Linköping Studies in Science and Technology. Theses.
No. 1383

Methods and Applications in Integer Programming: All-Integer Column Generation and Nurse Scheduling

Elina Rönnberg

Linköpings universitet
INSTITUTE OF TECHNOLOGY

Department of Mathematics
Linköping University, SE-581 83 Linköping, Sweden

Linköping 2008

ISBN 978-91-7393-760-3 ISSN 0280-7971

Printed in Sweden by LiU-Tryck, 2008.

Abstract

INTEGER PROGRAMMING can be used to provide solutions to complex decision and planning problems occurring in a wide variety of situations. Applying integer programming to a real life problem basically involves a first phase where a mathematical model is constructed, and a second phase where the problem described by the model is solved. While the nature of the challenges involved in the respective two phases differ, the strong relationship between the properties of models, and which methods that are appropriate for their solution, links the two phases. This thesis constitutes of three papers, of which the third one considers the modeling phase, while the first and second one consider the solution phase.

Many applications of column generation yield master problems of set partitioning type, and the first and second papers present methodologies for solving such problems. The characteristics of the methodologies presented are that all successively found solutions are feasible and integral, where the retention of integrality is a major distinction from other column generation methods presented in the literature.

The third paper concerns nurse scheduling and describes the results of a pilot implementation of a scheduling tool at a Swedish nursing ward. This paper focuses on the practical aspects of modeling and the challenges of providing a solution to a complex real life problem.

Acknowledgements

THANKS to my supervisor Torbjörn Larsson for excellent guidance and support. I also thank my present and former colleagues at the Department of Mathematics, my family, and friends.

The research presented in this thesis has been carried out with financial support from the Swedish Research Council.

Linköping, October 14th, 2008

Elina Rönnberg

Papers

THE FOLLOWING papers are appended and will be referred to by their Roman numerals.

[I] E. Rönnberg and T. Larsson. Column Generation in the Integral Simplex Method. *European Journal of Operational Research*, 192:333–342, 2009.

[II] E. Rönnberg and T. Larsson. An All-Integer Column Generation Methodology for Set Partitioning Problems. Technical Report LiTH-MAT-R-2008-09, October 2008.

[III] E. Rönnberg and T. Larsson. Automating the Self-Scheduling Process of Nurses in Swedish Healthcare: A Pilot Study. Technical Report LiTH-MAT-R-2008-08, October 2008.

Contents

Appended papers

1
Introduction

INTEGER PROGRAMMING can be used to model and solve complex decision and planning problems occurring in a wide variety of situations, as for example in logistics, infrastructure planning, and personnel and activity scheduling. A general linear integer program can be written on the form

$$\begin{aligned} z^* = \quad & \min \quad cx \\ & \text{s.t.} \quad Ax \geq b, \\ & \qquad\quad x \geq 0 \text{ and integer,} \end{aligned}$$

where x is a column vector of n variables, c is a row vector of n elements, A is an n by m matrix, and b is a column vector of m elements. If only some of the variables are required to be integral, the problem is called a mixed integer program.

Applying integer programming to a real life application basically involves two phases; first one needs to create a model for the problem to be solved, and then one applies a solution method to find a good, or an optimal, solution to the problem described by the model. These two phases are of course not isolated from each other, but the nature of the challenges involved in each phase differ.

In the first phase all aspects that need to be considered when solving the problem must be stipulated and quantified in order to create a foundation for the construction of a mathematical model describing the problem. When faced with a complex real life problem the challenge is to construct a model of manageable size that mirrors the reality sufficiently well and yields a solution practical interest. Considering this trade-off, one needs to delimit which aspects to include in the model, which ones to discard, and which ones that are to be replaced by some kind of approximation. It is also of crucial importance to resolve how to provide the model with input data of enough good quality for the solution to be reliable.

Paper [III] of this thesis describes a pilot implementation of an optimisation tool for the scheduling of nurses at a Swedish nursing ward and it focuses on the practical aspects of integer programming that we experienced during this work. An introduction to nurse scheduling is given in the paper, and this topic will therefore not be covered in this introduction to the thesis.

The second phase of integer programming aims at solving the problem described by the model constructed in the first phase. Whenever standard optimisation software is not useful, the challenge is to develop new methods or to tailor all ready existing ones to solve the problem in an effective, and preferably also efficient, way. There is of course a relation between how the model is formulated and which solution methods that can, or should, be used, creating an inevitable dependency between the first and second phase.

Papers [I] and [II] of this thesis consider this second phase, in the specific context of column generation methods for set partitioning problems. Section 1 below gives a brief introduction to column generation, and Section 2 gives a description of the set partitioning problem from a point of view that can facilitate the understanding of the contents of the first two papers.

