq c_i \quad (i = 1, \ldots, m) \\ & && \ell_j \leq x_j \leq u_j \ , \ x_j \text{ integer} \quad (j = 1, \ldots, n) \ , \end{aligned} \tag{IP2}$$

where $a_i, b_i, c_i$ $(i = 1, \ldots, m)$, and $d_j$ $(j = 1, \ldots, n)$ are rational, and $\ell_j$ and $u_j$ $(j = 1, \ldots, n)$ are integers. The coefficients $a_i$ and $b_i$ $(i = 1, \ldots, m)$ are nonnegative but the objective function coefficients $d_j$ $(j = 1, \ldots, n)$ may be negative. Note that in this section we allow nonzero lower bounds on the variables. The algorithm is reviewed here for the sake of completeness.

A directed graph $G$ is created where for each variable $x_j$ in the interval $[\ell_j, u_j]$, there are $u_j - \ell_j + 1$ nodes representing it, one for each integer value in the range. A set of nodes is said to be *closed* if it contains all the nodes that can be reached via a directed path from any node in the set. It is shown that a maximum weight closed set in this graph corresponds to an optimal solution of (IP2).

For each integer $k$ in the range, there is an arc $(k, k-1)$ from the node representing the value $k$ to the node representing the value $k-1$. The node representing $\ell_j$ has an arc directed to it from the source node $s$. Thus, if the source node is in a closed set then so are all $\ell_j$ nodes. The monotone inequalities are represented by arcs. For each potential value $k$ of variable $x_{k_i}$, all inequalities in which $x_{k_i}$ appears with negative coefficient impose a minimum value on the variable $x_{j_i}$ that appears in the same inequality with a positive coefficient,

$$x_{j_i} \geq \left\lceil \frac{b_i k + c_i}{a_i} \right\rceil = k_1 \ .$$

This is represented by an arc going from node $k$ of $x_{k_i}$ to node $k_1$ of $x_{j_i}$. If $k_1 > u_{j_i}$, then the value $k$ of the variable $x_{k_i}$ is infeasible, and the upper bound of $x_{k_i}$ is reset to $k-1$. A closed set containing $s$ corresponds to a feasible solution to (IP2) where the variable $x_j$ assumes the value of the largest node representing it in the closed set.

We now assign the node $\ell_j$ of variable $x_j$, the weight $-d_j \ell_j$, and all other nodes representing variable $x_j$ are assigned the weight $-d_j$. A maximum weight closed set corresponds then to an optimal solution to the minimization problem (IP2). The maximum closure in a graph is derived from solving a minimum cut problem in the graph after adding a source and a sink, placing arcs from the source to all nodes of positive weight with capacity equal to that weight, and placing arcs from all nodes with negative weight to the sink with capacity equal to the absolute value of that weight. All other arcs are assigned infinite capacity. The source set of a minimum cut in this graph corresponds to a maximum weight closed set with the weights as specified. The justification for the algorithm of maximum closure is given by Picard [13].

Notice that a 2-SAT formula can be generated from the graph $G$: there is a clause $y \vee \bar{x}$ corresponding to each directed edge $x \to y$. There is a 1-1 correspondence between the closed subsets in $G$ and the feasible solutions of the 2-SAT formula generated. The relationship between integer programs with two variables per inequality and 2-SAT formulae will be investigated more thoroughly in the next section.

## 4. A polynomial algorithm for the feasibility of (IP)

In this section we show that every bounded integer system with two variables per inequality can be equivalently written as a 2-SAT instance. We use an idea of T. Feder. Recall that for each variable $x_i$ we have $0 \le x_i \le u_i < \infty$ $(i = 1, \ldots, n)$. We replace each variable $x_i$ by $u_i$ binary variables $x_{i\ell}$ $(\ell = 1, \ldots, u_i)$, with the constraints $x_{i\ell} \ge x_{i,\ell+1}$ $(\ell = 1, \ldots, u_i - 1)$. Subject to these constraints, the correspondence between $x_i$ and the $u_i$-tuple $(x_{i1}, \ldots, x_{iu_i})$ is one-to-one and is characterized by $x_{i\ell} = 1$ if and only if $x_i \ge \ell$ $(\ell = 1, \ldots, u_i)$, or, equivalently, $x_i = \sum_{\ell=1}^{u_i} x_{i\ell}$.

