Integer Pivoting Revisited

Robert T. Firla[a,⋆] **Utz-Uwe Haus**[a,♯] **Matthias Köppe**[a,♯]
Bianca Spille[b] **Robert Weismantel**[a]

[a] *Institute of Mathematical Optimization, Otto-von-Guericke-University Magdeburg, Universitätsplatz 2, 39106 Magdeburg, Germany*
[b] *Departement of Mathematics, EPFL Lausanne, CH-1015 Lausanne, Switzerland*

Abstract. This paper deals with algorithmic issues related to the design of an augmentation algorithm for general and 0/1-integer programs. We recall the approach of integer pivoting and introduce the family of Gomory-Young augmentation vectors that can be derived from a simplex tableau. Furthermore, a technique of combining Gomory-Young vectors and combinatorial augmentation vectors in one augmentation scheme is presented. Two computational experiments demonstrate the potential of pivoting in an integer fashion.

Key words. integer programming – primal algorithm – integer simplex – integral basis

1 Primal versus Dual Integer Programming

This paper deals with algorithmic approaches to solve general *integer linear programs* (ILP) of the form $\max \mathbf{c}^\mathsf{T}\mathbf{x} : \mathbf{x} \in \mathcal{F} := \{\mathbf{x} \in \mathbb{Z}^n : \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$, with integer valued matrix $\mathbf{A}$ and vectors $\mathbf{b}$ and $\mathbf{c}$.

One common approach to solve such an optimization problem (OPT) is the *dual approach*, an LP-based technique integrating a separation strategy. From the point of view of computational complexity this approach relies on the polynomial time equivalence of the optimization problem (OPT) and the separation problem (SEP), see Grötschel, Lovász & Schrijver [13].

A dual algorithm works in principle via the following scheme. Let $\mathbf{y}^* \in \mathbb{R}^n$ be an optimal solution of the linear relaxation $\max \mathbf{c}^\mathsf{T}\mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}$. If $\mathbf{y}^*$ is integral, it is the integer optimum of (ILP) and we are done. Otherwise, a linear inequality—the cutting plane—must be determined that is valid for all $\mathbf{x} \in \mathcal{F}$, but separates the current LP-optimum $\mathbf{y}^*$ and strengthens the LP-relaxation.

Modern cutting plane algorithms make use of 'general' cutting planes for separation purposes. Such cuts are generated algebraically, i.e., they ignore the combinatorial structure of the problem. Valid inequalities implied by integrality constraints were first introduced in a study of the traveling salesman problem by Dantzig, Fulkerson & Johnson [2, 3].

⋆Supported by a "Gerhard-Hess-Forschungsförderpreis" (WE 1462/2-2) of the German Science Foundation (DFG) awarded to R. Weismantel.

♯Supported by grants FKZ 0037KD0099 and FKZ 2495A/0028G of the Kultusministerium of Sachsen-Anhalt.

The theory of general cutting planes is due to the pioneering work of Gomory in the late 1950's. Gomory [8, 9, 10] developed a systematic way of generating valid inequalities directly from the given system and gave a first finite algorithm with an appropriate use of the dual simplex method to solve general integer programming problems. The roundoff and integrality test problems of these algorithms are resolved with the (dual) all-integer algorithm of Gomory [11], where all calculations are done in integers; a variant of this method was designed by Glover [6].

A disadvantage of a dual algorithm is that an intermediate stopping does not yield a feasible integral solution. An integer linear program can, however, also be solved with a *primal approach*: starting with a feasible solution, one proceeds through a sequence of improvement steps until the IP-optimum is reached. Hence, at any time in the course of a primal algorithm a feasible solution is available. From an algorithmic point of view the primal approach relies on a study of the *augmentation problem* (AUG): Given vectors $\mathbf{c} \in \mathbb{Z}^n$ and $\mathbf{x} \in \mathcal{F}$, return an *augmentation vector* $\mathbf{t} \in \mathbb{Z}^n$ such that $\mathbf{c}^\mathsf{T}\mathbf{t} > 0$ and $\mathbf{x} + \mathbf{t} \in \mathcal{F}$, or assert that no such vector $\mathbf{t}$ exists.

Grötschel & Lovász [12] and Schulz, Weismantel & Ziegler [18] proved that for 0/1-problems (OPT) and (AUG) are polynomial time equivalent. In the general case, one needs access to a directed augmentation oracle in order to solve (OPT) with a polynomial number of augmentations, see Schulz & Weismantel [17].

There are two different ways to design an augmentation algorithm for an integer programming problem when no a-priori knowledge about the constraint set is available. One basic fact that makes both approaches computationally interesting is that they resort to the power of linear programming duality.

Starting with an integral feasible solution $\mathbf{x}^0$, one wants to detect an augmenting direction that is applicable at $\mathbf{x}^0$, or provide a proof that $\mathbf{x}^0$ is optimal. To achieve this, a system of linear inequalities is to be derived such that $\mathbf{x}^0$ becomes a basic feasible solution of this system. The next step is to make use of the simplex method. If the reduced cost of all the non-basic variables are non-positive, the current tableau provides a certificate for the optimality of $\mathbf{x}^0$. Suppose that this is not the case. Then there exist non-basic variables in the current tableau with positive reduced cost. The usual simplex method would then perform a pivot operation on a column with positive reduced cost. This is of course not feasible in an integer setting, because in general the execution of a simplex pivot results in a new basic feasible solution that is no longer integral.