1 Column generation

Column generation is originally a linear programming method that can be applied to a problem of the following form, henceforth referred to as the *master problem.*

$$\begin{aligned} z^* = \quad \min \quad & \sum_{j \in \mathcal{N}} c_j x_j \\ \text{s.t.} \quad & \sum_{j \in \mathcal{N}} a_{ij} x_j \geq b_i, \quad i = 1, \ldots, m, \\ & x_j \geq 0, \quad j \in \mathcal{N}. \end{aligned}$$

Here x_j, $j \in \mathcal{N}$ are the variables, and (c_j, a_j), $j \in \mathcal{N}$ are the columns, with $a_j = (a_{1j}, \ldots, a_{mj})$.

Typical in a column generation context is that the set of columns is huge and that it is disadvantageous, and usually also practically impossible, to have explicit knowledge of all of them. Considered instead is a *restricted master problem* where the columns (c_j, a_j), $j \in N \subseteq \mathcal{N}$, are assumed to be at hand.

Commonly, the columns have an authentic meaning in the application under consideration. To give an example, in a generalised assignment problem where a set of jobs should be assigned to a set of machines, a column can correspond to a specific set of jobs to be carried out by a machine. The role of the master problem is then to combine these columns so that each job is assigned to a machine. (In this example, the variables are of course required to be integral.)

The basic principles of column generation are closely related to those of the simplex method, which are assumed to be known by the reader. In the simplex method, a variable can be chosen to enter the basis if the reduced cost is negative, and if all reduced costs are non-negative, an optimal solution has been found. When applying column generation, if all reduced costs among the variables at hand in the restricted master problem are non-negative, an optimal solution to the restricted master problem has been found, but this solution is not necessarily an optimal solution to the complete master problem.

The column generation subproblem is to find a column $j \in \mathcal{N} \setminus N$, that has the lowest reduced cost, that is, given an optimal dual solution u_i, $i = 1, \ldots, m$, from the restricted master problem, one finds

$$\bar{c}_p = \min_{j \in \mathcal{N}} \bar{c}_j = \min_{j \in \mathcal{N}} c_j - \sum_{i=1}^{m} u_i a_{ij}.$$

If an optimal column has $\bar{c}_p < 0$, it should be added to the restricted master problem, which in turn should be reoptimised in the presence of this new column. On the other hand, if the optimal column has $\bar{c}_p \geq 0$, then the known optimal solution to the restricted master problem is also an optimal solution to the master problem.

In order to solve a master problem where the variables are required to be integral, the column generation principle is embedded in a branch-and-bound scheme, which is well described in Barnhart et al. [2], Vanderbeck and Wolsey [10], and Ryan and Foster [6]. For excellent general introductions to column generation in integer programming, see Lübbecke and Desrosiers [4] and Wilhelm [11].

2 The set partitioning problem

In many applications of column generation, the master problem to be solved is a set partitioning problem. As pointed out in the previous section, column generation is a linear programming method, and to solve the integer problem, the method needs to complemented with appropriate integer programming techniques.

The polytope of feasible solutions to the linear programming relaxation of the set partitioning problem has a property that guarantees that there exists a path between every pair of integer points consisting only of such edges of the polytope that connects integer points by simplex pivots on one-entries. This property, called quasi-integrality, is utilised in the integral simplex method for set partitioning problems, where only pivots on one-entries are allowed to be made. An advantage of this method is that it is primal and all-integer. Because of degeneracy, a basis that enables a pivot between two integer points can however be difficult to find. Performing simplex pivots on one-entries only is not sufficient to guarantee the finding of an optimal solution, but by complementing this method with implicit enumeration, optimality can eventually be obtained.

In the work described in Papers [I] and [II] of this thesis, we have tailored the column generation principle to the integral simplex method for set partitioning problems. The purpose of this section is to describe and, by using examples, illustrate the quasi-integrality property and provide insights into how it can be used when solving the set partitioning problem.

Throughout this section the following notation will be used to describe the set partitioning problem.

$$\begin{array}{llll} [SPP] & z^* = & \min & \sum_{j \in N} c_j x_j \\ & & \text{s.t.} & \sum_{j \in N} a_{ij} x_j = e_i, \quad i \in M \\ & & & x_j \in \{0,1\}, \quad j \in N. \end{array}$$

Here, $N = \{1, \ldots, n\}$ and $M = \{1, \ldots, m\}$ are the sets of indices for the variables and constraints respectively. The costs, c_j, $j \in N$, are integers, a_{ij}, $i \in M$, $j \in N$, are zeros and ones, and $e_i = 1$, $i \in M$. We assume that $n \geq m$ and that the problem is feasible. Let $a_j = (a_{1j}, \ldots, a_{mj})^{\mathrm{T}}$, $j \in N$, and assume, without loss of generality, that the matrix $(a_1, \ldots, a_n)$ has no zero column and full rank.