We now explain how to transform the constraints of the given system into constraints in terms of the $x_{i\ell}$'s. Suppose

$$a_{ki}x_i \ + \ a_{kj}x_j \ \ge \ b_k$$

is one of the given constraints. There are several cases to be distinguished. Without loss of generality, assume both $a_{ki}$ and $a_{kj}$ are nonzeros. Consider the case where both are positive, and assume without loss of generality that $0 < b_k < a_{ki}u_i + a_{kj}u_j$. For every $\ell$ $(\ell = 0, \ldots, u_i)$, let

$$\alpha_{k\ell} = \left\lceil \frac{b_k - \ell a_{ki}}{a_{kj}} \right\rceil - 1 \ .$$

It is easy to see that for an integer solution $\boldsymbol{x}$, $a_{ki}x_i + a_{kj}x_j \ge b_k$ if and only if for every $\ell$ $(\ell = 0, \ldots, u_i)$,

$$\text{either} \ \ x_i > \ell \ \ \text{or} \ \ x_j > \alpha_{k\ell}$$

or, equivalently,

$$\text{either} \ \ x_i \ge \ell + 1 \ \ \text{or} \ \ x_j \ge \alpha_{k\ell} + 1 \ .$$

Under the above transformation between the $x_i$'s and the $x_{i\ell}$'s, this is equivalent to:

(i) For every $\ell$ $(\ell = 0, 1, \ldots, u_i - 1)$, if $0 \le \alpha_{k\ell} < u_j$, then either $x_{i,\ell+1} = 1$ or $x_{j,\alpha_{k\ell}+1} = 1$, and if $\alpha_{k\ell} \ge u_j$, then $x_{i,\ell+1} = 1$.

(ii) For $\ell = u_i$, if $\alpha_{ku_i} \ge 0$, then $x_{j,\alpha_{ku_i}+1} = 1$ (since we have $\alpha_{ku_i} < u_j$).

The disjunction in (i) can be written as

$$x_{i,\ell+1} + x_{j,\alpha_{k\ell}+1} \geq 1 \ .$$

Thus, altogether we have replaced one original constraint on $x_i$ and $x_j$ by at most $u_i + 1$ constraints on the variables $x_{i\ell}$ and $x_{j\ell}$. The other cases, corresponding to different sign combinations of $a_{ki}$, $a_{kj}$, and $b_k$, can be handled in a similar way.

If the above transformation is applied to a monotone system of inequalities, then the resulting 2-SAT integer program is also monotone.

To summarize, we replace the $n$ original variables and $m$ original constraints by $\bar{u} = \sum_{j=1}^{n} u_j$ new variables and at most $mU + \bar{u}$ new constraints, where $U = \max_i u_i$. The time bounds for finding a feasible solution are as follows.

**Lemma 4.1.** *A feasible solution to a bounded linear program with two variables per inequality can be computed in $O(m + n + \bar{u} + mU)$ time.*

*Proof:* A feasible solution to a 2-SAT integer program can be found in linear time using the algorithm of [3]. Encoding a bounded integer program as a 2-SAT integer program generates $\bar{u}$ variables and at most $mU + \bar{u}$ constraints. Hence, the time bounds follow. ∎

It is an open question whether a feasible solution can be computed in polynomial time for an *unbounded* integer program with two variables per inequality, where the degree of the polynomial may depend on $n$, $m$, and the largest integer in the matrix $[\boldsymbol{A}\ \boldsymbol{b}]$ represented in unary.

## 5. Computing an approximate solution

In this section we show how to obtain a 2-approximation for the optimum of a bounded integer program with two variables per inequality in $O(mnU^2 \log(Un^2/m))$ time. We assume that the given integer program has a feasible integer solution denoted by $z_1, \ldots, z_n$. (This can be tested using Lemma 4.1).