There are two different ways to overcome this difficulty. One approach is based on the use of cutting planes in a way that the cut generates a feasible pivot element attaining the value one in the cut inequality and, secondly, that the cut itself will become a legitimate pivot row. If the right hand side of this cut is different from zero, pivoting on this cut results in an *integer pivot step*. This approach is based on the primal simplex algorithm and was first proposed by Ben-Israel & Charnes [1]. Simplified variants were given by Young [19, 20] and Glover [7].

The *integral basis method* of Haus, Köppe & Weismantel [14, 16] is a second approach to perform this task. It refrains from adding cuts, but replaces the cutting step by a step in which the columns of the given system are manipulated. Roughly speaking, their algorithm iteratively substitutes one column by columns that correspond to irreducible solutions of certain linear diophantine inequalities, maintaining the integral basis property of the tableau as in invariant. We refer to [14, 16] for more details on this algorithm.

The intention of this paper is to recall this classical and nearly forgotten approach of integer pivoting. We demonstrate, that the idea of associating a basic solution of a simplex tableau with a given integer point and performing integer pivots as long as possible is still interesting and deserves further investigations. We also point out some analogies between augmentation and separation strategies.

2 Basic Integer Solutions and Integer Pivoting

To provide a notational context for describing the procedure, we shall consider integer linear programs of the form:

$$\max \mathbf{c}^\top \mathbf{x} : \mathbf{x} \in \mathcal{F} := \{\mathbf{x} \in \mathbb{Z}^{n+m} : \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}\} \tag{1}$$

with $\mathbf{A} \in \mathbb{Z}^{m\times(n+m)}$, $\mathbf{b} \in \mathbb{Z}^m_+$, and $\mathbf{c} \in \mathbb{Z}^{n+m}$. Suppose that a feasible basic solution $\mathbf{x} = (\mathbf{x}_B, \mathbf{x}_N)$ is available; denote by x_{B_i} for $i = 1, \ldots, m$ the basic variables and by x_j for $j = 1, \ldots, n$ the non-basic variables. Then system (1) can be formulated as

$$\begin{array}{rcll} \max\ c_0 + \sum_{j\in N} \bar{c}_j x_j & & & \\ x_{B_i} + \sum_{j\in N} \bar{a}_{ij} x_j & = & \bar{b}_i & \text{for } 1 \leq i \leq m, \\ x_{B_i},\ x_j & \in & \mathbb{Z}_+ & \text{for } 1 \leq i \leq m,\ 1 \leq j \leq n, \end{array}$$

where $\bar{\mathbf{A}} = \mathbf{A}_B^{-1}\mathbf{A}_N$, $\bar{\mathbf{b}} = \mathbf{A}_B^{-1}\mathbf{b}$, $\bar{\mathbf{c}}^\top = \mathbf{c}_N^\top - \mathbf{c}_B^\top\mathbf{A}_B^{-1}\mathbf{A}_N$, and $c_0 = \mathbf{c}_B^\top\mathbf{A}_B^{-1}\mathbf{b}$.

We assume that all coefficients $\bar{a}_{ij}$, $\bar{b}_i$, and $\bar{c}_j$ are integral, so that we can write the problem as a primal feasible, all-integer simplex tableau:

$$\begin{array}{c|cc} & & \mathbf{x}_N \\ & -c_0 & \bar{\mathbf{c}} \\ \hline x_{B_i} & \bar{b}_i & \bar{\mathbf{A}}_i. \end{array} \tag{2}$$

If the current basis is dual feasible, i.e., $\bar{\mathbf{c}} \leq 0$, the integral solution $\mathbf{x} = (\bar{\mathbf{b}}, \mathbf{0})$ is optimal. If it is not, then there exists at least one column s with $\bar{c}_s > 0$. Choose this column as the pivot column for a primal pivot, i.e., the variable x_s becomes basic. Choose a pivot row r such that $\bar{b}_r/\bar{a}_{rs} = \min_{1\leq i\leq m}\{\bar{b}_i/\bar{a}_{is} \mid \bar{a}_{is} \geq 1\}$.

If $\bar{a}_{rs} = 1$, we can execute a pivot operation on the r-th row, i.e., x_{B_r} leaves the basis and x_s enters the basis, and we maintain integrality. If $\bar{a}_{rs} \neq 1$, then an ordinary pivot step on the chosen pivot element would yield a fractional solution.