Let SPP^{LP} be the linear programming relaxation of SPP. Because the feasible set of SPP^{LP} is contained within the unit hypercube, all feasible integer points will be extreme points of this set. For an extreme point to SPP^{LP}, denote a basis by $B \subseteq \{a_1, \ldots, a_n\}$. If an extreme point is degenerate, it is associated with more than one basis. For a given basis, let I and J denote the corresponding sets of basic and non-basic columns, respectively, and let $u^{\mathrm{T}} = c_B^{\mathrm{T}} B^{-1}$, where $c_B = (c_j)_{j \in I}$, be the complementary dual basic solution. In the simplex tableau for a given basis B, let $\bar{e} = B^{-1}e$, where $e = (e_1, \ldots, e_m)^{\mathrm{T}}$, and $\bar{a}_j = B^{-1}a_j$, $j \in N$, be the updated right-hand-side and constraint columns, respectively, and $\bar{c}_j = c_j - u^{\mathrm{T}}a_j$, $j \in J$, be the reduced costs.

Two bases are called *adjacent* if they differ in exactly one column, and two extreme points are called adjacent if there exists a basis belonging to one of the extreme points that is adjacent to a basis belonging to the other extreme point.

2.1 Quasi-integrality In this section we will study an interesting property of the feasible polytope of the linear programming relaxation of the set partitioning problem.

Definition 1 *Let X be a polytope and X_I its set of integer points. The polytope X is called* quasi-integral *if every edge of the convex hull of X_I is also an edge of X.*

A direct consequence of the quasi-integrality property is that any two integer points of the polytope can be joined by a path that consists only of such edges of the polytope that connect integer points.

Theorem 1 *The feasible polytope of SPP^{LP} is quasi-integral.*

The first to show that the polytope of feasible solutions to SPP^{LP} has this property was Trubin [9]. Given below is the, in our opinion more accessible, proof due to Yemelichev et al. [13], though with some minor rephrasings made. For the proof we need the following two lemmas. The first one is a sufficient condition for a polytope to be quasi-integral and it is taken from [13].

Lemma 1 *Let X_I be a subset of the extreme points of a polytope X. If, given any two integral vertices of X there is an integral face containing them, then X is quasi-integral.*

The next lemma gives a sufficient condition for a matrix to be totally unimodular, and it is taken from Wolsey [12, page 39].

Lemma 2 *A matrix A with elements a_{ij} is totally unimodular if*

(i) $a_{ij} \in \{0, 1, -1\}, \quad \forall i, j.$

(ii) Each column contains at most two nonzero coefficients.

(iii) There exists a partition (M_1, M_2) of the set of rows such that each column j containing two nonzero coefficients satisfies

$$\sum_{i \in M_1} a_{ij} - \sum_{i \in M_2} a_{ij} = 0.$$

Proof of Theorem 1: According to Lemma 1, and because all feasible integer points of SPP are extreme points of the feasible polytope of SPP^{LP}, it suffices to show that any two feasible integer points, x^1 and x^2, of SPP belong to an integral face of the polytope of SPP^{LP}.

Partition the index set N into the three disjoint subsets

$$N_0 = \{j : x_j^1 = x_j^2 = 0\},\ N_1 = \{j : x_j^1 = x_j^2 = 1\},\ \text{and } N_2 = \{j : x_j^1 \neq x_j^2\}.$$

The set of feasible solutions of SPP^{LP} that satisfy $x_j = 0$, $j \in N_0$, and $x_j = 1$, $j \in N_1$, constitutes a face of the polytope of feasible solutions to SPP^{LP}. To show that this face is integral it suffices to show that the matrix $A_2 = (a_j)_{j \in N_2}$ is totally unimodular. Since $x^0 = (x^1 + x^2)/2$ is a feasible solution to SPP^{LP} it holds that

$$2 = \sum_{j \in N} a_{ij}(x^1 + x^2) = \sum_{j \in N_2} a_{ij}.$$

Hence, every row of A_2 contains exactly two elements equal to one. The columns of A_2 can be partitioned into two disjoint sets (M_1, M_2), where the first contains the variables x_j such that $x_j^1 = 1$ and the second contains the variables x_j such that $x_j^2 = 1$. That is, the transpose of A_2 fulfils the conditions of Lemma 2 and is thereby totally unimodular. A matrix is totally unimodular if and only if its transpose is totally unimodular, see [12, page 39], so A_2 is totally unimodular and the face containing x^1 and x^2 is integral. ■

Examples of other problems having the quasi-integrality property are the simple plant location problem, see [13], the uncapacitated network design problem, see [3], and the one and two facility, one commodity, network design problem, see [7].