We first transform the integer program into a monotone integer system as outlined in Section 2 and compute an optimal solution for the monotone system as outlined in Section 3. For every variable $x_i$ ($i = 1, \ldots, n$), let $m_i^+$ and $m_i^-$ denote the respective values of $x_i^+$ and $x_i^-$ in the optimal solution of the monotone system. For $i = 1, \ldots, n$, let $m_i^* = \frac{1}{2}(m_i^+ - m_i^-)$. We define the following solution vector, denoted by $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_n)$, where for $i = 1, \ldots, n$:

$$\ell_i = \begin{cases} \min\{m_i^+, -m_i^-\} & \text{if} \quad z_i \leq \min\{m_i^+, -m_i^-\} \\ z_i & \text{if} \quad \min\{m_i^+, -m_i^-\} \leq z_i \leq \max\{m_i^+, -m_i^-\} \\ \max\{m_i^+, -m_i^-\} & \text{if} \quad z_i \geq \max\{m_i^+, -m_i^-\} \ . \end{cases}$$

**Lemma 5.1.** *The vector $\boldsymbol{\ell}$ is a feasible solution of the given integer program.*

*Proof:* Let $ax_i + bx_j \geq c$ be an inequality where $a$ and $b$ are non-negative. We check all possible cases. If $\ell_i$ is equal to $z_i$ or $\min\{m_i^+, -m_i^-\}$, and $\ell_j$ is equal to $z_j$ or $\min\{m_j^+, -m_j^-\}$, then clearly,

$$a\ell_i + b\ell_j \geq az_i + bz_j \geq c\ .$$

Suppose $\ell_i \geq z_i$ and $\ell_j = \max\{m_j^+, -m_j^-\}$. By construction, we know that

$$am_i^+ - bm_j^- \geq c \ \text{ and } \ -am_i^- + bm_j^+ \geq c\ .$$

If $\ell_i \geq -m_i^-$, then,

$$a\ell_i + b\ell_j \geq -am_i^- + bm_j^+ \geq c\ .$$

Otherwise,

$$a\ell_i + b\ell_j \geq am_i^+ - bm_j^- \geq c\ .$$

The last case is when $\ell_i = \max\{m_i^+, -m_i^-\}$, and $\ell_j = \max\{m_j^+, -m_j^-\}$. In this case,

$$a\ell_i + b\ell_j \geq \ am_i^+ - bm_j^- \geq c\ .$$

The other types of inequalities are handled similarly. ∎

We showed that vector $\boldsymbol{\ell}$ is a feasible solution. We now argue that it also approximates the optimum.

**Theorem 5.2.**

(i) *The vector $\boldsymbol{\ell}$ is a 2-approximate solution of the bounded integer program.*
(ii) *The value of the objective function at the vector $\boldsymbol{m}^*$ is at least a half of the value of the objective function of the best integer solution.*

*Proof:* By construction, $\boldsymbol{\ell} \leq 2\boldsymbol{m}^*$. From Section 2 we know that the vector $\boldsymbol{m}^*$ provides a lower bound on the value of the objective function for any integral solution. Hence, the theorem follows. ∎

The complexity of the algorithm is dominated by the complexity of the procedure in [6] for optimizing over a monotone system. The running time is $O(mnU^2 \log(Un^2/m))$.

What happens when we are optimizing over a 2-SAT integer program? The integer program that we are given may be of this type, or we may get a 2-SAT integer program by applying the transformation outlined in Section 4.

In this case, for all $i$ such that $m_i^+ = -m_i^-$, we get $\ell_i = m_i^*$. If $m_i^+ \neq -m_i^-$, we get $\ell_i = z_i$. In other words, the 2-approximate solution is obtained by rounding, either up or down, the fractional coordinates of $\boldsymbol{m}^*$. Lemma 5.1 ensures that there always exists a feasible rounding.

We note that in the special case of a *monotone* 2-SAT integer programming problem, all the basic solutions of the linear programming relaxation are integer, hence no rounding is required in this case.