Therefore, before carrying out the pivot, we derive a cut from the source row r,

$$x_s + \sum_{j \in N \setminus \{s\}} \left\lfloor \frac{\bar{a}_{rj}}{\bar{a}_{rs}} \right\rfloor x_j \le \left\lfloor \frac{\bar{b}_r}{\bar{a}_{rs}} \right\rfloor . \tag{3}$$

This inequality is a Chvátal-Gomory cut that is valid for all integral solutions $\mathbf{x} \in \mathcal{F}$. Adding a new slack variable s_1 and adjoining this cut as the bottom row to the initial simplex tableau (2) yields a tableau

$$\begin{array}{c|cc} & & \mathbf{x}_N \\ & -c_0 & \bar{\mathbf{c}} \\ \hline x_{B_i} & \bar{b}_i & \bar{\mathbf{A}}_{i\cdot} \\ s_1 & \lfloor \frac{\bar{b}_r}{\bar{a}_{rs}} \rfloor & \lfloor \frac{\bar{\mathbf{A}}_{r\cdot}}{\bar{a}_{rs}} \rfloor \end{array}$$

Obviously, the vector $(\mathbf{x}, \lfloor \frac{\bar{b}_r}{\bar{a}_{rs}} \rfloor)$ is the integral basic solution associated with the extended system. Since

$$\left\lfloor \frac{\bar{b}_r}{\bar{a}_{rs}} \right\rfloor \Big/ \left\lfloor \frac{\bar{a}_{rs}}{\bar{a}_{rs}} \right\rfloor = \left\lfloor \frac{\bar{b}_r}{\bar{a}_{rs}} \right\rfloor \le \frac{\bar{b}_r}{\bar{a}_{rs}},$$

the cut row (3) is a legitimate pivot row. As the coefficient of x_s in this row is 1, one can execute a primal pivot operation on column s that maintains integrality of the resulting tableau.

Let $(\hat{\mathbf{x}}, 0)$ denote the new basic integer solution after applying this pivot operation. The difference vector $\hat{\mathbf{x}} - \mathbf{x}$ of the feasible solutions, if different from $\mathbf{0}$, is called a *Gomory-Young augmentation vector*. Geometrically, a Gomory-Young augmentation vector is the difference vector of adjacent extreme points of the convex hull defined by the feasible integral solutions of the given problem.

Unfortunately, the above algorithm does not automatically support a proof of finiteness, because the right hand side of the cut row may be zero. In this case, the values of all variables remain unchanged—in particular, the objective value does not change—and we do not move away from the old basic feasible solution but represent it by a new basis only. This problem is related to degeneracy that can occur in linear programming problems. The problem of degenerate cuts can be overcome by using specific pivot column and source row selection rules. The first finite convergent algorithm based exclusively on the cuts (3) was given by Young [19]. It uses, however, quite complicated rules for the selection of pivot columns and rows. Simplified versions including finiteness proofs were given by Glover [7] and Young [20], see also Garfinkel & Nemhauser [5].

Example 1. To exemplify the described procedure consider the integer program

$$\begin{array}{rrcrcrl} & \max & x_1 & & & & \\ \text{subject to} & -3x_1 & + & 5x_2 & \le & 5, & \text{(a)} \\ & 6x_1 & - & 4x_2 & \le & 7, & \text{(b)} \\ & 10x_1 & - & 4x_2 & \le & 15, & \text{(c)} \\ & & & x_1, x_2 & \in & \mathbb{Z}_+ & \end{array}$$

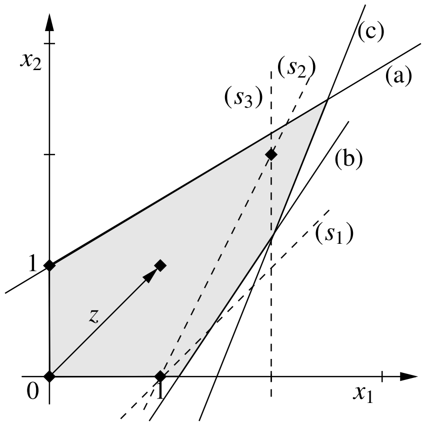

Fig. 1: *The feasible region of the integer program and the cuts generated.*

We extend the system by adding the slack variables $\{x_3, x_4, x_5\}$ and start with the primal feasible integer solution $\mathbf{x}^0 = (0, 0, 5, 7, 15)$. Thus, $B = \{3, 4, 5\}$ and $N = \{1, 2\}$. The primal feasible, all-integer tableau corresponding to this solution is T_0 (ignore the s_1-row for the moment being). We perform the steps of the algorithm and choose the x_1-column as the (unique) pivot column and the x_4-row as the source row. The resulting pivot element is marked by an asterisk (∗). From the source row we generate the Chvátal-Gomory cut as defined in (3), adjoin it as the bottom row of the tableau T_0, and add the new slack variable s_1 to the current basis. The generated cut is (s_1) $x_1 - x_2 \leq 1$, see Fig. 1.

T_0:

		x_1	x_2
	0	1	0
x_3	5	−3	5
x_4	7	6^*	−4
x_5	15	10	−4
s_1	1	$\boxed{1}$	−1

The adjoined row is chosen as the pivot row, and performing the primal pivot operation—the pivot element is marked by a box—leads to the new tableau T_1 and the improved solution $\mathbf{x}^1 = (1, 0, 8, 1, 5)$.