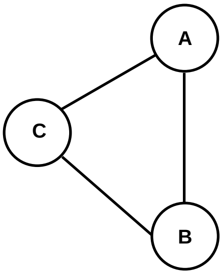
To illustrate the quasi-integrality property, the following example will be used during the remainder of this section.

$$[SPP_{EX}] \qquad z^* = \min \begin{pmatrix} c_1 & c_2 & c_3 & c_4 & c_5 \end{pmatrix} x$$

$$\text{s.t.} \begin{pmatrix} 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \end{pmatrix} x = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

$$x \in \{0,1\}^5,$$

Because the example is to be used to illustrate the quasi-integrality property, which is a property of the feasible polytope of SPP^{LP}, independently of the objective function, we have not specified any objective function in the example.



$$\begin{aligned} x^A = &\quad (1,0,0,0,0) \\ x^B = &\quad (0,1,0,0,1) \\ x^C = &\quad (0,1/2,1/2,1/2,0) \end{aligned}$$

Figure 2.1: *A graph where the vertices and edges are representing the extreme points of SPP_{EX}^{LP} and the polytope edges that connects them, respectively.*

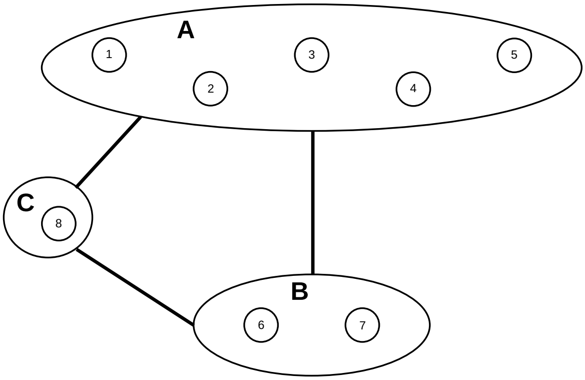


Figure 2.2: *A graph where the vertices within a super-vertex represent the possible bases for each extreme point. Each edge represents that there exists an edge of the polytope between the two extreme points.*

There are three extreme points associated with SPP_{EX}^{LP}, and these are $x^A = (1,0,0,0,0)$, $x^B = (0,1,0,0,1)$, and $x^C = (0,1/2,1/2,1/2,0)$. In Figure 2.1 the polytope describing the feasible set of SPP_{EX}^{LP} is illustrated by a graph, where the extreme points are represented by vertices, and the edges of the polytope are represented by edges in the graph.

Because of degeneracy, an extreme point can be associated with more than one basis. For x^A, there are five possible sets of basic variables, namely $\{x_1, x_2, x_3\}$, $\{x_1, x_2, x_4\}$, $\{x_1, x_3, x_4\}$, $\{x_1, x_3, x_5\}$, and $\{x_1, x_4, x_5\}$. For x^B there are two possible sets of basic variables, $\{x_2, x_3, x_5\}$ and $\{x_2, x_4, x_5\}$, and for x^C there is only one, $\{x_2, x_3, x_4\}$. In Figure 2.2, all these possible bases for each extreme point are illustrated as vertices within the super-vertex representing the extreme point. As in Figure 2.1, each edge of the graph represents that there exists an edge of the polytope between the two extreme points.

This far, quasi-integrality has been described as a geometric property of the polytope of feasible solutions of SPP^{LP}, but there is also a corresponding algebraic

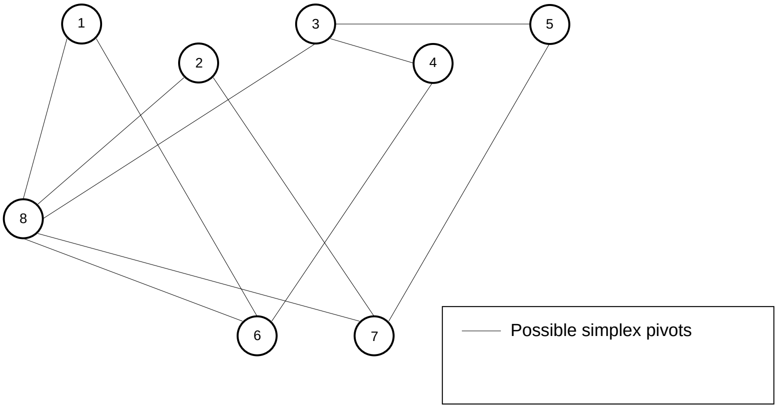


Figure 2.3: *The possible simplex pivots between the bases of SPP_{EX}^{LP}*

property that can be described within the context of the simplex method. The simplex method for linear programs is a search method in which pivots between adjacent bases of the polytope of feasible solutions are performed. When carrying out a simplex pivot, a variable x_s, $s \in J$, such that $\bar{c}_s < 0$, is chosen to enter the basis, and a variable x_r, $r \in I$, such that

$$\min_{i \in M} \left\{ \frac{\bar{e}_i}{\bar{a}_{is}} \;\middle|\; \bar{a}_{is} > 0 \right\} = \frac{\bar{e}_r}{\bar{a}_{rs}}$$

is chosen to leave the basis. In Figure 2.3, the feasible bases of SPP_{EX}^{LP} are represented by vertices the same way as in Figure 2.2, and all possible simplex pivots between them are represented by edges in the graph. Within the theory of degeneracy graphs, the graph in Figure 2.3 is called a representation graph. For further reading on degeneracy graphs, see Zörnig [14].