## 6. Properties of binary integer programs

In this section we further investigate the properties of 2-SAT integer programs. We first consider the linear relaxation of a 2-SAT integer programming problem. It turns out that solutions of this relaxation always have denominator not greater than 2. This follows from the statement in the next lemma about the determinants of 2-SAT's *nonseparable* submatrices. A matrix is *nonseparable* if there do not exist partitions of the columns and rows to two subsets (or more) $C_1, C_2$ and $R_1, R_2$ such that all nonzero entries in every row and column appear only in the submatrices defined by the sets $C_1 \times R_1$ and $C_2 \times R_2$.

**Lemma 6.1.** *The determinants of all nonseparable submatrices of a* 2-SAT *linear programming problem have absolute value at most* 2*.*

*Proof:* Let $\boldsymbol{A}$ denote the constraint matrix of a 2-SAT integer program. Thus, $\boldsymbol{A}$ has at most two non-zero entries in every column. We show that the absolute value of the determinant of any nonseparable square submatrix of $\boldsymbol{A}$ can be either 0, 1, or 2. The proof of this claim is by induction on the size of the submatrix. Since the entries of $\boldsymbol{A}$ are from $\{-1, 0, 1\}$, the claim holds for $1 \times 1$ submatrices. Assume it holds for any $(m-1) \times (m-1)$ submatrix and we show that the claim holds for any $m \times m$ submatrix.

We may assume that each row and column in $\boldsymbol{A}$ has exactly two non-zero entries. Otherwise, there must be a row or a column where all entries, possibly with the exception of one, are zero. In either case, we can apply the inductive assumption directly and prove the claim. Let $\boldsymbol{A}_{ij}$ denote the submatrix obtained by deleting the $i$'th row and the $j$'th column from $\boldsymbol{A}$.

Without loss of generality, we may assume that the two non-zero elements in row $i$ of $\boldsymbol{A}$ are in columns $i$ and $i+1$ (modulo $m$). (Due to the nonseparability of the submatrix, this can be achieved by appropriate row and column interchanges.) Hence,

$$\det(\boldsymbol{A}) = A[1,1] \cdot \det(\boldsymbol{A}_{11}) - (-1)^m A[m,1] \cdot \det(\boldsymbol{A}_{m1}) \ .$$

The absolute values of the determinants of $\boldsymbol{A}_{11}$ and $\boldsymbol{A}_{m1}$ are equal to 1, since both are triangular matrices with nonzero diagonal elements. Therefore, the absolute value of the determinant of $\boldsymbol{A}$ is at most 2. ∎

An immediate corollary of Lemma 6.1 is the fact that the value of every variable in a basic solution of the 2-SAT linear program is in the set $\{0, \frac{1}{2}, 1\}$. Although for binary integer problems the subdeterminants can be of value greater than 2, and hence the solutions would not be in this set, we get rid of these "unnecessary" solutions by reducing the problem first to 2-SAT, as in Section 4. In a 2-SAT system the variables are assumed to be binary. Lemma 6.1, however, applies to any linear programming problem with a constraint matrix with coefficients $0, 1, -1$, and at most two nonzero elements in each row. We call such a system *generalized* 2-SAT. Note that we do not assume the existence of finite upper bounds on the variables. We will show that a 2-approximation can be achieved even for such systems.

**Lemma 6.2.** *A generalized* 2-SAT *has the property that* $S_I^{(2)} = S^{(\frac{1}{2})}$.

*Proof:* It suffices to prove that $S^{(\frac{1}{2})}$ is contained in $S_I^{(2)}$. Let $\boldsymbol{x} \in S^{(\frac{1}{2})}$. Define a solution $(\boldsymbol{x}^+, \boldsymbol{x}^-)$ as follows. For $j = 1, \ldots, n$,

(i) If $x_j$ is an integer, set $x_j^+ = -x_j^- = x_j$.
(ii) If $x_j$ is a noninteger, then set $x_j^+ = x_j + \frac{1}{2}$ and $x_j^- = -x_j + \frac{1}{2}$.