T_1:

		s_1	x_2
	−1	−1	1
x_1	1	1	−1
x_3	8	3	2
x_4	1	−6	2
x_5	5	−10	6^*
s_2	0	−2	$\boxed{1}$

T_2:

		s_1	s_2
	−1	1	−1
x_1	1	−1	1
x_2	0	−2	1
x_3	8	7^*	−2
x_4	1	−2	−2
x_5	5	2	−6
s_3	1	$\boxed{1}$	−1

T_3:

		s_3	s_2
	−2	−1	0
x_1	2	1	0
x_2	2	2	−1
x_3	1	−7	5
x_4	3	2	−4
x_5	3	−2	−4

This process is repeated according to the given selection rules and produces the tableaus T_2 and T_3. The cuts generated in these steps are (s_2) $2x_1 - x_2 \leq 2$ and (s_3) $x_1 \leq 2$, see Fig. 1. The final tableau T_3 is dual feasible, since there are no positive reduced cost coefficients. Hence, the corresponding basic solution $x^* = (2, 2, 1, 3, 3)$ is optimal to the initial problem.

3 Integrating Combinatorial Augmentation Vectors

Branch and cut algorithms of today make use of a combination of general and combinatorial cutting planes. For this reason, it seems interesting to study interactions between general and combinatorial augmentation vectors.

To be more formal, let us assume that we have a feasible basic solution $\mathbf{x}$. Suppose furthermore, that there is a family $\mathcal{T}$ of (combinatorial) augmentation vectors and a special vector $\mathbf{t} \in \mathcal{T}$ such that $\mathbf{x} + \mathbf{t} \in \mathcal{F}$ and $\mathbf{c}^\mathsf{T}\mathbf{t} > 0$. Our objective is to augment $\mathbf{x}$ by $\mathbf{t}$ (or a suitable positive integer multiple of $\mathbf{t}$) in such a way, that after performing this operation, in the subsequent step, we are still left with the possibility of generating a Gomory-Young augmentation vector or using another vector from the family $\mathcal{T}$ for further improvement. Note, however, that in order to produce a Gomory-Young vector a basic feasible solution is needed. In general, we can neither expect that $\mathbf{x} + \lambda\mathbf{t}$ is a basic feasible solution of the original problem for any $\lambda \in \mathbb{Z}_+$, nor that there is an obvious primal feasible, all-integer tableau corresponding to the new vector. Fortunately, this problem can be overcome by lifting the problem into higher dimensional space as we now show.

Again we study an integer program as given in (1) and suppose that a feasible basic solution $\mathbf{x} = (\mathbf{x}_B, \mathbf{x}_N)$ and a corresponding primal feasible, all-integer simplex tableau are available. Then we can rewrite system (1) as follows

$$\max c_0 + \bar{\mathbf{c}}^\mathsf{T}\mathbf{x}_N : \mathbf{x} \in \mathcal{F} = \{\mathbf{x} \in \mathbb{Z}^{n+m} : (\mathbf{I}, \bar{\mathbf{A}})(\mathbf{x}_B, \mathbf{x}_N) = \bar{\mathbf{b}}, \mathbf{x} \geq \mathbf{0}\}. \tag{4}$$

with $\bar{\mathbf{A}} \in \mathbb{Z}^{m\times n}$, $\bar{\mathbf{b}} \in \mathbb{Z}^m$, $\bar{\mathbf{c}} \in \mathbb{Z}^n$ and $c_0 \in \mathbb{Z}$.

Suppose, $\mathbf{t}$ is a feasible (combinatorial) augmentation vector with which we want to improve $\mathbf{x}$, i.e., $\mathbf{x} + \mathbf{t} \in \mathcal{F}$ and $\mathbf{c}^\mathsf{T}\mathbf{t} > 0$. Decomposing $\mathbf{t} = (\mathbf{t}_B, \mathbf{t}_N)$, then (4) implies

$$\mathbf{t}_B + \bar{\mathbf{A}}\,\mathbf{t}_N = \mathbf{0}, \mathbf{x}_B + \mathbf{t}_B \geq \mathbf{0}, \text{ and } \mathbf{t}_N \geq \mathbf{0}. \tag{5}$$

We use $\mathbf{t}$ to define a new reduced cost coefficient $\bar{\gamma} := \bar{\mathbf{c}}^\mathsf{T}\mathbf{t}_N = \mathbf{c}^\mathsf{T}\mathbf{t} > 0$ and an integral column vector $\bar{\mathbf{a}} := \bar{\mathbf{A}}\mathbf{t}_N = -\mathbf{t}_B$. Then system (4) is enlarged by adjoining an additional column,

$$\mathbf{c}_\mathbf{t} := (\mathbf{c}, \mathbf{c}_N^\mathsf{T}\mathbf{t}_N),\ \bar{\mathbf{c}}_\mathbf{t} := (\bar{\mathbf{c}}, \bar{\gamma}) \text{ and } \mathcal{F}_\mathbf{t} := \{\mathbf{y} \in \mathbb{Z}^{n+m+1} : (\mathbf{I}, \bar{\mathbf{A}}, \bar{\mathbf{a}})\,\mathbf{y} = \bar{\mathbf{b}},\ \mathbf{y} \geq \mathbf{0}\}.$$

Lemma 2 summarizes the correspondence between the points in $\mathcal{F}$ and $\mathcal{F}_\mathbf{t}$.