When solving the set partitioning problem, we are only interested in moving between bases representing integer extreme points, and derived below are sufficient conditions for integrality.

For $j \in I$, introduce the notation $j(i)$ such that $a_{j(i)} = (B)_i$, $i \in M$. The value of the basic variables, $j \in I$, can then be calculated by using Cramer's rule, that is,

$$x_{j(i)} = \frac{det(B^i)}{det(B)}, \qquad i \in M,$$

where B^i is the matrix obtained when replacing the ith-column vector of B by the column vector e. The matrix B^i is integral and so is $det(B^i)$, and therefore the following holds.

Observation 1 *If the basis B is such that $|det(B)| = 1$, then the corresponding extreme point is integral.*

In the simplex method, the pivot operation that transforms the current basis B into the new basis B_{new}, by letting the variable x_s enter the basis and the variable x_r leave the basis, can be represented by a pivot matrix, P, see Murty [5, page 141]. The pivot matrix P is a unit matrix of order m with column r replaced by the eta vector,

$$\eta = \left(\frac{-\bar{a}_{1s}}{\bar{a}_{rs}}, \ldots, \frac{-\bar{a}_{r-1,s}}{\bar{a}_{rs}}, \frac{1}{\bar{a}_{rs}}, \frac{-\bar{a}_{r+1,s}}{\bar{a}_{rs}}, \ldots, \frac{-\bar{a}_{ms}}{\bar{a}_{rs}} \right)^T .$$

If starting from a basis with $|det(B)| = 1$, integrality can be maintained by allowing only such pivots that lead to a basis with $|det(B_{new})| = 1$. The relation between the determinants of the current basis and the new one is that

$$det(B_{new}) = det\left((PB^{-1})^{-1}\right) = \frac{1}{det(P)det(B^{-1})}$$

where the determinant of P can be calculated by using expansion by minors on the rth row, yielding

$$det(P) = \frac{(-1)^{r+s}}{\bar{a}_{rs}}.$$

Then, if $|det(B)| = 1$, it holds that

$$|det(B_{new})| = \left|\frac{1}{det(P)}\right| \left|\frac{1}{det(B^{-1})}\right| = \left|\frac{1}{\bar{a}_{rs}}\right|,$$

which leads to the following observation.

Observation 2 *If a pivot is performed on an entry $\bar{a}_{rs} = 1$ in the simplex tableau at a given a basis B such that $|det(B)| = 1$, then the pivot yields a new basis B_{new} such that $|det(B_{new})| = 1$.*

In the integral simplex method to be described in Section 2.2 the simplex pivots are restricted to be carried out on one-entries only, and thereby integrality is maintained. Because of the importance of these pivots we make the following definition.

Definition 2 *Given a simplex tableau associated with an integer basic feasible solution to SPP^{LP} and an $s \in J$ such that $\bar{c}_s < 0$.*

(i) A non-degenerate pivot-on-one *is a pivot operation on an entry $\bar{a}_{rs} = 1$ such that*

$$\min_{i \in M} \left\{ \frac{\bar{e}_i}{\bar{a}_{is}} \;\middle|\; \bar{a}_{is} > 0 \right\} = \frac{\bar{e}_r}{\bar{a}_{rs}} = 1.$$

(ii) A degenerate pivot-on-one *is a pivot operation on an entry $\bar{a}_{rs} = 1$ such that*

$$\min_{i \in M} \left\{ \frac{\bar{e}_i}{\bar{a}_{is}} \;\middle|\; \bar{a}_{is} > 0 \right\} = \frac{\bar{e}_r}{\bar{a}_{rs}} = 0.$$

Both (i) and (ii) are referred to as a pivot-on-one*.*

Vertex 3:

		$-\bar{c}_2$			$-\bar{c}_5$	$\bar{z}$
x_1	1	2	0	0	-1	1
x_3	0	-1	1	0	1	0
x_4	0	-1	0	1	1	0

Vertex 4:

		$-\bar{c}_2$		$-\bar{c}_4$		$\bar{z}$
x_1	1	1	0	1	0	1
x_3	0	0	1	-1	0	0
x_5	0	-1	0	1	1	0

Figure 2.4: *The simplex tableaus corresponding to vertices 3 and 4 in the example.*

For a pivot to be between to distinct extreme points, it needs to be non-degenerate. If a non-degenerate pivot-on-one is carried out from an integer extreme point and a basis B such that $|det(B)| = 1$, then it is certain that the new basis, B_{new}, is also associated with an integer extreme point, and different from the one that B is associated with. If instead $|det(B)| \neq 1$, then the non-degenerate pivot-on-one can lead to a non-integer extreme point. Illustrated in the following example is that not all bases associated with an integer extreme point have $|det(B)| = 1$.