It is easy to show that $(\boldsymbol{x}^+, \boldsymbol{x}^-)$ satisfies the (three) generic types of constraints defining $S_I^{(2)}$. For example, consider a constraint of the form $x_j^+ - x_k^- \geq c$. Since $\boldsymbol{x}$ is feasible, we have $x_j + x_k \geq c$. If either both $x_j$ and $x_k$ are integer or both are noninteger, then we have $x_j^+ - x_k^- = x_j + x_k \geq c$. Assuming that $x_j + x_k$ is noninteger, if $x_j + x_k \geq c$ then $x_j + x_k - \frac{1}{2} \geq c$. Using the fact that $x_j^+ \geq x_j$ and $-x_k^- \geq x_k - \frac{1}{2}$, it follows that $x_j^+ - x_k^- \geq x_j + x_k - \frac{1}{2} \geq c$. The other cases follow from similar considerations. ∎

One corollary of Lemmas 6.1 and 6.2 is that the linear programming relaxation of a 2-SAT and a generalized 2-SAT can be solved by optimizing over the respective monotone system. Both problems are then solvable in strongly polynomial time: the 2-SAT as a maximum flow (or rather minimum cut) problem, and the generalized 2-SAT as a dual of a linear flow problem. Note that one could also solve these linear programs in strongly polynomial time, without using the transformation to a monotone system, by directly applying the algorithm of [14]. The latter, however, is not as efficient as the best known algorithms for solving maximum flow problems or linear flow problems.

We next show how to obtain a 2-approximation for a generalized 2-SAT integer program. First, we note that the procedure described above in Section 5 is not applicable here since the variables might not have finite upper bounds. Since we already know how to solve the monotone system, the difficulty lies in finding a feasible integer solution or verifying that none exists. We perform this latter task as follows.

Let $(\boldsymbol{x}^+, \boldsymbol{x}^-)$ be an optimal solution of the monotone system, *i.e.*, $\boldsymbol{x} = \frac{1}{2}(\boldsymbol{x}^+ - \boldsymbol{x}^-)$ solves the linear programming relaxtion. Using, if necessary, the transformation in the proof of Lemma 6.2, we may assume that $x_j^+ = -x_j^-$ or $x_j^+ = -x_j^- + 1$ $(j = 1, \ldots, n)$. Next, we apply Lemma 5.1 to conclude that the given generalized 2-SAT integer program is feasible if and only if there exists a feasible rounding of $\boldsymbol{x}$. The latter can be tested by the linear time algorithm in [3]. Moreover, Theorem 5.2 ensures that if such a rounding exists, then it is a 2-approximation.

## 7. "Fixing" Variables

As discussed in the introduction, the solution to the relaxation of the vertex cover problem (VCR) has the property that there exists an optimal solution that coincides with the relaxed solution in all integer components. This allows to "fix" the variables that are integer in the relaxation and remove them from further consideration, hence reduce the size of the problem. Not only is the size of the problem reduced, but also *any* solution on the remaining set of variables has an objective function value at most twice the optimum. This idea was proposed in [5] as a method for generating approximation algorithms with a worst-case ratio smaller than 2 for various special classes of graphs. We show here that precisely the same idea applies to any integer programming problem (IP), after it is transformed to a 2-SAT.

**Lemma 7.1.** *Let $\boldsymbol{x}^{(2)}$ be an optimal solution of* 2-SAT *in the set $S_I^{(2)}$. Let*

$$\mathrm{INT} = \{j \mid x_j^{(2)} = 0 \quad or \quad x_j^{(2)} = 1\} \ .$$

*Then there is an optimal integer solution $\boldsymbol{z}$ of* 2-SAT*such that $z_j = x_j^{(2)}$ for $j \in$* INT.

*Proof:* The proof is a generalization of that in [12]. For a set $A \subseteq \{1, 2, \ldots, n\}$, let $w(A) = \sum_{j \in A} w_j$. Let $P_1 = \{j | x_j^{(2)} = 1\}$, and $P_0 = \{j | x_j^{(2)} = 0\}$, *i.e.*, $P_1 \cup P_0 = \mathrm{INT}$. In the first part we prove the claim that there exists an optimal integer solution $\boldsymbol{z}$ such that $P_1$ is a suset of $\{j | z_j = 1\}$. In the second part of the proof we use such an integer solution to construct an optimal integer solution satisfying the statement of the Lemma.