Lemma 2. *Let $\mathcal{F}$ and $\mathcal{F}_\mathbf{t}$ as above. Then $\mathbf{x} \in \mathcal{F}$ implies $\mathbf{y} := (\mathbf{x}, 0) \in \mathcal{F}_\mathbf{t}$. Conversely, $\mathbf{y} = (\mathbf{y}_B, \mathbf{y}_N, \hat{y}) \in \mathcal{F}_\mathbf{t}$ implies $\mathbf{x} := (\mathbf{y}_B, \mathbf{y}_N + \hat{y}\,\mathbf{t}_N) \in \mathcal{F}$. Moreover, $\mathbf{c}^\mathsf{T}\mathbf{x} = \mathbf{c}_\mathbf{t}^\mathsf{T}\mathbf{y}$.*

Proof. The first implication is easy to see. For the second we have

$$\begin{aligned} \mathbf{y} \in \mathcal{F}_\mathbf{t} \quad &\Longleftrightarrow \quad (\mathbf{I}, \bar{\mathbf{A}}, \bar{\mathbf{a}})\,\mathbf{y} = \bar{\mathbf{b}}, \mathbf{y} \geq \mathbf{0} \;\Longleftrightarrow\; (\mathbf{I}, \bar{\mathbf{A}}, \bar{\mathbf{A}}\mathbf{t}_N)\,\mathbf{y} = \bar{\mathbf{b}}, \mathbf{y} \geq \mathbf{0} \\ &\Longleftrightarrow \quad \mathbf{y}_B + \bar{\mathbf{A}}(\mathbf{y}_N + \hat{y}\,\mathbf{t}_N) = \bar{\mathbf{b}}, (\mathbf{y}_B, \mathbf{y}_N, \hat{y}) \geq \mathbf{0}. \end{aligned}$$

Setting $\mathbf{x} := (\mathbf{y}_B, \mathbf{y}_N + \hat{y}\,\mathbf{t}_N)$, we obtain $(\mathbf{I}, \bar{\mathbf{A}})\mathbf{x} = \bar{\mathbf{b}}$. Since $\mathbf{t}_N \geq \mathbf{0}$ (see (5)) we also have $\mathbf{x} \geq \mathbf{0}$ and therefore $\mathbf{x} \in \mathcal{F}$. Furthermore,

$$\mathbf{c}_\mathbf{t}^\mathsf{T}\mathbf{y} = c_0 + \bar{\mathbf{c}}_\mathbf{t}^\mathsf{T}(\mathbf{y}_N, \hat{y}) = c_0 + (\bar{\mathbf{c}}, \bar{\gamma})^\mathsf{T}(\mathbf{y}_N, \hat{y}) = c_0 + \bar{\mathbf{c}}^\mathsf{T}(\mathbf{y}_N + \hat{y}\,\mathbf{t}_N) = \mathbf{c}^\mathsf{T}\mathbf{x}.$$

□

According to Lemma 2, we may consider instead of the integer program (4) the extended integer program $\max\{\mathbf{c}_{\mathbf{t}}^{\mathsf{T}}\mathbf{y} : \mathbf{y} \in \mathcal{F}_{\mathbf{t}}\}$. Since $\mathbf{c}_{\mathbf{t}}^{\mathsf{T}}\mathbf{y} = c_0 + \bar{\mathbf{c}}_N^{\mathsf{T}}\mathbf{y}_N + \bar{\gamma}\hat{y}$, the all-integer simplex tableau extended to the higher dimensional space reads

$$
\begin{array}{c|ccc}
 & & \mathbf{y}_N & \hat{y} \\
 & -c_0 & \bar{\mathbf{c}} & \bar{\gamma} \\
\hline
\mathbf{y}_B & \bar{\mathbf{b}} & \bar{\mathbf{A}} & \bar{\mathbf{a}}
\end{array}
$$

After this extension we perform the same operations as in the previous section. First we choose as pivot column the new $\hat{y}$-column with reduced cost $\bar{\gamma} > 0$; next we select a source row r according to $\bar{b}_r/\bar{a}_r = \min_{1\le i\le m}\{\bar{b}_i/\bar{a}_i \mid \bar{a}_i \ge 1\}$. From this row we generate the Chvátal-Gomory cut as defined in (3)

$$\hat{y} \ + \ \sum_{j\in N}\Big\lfloor\frac{\bar{a}_{rj}}{\bar{a}_r}\Big\rfloor y_j \ \le \ \Big\lfloor\frac{\bar{b}_r}{\bar{a}_r}\Big\rfloor.$$