Example 1 *For a set partitioning problem with $m = 4$ constraints and $n \geq m$ variables, assume that the basis is given by*

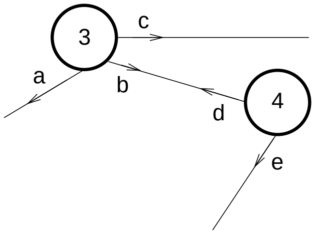
$$B = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}.$$

This basis has $det(B) = 2$, and is associated with the integer extreme point $x = (0, 0, 0, 1)^{\mathrm{T}}$.

Hence, by allowing pivots-on-one only, integrality is ensured to be maintained, but with the drawback that some bases associated with an integer extreme point are not possible to reach. The consequences of this drawback are further discussed in Section 2.2.

To illustrate how to determine which of the pivots in Example 1 that are pivots-one-one, the simplex tableaus for the vertices 3 and 4 of the example are given in Figure 2.4, and the associated movements in the graph are found in Figure 2.5. In example SPP_{EX}, all bases for the integer extreme points have $|det(B)| = 1$.

There are three possible pivots associated with leaving vertex 3. If x_2 enters and x_1 leaves the basis, the pivot will be non-degenerate, and not on a one-entry, hence the new solution will be the fractional extreme point x^C; if x_5 enters the basis and x_4 leaves the basis, the pivot is a degenerate pivot-on-one, and the solution will still be x^A; if x_5 enters the basis and x_3 leaves the basis, it is also a degenerate pivot-on-one. For vertex 4 there are two possible pivots. First, if x_2 enters and x_1 leaves



vertex	movement	entering variable	leaving variable
3	a	x_2	x_1
3	b	x_5	x_4
3	c	x_5	x_3
4	d	x_4	x_5
4	e	x_2	x_1

Figure 2.5: *The possible simplex pivots for leaving vertices 3 and 4.*

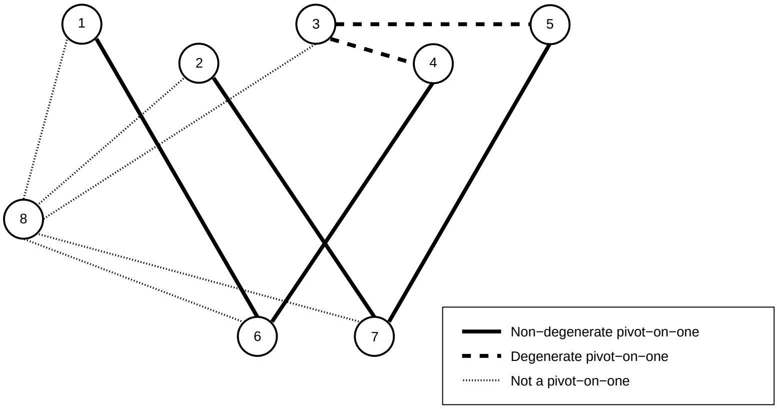


Figure 2.6: *A categorisation of the possible simplex pivots between the bases of* SPP^{LP}_{EX}.

the basis, it is a non-degenerate pivot-on-one leading to x^B. Second, if x_4 enters and x_5 leaves the basis, it is a degenerate pivot-on-one. In Figure 2.6, all the pivots between the bases of SPP^{LP}_{EX} are categorised in the same manner as for the pivots associated with vertex 3 and 4.

Now, assume that one would like to pivot from vertex 3 to vertex 1. It is then possible to do so by pivoting as shown by the arrows in Figure 2.7. The quasi-integrality property guarantees that this path exists, but not how to find it. At vertex 3, a choice is made between pivoting to vertex 4 or to vertex 5, and this choice is made without any knowledge of the consequences with respect to forthcoming pivots. Assuming instead that instead of choosing vertex 4 to follow on vertex 3, vertex 5 is chosen, then the path would be as shown by the arrows in Figure 2.8. As can be seen from the figure, it is not possible to continue with a pivot-on-one from vertex 2, except for reversing the pivot that led from vertex 7 to vertex 2. If restricted to pivots-on-one, the only way to get to vertex 1 from vertex 2 is by reversing the pivots back to vertex 3 and there choose to pivot to vertex 4 instead.