Consider first the proof of the claim. The proof is by contradiction. Suppose that $I = \{j | z_j = 0\} \cap P_1$ is nonempty for every optimal integer solution $\boldsymbol{z}$. The contradiction will follow by proving that there exists a solution in $S^{(\frac{1}{2})}$ that is strictly better than $\boldsymbol{x}^{(2)}$ which, with Lemma 5.1, contradicts the optimality of $\boldsymbol{x}^{(2)}$.

Consider some optimal solution $\boldsymbol{z}$. Let $J = \{j | z_j = 1\} \cap P_0$. We now prove that

$$w(I) > w(J).$$

Indeed, if the opposite inequality were to hold for some $\boldsymbol{z}$, then a solution, say $\boldsymbol{u}$, obtained from $\boldsymbol{z}$ by setting the variables with indices in $I$ to 1 and the variables with indices in $J$ to 0, would be feasible and its objective value would not be greater than the optimum. But $\{j|u_j = 0\} \cap P_1$ is empty, contradicting the assumption. The feasibility of $\boldsymbol{u}$ follows from the feasibility of $\boldsymbol{z}$ and $\boldsymbol{x}^{(2)}$. To prove the feasibility, consider first a constraint of the type $x_p + x_q \geq 1$. We may assume, without loss of generality, that $p \in J$. Under this assumption $x_p^{(2)} = 0$ and therefore $x_q^{(2)} = 1$. If $z_q = 1$ then $u_q = z_q$ since $q \notin J$ and the constraint is satisfied. If $z_q = 0$ then $q \in I$, $u_q = 1$, and again the constraint is satisfied. Second consider a constraint of the type $x_p + x_q \leq 1$. We may assume, without loss of generality, that $p \in I$. Thus $x_p^{(2)} = 1$ and therefore $x_q^{(2)} = 0$ and $q \notin I$. If $q \in J$, then $u_q = 0$ and $\boldsymbol{u}$ satisfies the constraint. Otherwise, we conclude that $u_q = z_q = 0$, and again $\boldsymbol{u}$ satisfies the constraint. Finally, consider the last type of constraint, $x_p \geq x_q$. If $p \in I$, this constraint is certainly satisfied by $\boldsymbol{u}$. Suppose that $p \notin I$. If $z_p = 0$, then $x_p^{(2)} = 0$. From the feasibility of both $\boldsymbol{z}$ and $\boldsymbol{x}^{(2)}$ it follows that $z_q = 0$ and $x_q^{(2)} = 0$. Thus, $u_p = u_q = 0$ and $\boldsymbol{u}$ satisfies the constraint. Next, suppose that $z_p = 1$. If $x_p^{(2)} = 1$ then $u_p = z_p = 1$ and again the constraint is satisfied. If $x_p^{(2)} = 0$, then $p \in J$ and also $x_q^{(2)} = 0$. If $z_q = 0$, then $u_q = z_q = 0$ since $q \notin J$ , while if $z_q = 1$, then $q \in J$ and therfore $u_q = 0$. In both cases, $u_q = 0$ implies that $\boldsymbol{u}$ satisfies the constraint. This concludes the proof that $\boldsymbol{u}$ is a feasible integer solution. Therefore, $w(I) > w(J)$.

Consider now the vector $\boldsymbol{x}'$ where

$$x'_j = \begin{cases} 1/2 & \text{if} \quad j \in I \\ 1/2 & \text{if} \quad j \in J \\ x_j^{(2)} & \text{otherwise.} \end{cases}$$

We claim that $\boldsymbol{x}'$ is feasible, and hence $\boldsymbol{x}' \in S^{(\frac{1}{2})}$. To prove feasibility, we consider the three types of possible inequalities, and show that they are satisfied. Since all inequalities involving two variables that are both equal to $\frac{1}{2}$ are satisfied, we need only verify inequalities in which one of the variables is in $I \cup J$ and the other is in $P_1 \setminus I$ or in $P_0 \setminus J$.