Adjoining this cut as an equation with a new slack variable s_1 as the bottom row of the current tableau, the resulting new simplex tableau reads

$$
\begin{array}{c|ccc}
 & & \mathbf{y}_N & \hat{y} \\
 & -c_0 & \bar{\mathbf{c}} & \bar{\gamma} \\
\hline
\mathbf{y}_B & \bar{\mathbf{b}} & \bar{\mathbf{A}} & \bar{\mathbf{a}} \\
s_1 & \lfloor\frac{\bar{b}_r}{\bar{a}_r}\rfloor & \lfloor\frac{\bar{\mathbf{A}}_r}{\bar{a}_r}\rfloor & 1
\end{array}
$$

The new s_1-row can now be chosen as the pivot row whose pivot element attains a value of 1. Executing the primal pivot operation yields a new feasible basic solution $\mathbf{y}' \in \mathcal{F}_{\mathbf{t}}$. Using the update formulas for simplex tableaus this new solution is calculated in the following way (ignore the s_1-component):

$$\mathbf{y}' \ = \ (\mathbf{y}_B', \mathbf{y}_N', \hat{y}') := \Big(\bar{\mathbf{b}} - \Big\lfloor\frac{\bar{b}_r}{\bar{a}_r}\Big\rfloor\bar{\mathbf{a}},\, \mathbf{0},\, \Big\lfloor\frac{\bar{b}_r}{\bar{a}_r}\Big\rfloor\Big).$$

On account of Lemma 2, this new basic solution $\mathbf{y}' \in \mathcal{F}_{\mathbf{t}}$ corresponds to the vector $\mathbf{x}' \in \mathcal{F}$ with

$$\mathbf{x}' := \Big(\bar{\mathbf{b}} - \Big\lfloor\frac{\bar{b}_r}{\bar{a}_r}\Big\rfloor\bar{\mathbf{a}},\, \mathbf{0} + \Big\lfloor\frac{\bar{b}_r}{\bar{a}_r}\Big\rfloor\mathbf{t}_N\Big) = (\bar{\mathbf{b}}, \mathbf{0}) + \Big\lfloor\frac{\bar{b}_r}{\bar{a}_r}\Big\rfloor(\mathbf{t}_B, \mathbf{t}_N) = \mathbf{x} + \Big\lfloor\frac{\bar{b}_r}{\bar{a}_r}\Big\rfloor\mathbf{t}.$$

From (5) follows that $\bar{\mathbf{a}} = -\mathbf{t}_B \le \mathbf{x}_B = \bar{\mathbf{b}}$ and hence, $\lfloor\frac{\bar{b}_r}{\bar{a}_r}\rfloor \ge 1$, which means that we augment the initial solution $\mathbf{x}$ by an integer multiple of $\mathbf{t}$ and move away from the old to a new basic solution. Moreover, $\lfloor\frac{\bar{b}_r}{\bar{a}_r}\rfloor = \max\{\lambda \in \mathbb{Z}_+ : \mathbf{x} + \lambda\mathbf{t} \in \mathcal{F}\}$, implying that $\mathbf{x}$ is augmented by $\mathbf{t}$ as often as to remain feasible.

After executing this operation we have the possibility of continuing the algorithm by applying a further combinatorial augmentation vector, if available, or generating a Gomory-Young augmentation vector.

Example 3. (Example 1 continued)

We consider the same integer program as in Example 1. Again we start with the equation representation of this problem and the same initial feasible basic solution $\mathbf{x} = (\mathbf{x}_B, \mathbf{x}_N) = (0, 0, 5, 7, 15)$ with $B = \{3, 4, 5\}$ and $N = \{1, 2\}$. The simplex tableau corresponding to this basic solution is T_0.

Suppose that the feasible augmentation vector $\mathbf{t} = (1, 1, -2, -2, -6)$ has been obtained; this vector corresponds to the vector $\mathbf{z} = (1, 1)$ illustrated in Fig. 1. We want to use $\mathbf{t}$ to augment the current solution $\mathbf{x}$. We calculate the reduced cost coefficient $\bar{\gamma} := \bar{\mathbf{c}}^{\mathsf{T}}\mathbf{t}_N = 1$ and the column vector $\bar{\mathbf{a}} := \bar{\mathbf{A}}\mathbf{t}_N = -\mathbf{t}_B = (2, 2, 6)$. Extending the current simplex tableau T_0 by the column vector $\bar{\mathbf{a}}$ yields a simplex tableau T'_1 (ignore the s_1-row in the meantime):

T'_1:

		y_1	y_2	$\hat{y}$
	0	1	0	1
y_3	5	−3	5	2^*
y_4	7	6	−4	2
y_5	15	10	−4	6
s_1	2	−2	2	$\boxed{1}$

T'_2:

		y_1	y_2	s_1
	−2	3	−2	−1
y_3	1	1	1	−2
y_4	3	2	−8	−2
y_5	3	22	−16	−6
$\hat{y}$	2	−2	2	1

Next we select the new $\hat{y}$-column corresponding to $\bar{\mathbf{a}}$ as the pivot column and the y_3-row as the source row. The cut generated from this row is adjoined as the bottom row of the tableau and the new slack variable s_1 enters the basis. The new row becomes the pivot row and executing the primal pivot operation yields the new tableau T'_2. The feasible basic solution associated with this tableau is given by the vector $\mathbf{y}' = (\mathbf{y}'_B, \mathbf{y}'_N, \hat{y}') = (2, 2, 1, 3, 3, 2) \in \mathcal{F}_{\mathbf{t}}$. By Lemma 2, the corresponding feasible solution in the set $\mathcal{F}$ is $\mathbf{x}' := (0, 0, 1, 3, 3) + 2 \cdot (1, 1, 0, 0, 0) = (2, 2, 1, 3, 3)$ which is equal to $\mathbf{x} + 2\mathbf{t}$.