In this case it was possible to, after reversing the pivots, find another path of pivots-on-one to get from vertex 3 to vertex 1, but this is necessarily not always

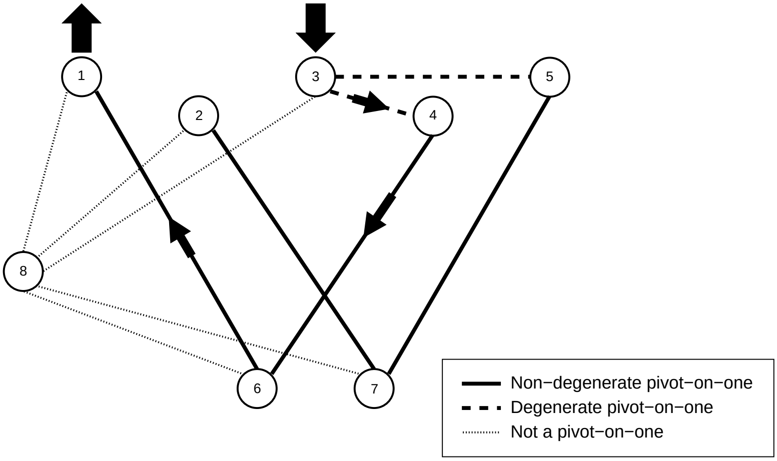


Figure 2.7: *A possible path of pivots-on-one from vertex 3 to vertex 1.*

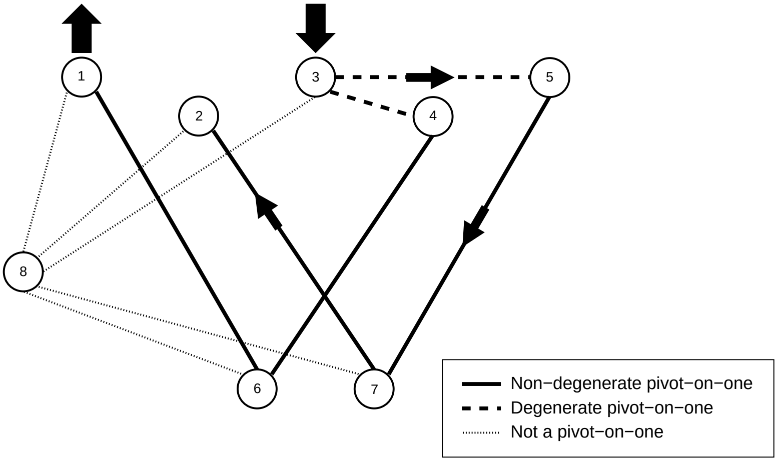


Figure 2.8: *A path of pivots-on-one from vertex 3, not leading to vertex 1.*

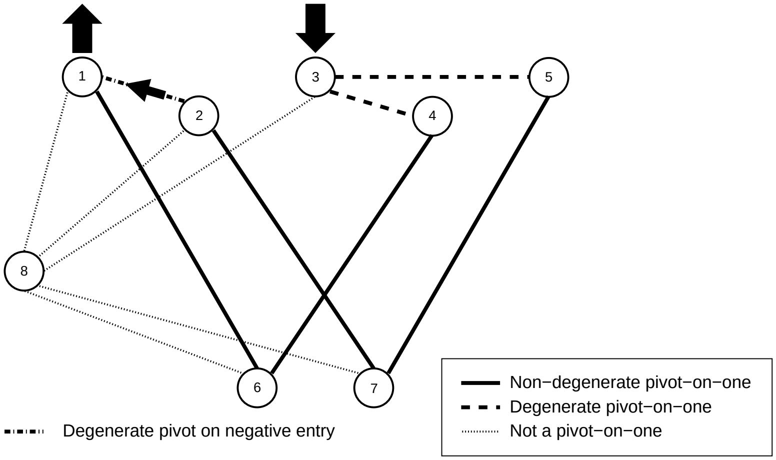

1
2
3
4
5
8
6
7
Degenerate pivot on negative entry
Non−degenerate pivot−on−one
Degenerate pivot−on−one
Not a pivot−on−one

Figure 2.9: *Vertex 1 can be reached from vertex 2 by a degenerate pivot on a negative entry.*

possible. The natural question that arises is if there are any alternatives to reversing pivots when no forward pivot-on-one is possible at a basis. One possibility would be to allow a degenerate pivots on all non-zero entries, and it is shown by Balas and Padberg [1] that if such pivots are allowed, the following result is obtained.