Consider the inequality $x_p + x_q \leq 1$. This inequality may not be satisfied for $\boldsymbol{x}'$ if $x'_p = \frac{1}{2}$ and $x'_q = 1$. Since $x'_q = 1$ we have $q \in P_1 \setminus I$. Now, $p \notin I$, or else $\boldsymbol{x}^{(2)}$ is infeasible because in that case both values of $x_p$ and $x_q$ in $\boldsymbol{x}^{(2)}$ are 1. If $p \in J$, then $z_p = 1$ and also $z_q = 1$ (as $q \in P_1 \setminus I$), hence $\boldsymbol{z}$ is infeasible. Therefore, whenever this inequality arises it is feasible.

Consider now the inequality $x_p + x_q \geq 1$. This inequality may not be satisfied for $\boldsymbol{x}'$ if $x'_p = \frac{1}{2}$ and $x'_q = 0$. So $q \in P_0 \setminus J$. $p$ must therefore be in $I$, since otherwise $p \in J$ , $x_p^{(2)} = x_q^{(2)} = 0$ and $\boldsymbol{x}^{(2)}$ is infeasible. $p$ in $I$ implies that $z_p = 0$, which in turn implies that $z_q = 1$. Thus, we conclude that $q \in J$ which contradicts the fact that $q \in P_0 \setminus J$. So, again, this inequality is feasible for $\boldsymbol{x}'$.

Finally, consider the inequality $x_p \geq x_q$ . This inequality may not be satisfied for $\boldsymbol{x}'$ if $x'_p = \frac{1}{2}$ and $x'_q = 1$ or if $x'_p = 0$ and $x'_q = \frac{1}{2}$. Consider the first case: $q \in P_1 \setminus I$

and $p \in I$, or else $\boldsymbol{x}^{(2)}$ is infeasible. But then $z_p = 0$ and $z_q = 1$, which implies that $\boldsymbol{z}$ is infeasible. Hence, this case cannot occur. In the second case $q \in J$ or else $q \in I$, $x_q^{(2)} = 1$ and $\boldsymbol{x}^{(2)}$ is infeasible. Thus, $z_q = 1$. On the other hand, from $p \in P_0 \setminus J$ we have $z_p = 0$ and the feasibility of $\boldsymbol{z}$ is contradicted.

It follows that $\boldsymbol{x}'$ is a feasible solution and $\boldsymbol{x}' \in S^{(\frac{1}{2})}$. Now,

$$\sum_{j=1}^{n} w_j x_j^{(2)} - \sum_{j=1}^{n} w_j x'_j = w(I) - \tfrac{1}{2}w(I) + \tfrac{1}{2}w(J) = \tfrac{1}{2}w(I) - \tfrac{1}{2}w(J) > 0 \ .$$

This contradicts the optimality of $\boldsymbol{x}^{(2)}$ in $S_I^{(2)} = S^{(\frac{1}{2})}$. Hence, there is an optimal solution $\boldsymbol{z}$ such that $\{j \mid z_j = 1\} \supset P_1$.

If $P_0$ is a subset of $\{j|z_j = 0\}$ then the proof is complete. Thus, suppose that $I_0$, defined as the intersection of $P_0$ and $\{j|z_j = 1\}$, is nonempty. Define an integer solution $\boldsymbol{u}$ by setting $u_j = 0$ if $j \in I_0$ and $u_j = z_j$ otherwise. It is clear that $P_1$ is a subset of $\{j|u_j = 1\}$ and $P_0$ is a subset of $\{j|u_j = 0\}$. Also, the objective value at $\boldsymbol{u}$ is not larger than the objective value at $\boldsymbol{z}$. To complete the proof, it suffices to show that $\boldsymbol{u}$ is a feasible solution. Consider first a constraint of the type $x_p + x_q \geq 1$. Without loss of generality, suppose that $p \in I_0$. Thus, $x_p^{(2)} = 0$. Since $\boldsymbol{x}^{(2)}$ is a feasible solution, it follows that $x_q^{(2)} = 1$. The latter implies that $z_q = 1$. In particular, $q \notin I_0$ and therefore $u_q = 1$, so the constraint is satisfied. It is clear that $\boldsymbol{u}$ satisfies every constraint of the type $x_p + x_q \leq 1$. Finally, consider a constraint of the type $x_p \geq x_q$. If either $q \in I_0$ or $p \notin I_0$ the constraint is trivially satisfied. Thus, suppose that $q \notin I_0$ and $p \in I_0$. Hence, $x_p^{(2)} = 0$. Moreover, from the feasibility of $\boldsymbol{x}^{(2)}$ it follows that $x_q^{(2)} = 0$. If $z_q$ were equal to 1 then we would get that $q \in I_0$. Therefore $z_q = 0$, which in turn implies that $u_q = 0$, and the constraint is satisfied by the integer vector $\boldsymbol{u}$. This concludes the proof. ∎