4 Two Computational Experiments

This section is meant to give some evidence that the approach of integer pivoting and associating a basic feasible solution of a simplex tableau with a specific integral solution is promising computationally.

In our first series of experiments we start with an optimal solution for some of the pure 0/1-MIPLIB instances that we get from running *CPLEX*[1]. We turn it into a basic feasible solution and perform a bound analysis on the non-basic variables. Note that for pure 0/1-problem instances such a basic feasible solution may be easily constructed, since we can always assume that the constraints $x_i \geq 0$ and $x_i \leq 1$ are given explicitly. The bound analysis consists in solving a sequence of linear programming problems. More precisely, for each of the non-basic variables in the tableau we set up an LP in which we maximize the variable subject to the given set of constraints plus the additional requirement that the LP-solution must yield

[1]CPLEX Linear Optimizer 6.6.0 with Mixed Integer & Barrier Solvers. © ILOG 1997–1999.

an improvement in terms of the original objective function. If the optimal value of such a specific LP is smaller than 1, we are guaranteed that the corresponding non-basic variable will never attain a value of 1 in any integer improving direction. As a consequence, this particular variable can be set to 0. Though being expensive, it usually pays to perform this preprocessing step. In general, it is extremely effective. Table 1 shows our results for some pure 0/1-MIPLIB instances. For all these instances, optimality of an integer solution can be proved by making this integer solution basic and performing a bound analysis on the non-basic variables of the resulting tableau. All computation times are given in CPU seconds on a Sun Enterprise 450 with 450 MHz.

Table 1: *Results for 0/1-MIPLIB instances*

	Columns	Rows	Time [s]
`enigma`	121	21	0.1
`nw08`	434	24	1.0
`cracpb1`	572	144	3.1
`air03`	10757	124	8770.4

As a second computational experiment we have been testing, whether certain pure 0/1-instances can be solved to optimality by a sequence of non-degenerate integer pivots. We selected a couple of max-clique problems that are collected in the DIMACS Challenge library[2]. Table 2 lists these instances. We model these instances as maximum stable set problems on the respective complement graph in a node-edge formulation. For all these examples we start with the trivial solution, i.e., all variables are set to 0. We associate with this solution a basis and create the corresponding tableau with data

$$\begin{array}{c|cc} & & \mathbf{x}_N \\ & -c_0 & \bar{\mathbf{c}} \\ \hline \mathbf{x}_B & \bar{\mathbf{b}} & \bar{\mathbf{A}} \end{array}$$

Next we check the reduced cost of all non-basic variables. If all these reduced cost are non-positive, optimality of our present solution has been verified. Otherwise, we perform for each of the non-basic variables the bound analysis that we described above. Thereafter we select among the columns one with positive reduced cost, s say, that attains a maximal pivot ratio, i.e., we select a pivot column s and a pivot row r such that $\bar{b}_r/\bar{a}_{rs} = \min_{1\le i\le m}\{\bar{b}_i/\bar{a}_{is} \mid \bar{a}_{is} \ge 1\}$ is maximal over all columns with positive reduced cost. Note that this value $\bar{b}_r/\bar{a}_{rs}$ is bounded by 1, since we work with a pure 0/1-problem. If $\bar{a}_{rs} = 1$, we can execute a pivot operation on the r-th row, i.e., x_{B_r} leaves the basis, x_s enters the basis, and we maintain

[2]Second DIMACS Implementation Challenge, `http://dimacs.rutgers.edu/Challenges/`

Table 2: *DIMACS challenge cliques*

	Nodes	Edges	Clique Size	Time [s]
`hamming6-2.clq`*	64	192	32	0.6
`hamming6-4.clq`	64	1312	4	3.4
`hamming8-2.clq`*	256	1024	128	12.3
`johnson8-2-4.clq`*	28	168	4	0.4
`johnson8-4-4.clq`	70	560	14	10.2
`johnson16-2-4.clq`*	120	1680	8	6.0
`johnson32-2-4.clq`*	496	14880	16	191.2
`c-fat200-1.clq`*	200	18366	12	112.0
`c-fat200-2.clq`*	200	16665	24	272.2
`c-fat200-5.clq`*	200	11427	58	864.5
`c-fat500-1.clq`*	500	120291	14	2300.8
`c-fat500-2.clq`*	500	115611	26	5336.3
`c-fat500-5.clq`*	500	101559	64	29314.5

integrality. Thus, we move away from the old integral solution by a non-degenerate integer pivot to another feasible integral solution, and we repeat this process.

If $\bar{a}_{rs} < 1$, then the right hand side of the cut row, that was introduced in Section 2, will be equal to 0. Pivoting on this cut would result in a degenerate pivot which we refrain from performing in our experiments. We stop in this case.

It turns out that, for all examples shown in Table 2, we can reach the optimal solution by a sequence of non-degenerate pivots. Moreover, once we reach the integer optimal solution, for all instances marked with a star, we can prove optimality just by the power of LP-duality and the bound analysis described above.

It was indeed a surprise for us that quite a significant number of problems can be solved by this augmentation strategy: Determine a basis for an integer solution, perform a bound analysis on the non-basic variables, and pivot in an integer fashion as long as possible. Of course, many other examples remain in which just integer pivoting is not sufficient to reach optimality or to prove it. Many of such instances can be solved using the integral basis method, see Haus, Köppe & Weismantel [15].

Acknowledgments: We thank Giovanni Rinaldi for many discussions on the subject of this paper. Thanks also due to Laurence Wolsey for providing us the literature of Gomory's work. The authors gratefully acknowledge the useful hints of anonymous referees.

References

[1] A. B -I and A. C , *On Some Problems of Diophantine Programming*, Cahirs du Centre d'Etudes de Recherche Operationelle **4**, 215–280, 1962

[2] G. B. D , D. R. F , and S. M. J , *Solution of a Large-scale Traveling Salesman Problem*, Operations Research **2**, 393–410, 1954

[3] G. B. D , D. R. F , and S. M. J , *On a Linear-Programming, Combinatorial Approach to the Traveling Salesman Problem*, Operations Research **7**, 58–66, 1959

[4] R. T. F , B. S , and R. W , *A Primal Analogue of Cutting Plane Algorithms*, Preprint 99-22, Otto-von-Guericke-Universität Magdeburg, Department of Mathematics, September 1999

[5] R. S. G and G. L. N , *Integer Programming*, Wiley, New York, 1972

[6] F. G , *A Pseudo Primal-Dual Integer Programming Algorithm*, Journal of Research of the National Bureau of Standards **13**, 879–919, 1967

[7] F. G , *A New Foundation for a Simplified Primal Integer Programming Algorithm*, Operations Research **16** (4), 727–740, 1968

[8] R. E. G , *Outline of an Algorithm for Integer Solutions to Linear Programs*, Bulletin of the American Mathematical Society **64**, 275–278, 1958

[9] R. E. G , *Solving Linear Programming Problems in Integers*, in: Combinatorial Analysis (R. E. Bellman and M. Hall, jr., eds.), Proceedings of Symposia in Applied Mathematics X, American Mathematical Society, Providence, Rhode Island, 211–216, 1960

[10] R. E. G , *An Algorithm for Integer Solutions to Linear Programming*, in: Recent Advances in Mathematical Programming (R. L. Graves and P. Wolfe, eds.), McGraw-Hill, New York, 269–302, 1963

[11] R. E. G , *An All-Integer Integer Programming Algorithm*, in: Industrial Scheduling (J. F. Muth and G. L. Thompson, eds.), Prentice-Hall, Englewood Cliffs, N.J., 193–206, 1963

[12] M. G ¨ and L. L ´ , *Combinatorial Optimization*, in: Handbook of Combinatorics, Volume II (R. L. Graham, M. Grötschel, and L. Lovász, eds.), North-Holland, 1541–1597, 1995

[13] M. G ¨ , L. L ´ , and A. S , *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*, Springer, Berlin, 1988

[14] U.-U. H , M. K¨ , and R. W , *The Integral Basis Method for Integer Programming*, Mathematical Methods of Operations Research **53** (3), 2001

[15] U.-U. H , M. K¨ , and R. W , *The Integral Basis Method in an Augmentation Framework for Integer Programming*, Preprint 01-23, Otto-von-Guericke-Universität Magdeburg, Department of Mathematics, October 2001

[16] U.-U. H , M. K¨ , and R. W , *A Primal All-Integer Algorithm Based on Irreducible Solutions*, Preprint 01-05, Otto-von-Guericke-Universität Magdeburg, Department of Mathematics, April 2001

[17] A. S. S and R. W , *An Oracle-Polynomial Time Algorithm for Integer Programming*, in: Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms, Baltimore, USA, 967–968, 1999

[18] A. S. S , R. W , and G. M. Z , *0/1 Integer Programming: Optimization and Augmentation are Equivalent*, in: Algorithms - ESA '95 (P. Spirakis, ed.), Lecture Notes in Computer Science 979, Springer, Berlin, 473–483, 1995

[19] R. D. Y , *A Primal (All Integer), Integer Programming Algorithm*, Journal of Research of the National Bureau of Standards **69B**, 213–250, 1965

[20] R. D. Y , *A Simplified Primal (All Integer) Integer Programming Algorithm*, Operations Research **16**, 750–782, 1968