**Notes.** This paper is based on an earlier version by the first author alone. The contribution of N. Megiddo and A. Tamir was to indicate that the qualification of "rounding property" in that earlier version was unnecessary, and a feasible solution can always be found provided that the original problem is feasible. They also proposed an alternative algorithm, where the problem (IP) is first transformed to 2-SAT and then the monotone transformation and the [6] procedure is applied. This algorithm is in fact identical in its outcome, the graph created, to the algorithm presented here, and hence not discussed explicitly. The extension to the generalized 2-SAT is also due to them, as well as various improvements to the presentation of the paper.

**Acknowledgement.** Dorit Hochbaum is grateful to two referees for their suggestions that clarified the presentation significantly. She thanks one of the referees for identifying a flaw in a previous version of the proof of Lemma 7.1. She also thanks I. Adler for his helpful comments.

## References


[1] E. Cohen and N. Megiddo, "Improved Algorithms for Linear Inequalities with Two Variables per Inequality," in: *Proceedings of the Twenty Third Symposium on Theory of Computing,* New Orleans (1991) 145–155.

[2] H. Edelsbrunner, G. Rote, and E. Welzl, "Testing the Necklace Condition for Shortest Tours and Optimal Factors in the Plane," *Theoretical Computer Science* **66** (1989) 157–180.

[3] S. Even, A. Itai, and A. Shamir, "On the Complexity of Timetable and Multicommodity Flow Problems," *SIAM Journal on Computing* **5** (1976) 691–703.

[4] D. Gusfield and L. Pitt, "A bounded approximation for the minimum cost 2-SAT problem," *Algorithmica* **8** (1992) 103–117.

[5] D. S. Hochbaum, "Efficient bounds for the stable set, vertex cover and set packing problems," *Discrete Applied Mathematics* **6** (1983) 243–254.

[6] D. S. Hochbaum and J. Naor, "Simple and fast algorithms for linear and integer programs with two variables per inequality," UC Berkeley manuscript, June 1991. (See also *Proceedings of the second Integer Programming and Combinatorial Optimization Conference* (1992) pp. 41–60).

[7] R. Kannan, "A polynomial algorithm for the two-variable integer programming problem," *J. ACM* **27**(1980) 118–122.

[8] R. M. Karp, "Reducibility among combinatorial problems," in R. E. Miller and J. W. Thatcher, Eds., *Complexity of Computer Computations,* Plenum Press, N.Y. (1972) pp. 85–103.

[9] J. C. Lagarias, "The computational complexity of simultaneous diophantine approximation problems," *SIAM J. Comput.* **14** (1985) 196–209.

[10] H. W. Lenstra, Jr., "Integer programming with a fixed number of variables," *Mathematics of Operations Research* **8** (1983) 538–548.

[11] N. Megiddo, "Towards a genuinely polynomial algorithm for linear programming," *SIAM J. Comput.* **12** (1983) 347–353.

[12] G. L. Nemhauser and L. E. Trotter, Jr., "Vertex packings: Structural properties and algorithms," *Mathematical Programming* **8** (1975) 232–248.

[13] J. C. Picard, "Maximal closure of a graph and applications to combinatorial problems," *Management Science* **22** (1976) 1268–1272.

[14] É. Tardos, "A strongly polynomial algorithm to solve combinatorial linear programs," *Operations Research* **34** (1986) 250–256.