Theorem 2 *Let x^1 be a feasible and integer solution to SPP^{LP}, associated with the basis B_1, and suppose that x^1 is not optimal in SPP. Denote by x^2 an optimal solution to SPP, and let B_2 be an associated basis in SPP^{LP}. Let J_1 and Q_2 be the index sets of the non-basic columns in B_1 and the one-valued basic variables in x^2, respectively. Then there exists a sequence of adjacent bases $B_{10}, B_{11}, B_{12}, \ldots, B_{1p}$ in SPP^{LP}, such that $B_{10} = B_1$, $B_{1p} = B_2$, and (a) the associated vertices $x^1 = x^{10}, x^{11}, x^{12}, \ldots, x^{1p} = x^2$, are all feasible and integer, (b) $\sum_{j \in N} c_j x_j^{10} \geq \sum_{j \in N} c_j x_j^{11} \geq \ldots \geq \sum_{j \in N} c_j x_j^{1p}$, and (c) $p = |J_1 \cap Q_2|$.*

Since any vertex can be made optimal by adjusting the objective function, it follows from the theorem that for *any* two integer feasible basic solutions to SPP^{LP}, x^1 and x^2, there exists a sequence of adjacent bases of length $p = |J_1 \cap Q_2|$, such that the associated vertices are all feasible and integer. Observe that a consequence of Theorem 2 is that Hirsch conjecture, see [5, page 441], is true for integer solutions to SPP^{LP}.

In our example, performing a pivot on a negative entry at vertex 2, by letting x_3 become basic instead of x_4, corresponds to the movement illustrated in Figure 2.9, where vertex 1 is reached from vertex 2. If allowing degenerate pivots on all non-zero entries, the pivots will no longer be only between bases with $|det(B)| = 1$, and then a non-degenerate pivot-on-one can lead to a non-integer extreme point; hence, new complications are then introduced.

To summerise, the quasi-integrality property of the polytope SPP^{LP} implies that any two integer points of the polytope can be joined by a path that consists only of such edges of the polytope that connect integer points by a pivot-on-one. The difficulties encountered are due to the fact that at each extreme point, degeneracy makes it hard to find a basis that enables such a pivot.

2.2 The integral simplex method In the late sixties, Trubin [9] outlined the idea of a method, the integral simplex method, that utilises that the polytope of SPP^{LP} has the quasi-integrality property. This method is the simplex method restricted to performing pivots-on-one only. Because of the restriction to pivots-on-one, one might reach a basis associated with a non-optimal extreme point where no further pivot-on-one is possible, as illustrated for the example SPP_{EX} in Figure 2.8, Section 2.1. Because pivots-on-one are not sufficient to guarantee the finding of a global optimum, the integral simplex method can be regarded as a local search method that can get trapped at a local optimum, as defined below.

Definition 3 *A basis associated with an integer extreme point to SPP^{LP} is called a* local optimum *if it is not possible to make a pivot-on-one.*

A strategy for handling the existence of local optima that are not global is presented by Thompson [8]. In his method, the feasible set is partitioned through a branching, when the integral simplex method is trapped at a local optimum. When branching, a subproblem is created for each variable with negative reduced cost, and in the subproblem this variable is forced to take the value one. The fixation of a variable enables a reduction of the original problem, and each of the reduced problems is solved by the integral simplex method, with further branchings performed if necessary. Thompson has applied his method to randomly generated problem instances and problems arising in crew scheduling, and he reports the results to be promising.

Set partitioning problems are often highly degenerate, and a deficiency in the integral simplex method is the absence of a pivoting rule that prevents cycling, and to the best of our knowledge, no such pivoting rule has been suggested in the literature. If the integral simplex method is to be used for solving problems of practical interest, a rule for efficient prevention of cycling needs to be constructed.

3 Thesis contributions and future work

The column generation methodologies for set partitioning problems presented in Papers [I] and [II] are closely related in the sense that they both utilise the quasi-integrality property and thereby yield all-integer methods. The method presented in Paper [I] is an integration of the column generation principle and the integral simplex method, and to ensure the finding of a global optimum, a branching strategy is used. Presented in Paper [II] is a general framework for an all-integer column generation methodology for set partitioning problems. This methodology consists of strategies for finding improved solutions to the restricted master problem, as well as strategies for augmenting it with new columns.

The major contribution of these two papers is the, in a column generation context, novel way of handling the integrality requirements on the variables; the strategies we suggest are fundamentally different from what is described in the literature. What remains to be explored is how the components of the methodology should be realised and combined with each other, and perhaps also with other methods, and if this can yield computationally efficient methods.

Our intention is to apply the developed methodology to the generalised assignment problem, which would result in a knapsack type column generation problem, and to some application with a column generation problem of shortest path type. Our column generation methodology transfers complications due to the integrality requirements in the master problems to the column generation phase, either in terms of extra restrictions on the columns to be generated, or by necessitating that more column than an optimal one are produced by the column generation problem. How to handle these complications in the column generation phase must be taken under careful consideration to facilitate for a successful method.

Paper [III] is of an essentially different nature than the first two papers, and it describes an attempt to incorporate an optimisation tool in a kind of self-scheduling that is popular at Swedish nursing wards. The contribution of this paper is that we managed to take into account all crucial real life aspects of the scheduling, and thereby were able to create schedules that could be used by a pilot ward. The mathematical model used in this study was partially tailored for the pilot ward, and we are now working on a generalised model that should be valid for scheduling at most Swedish nursing wards. To solve the problem for the pilot ward, we used CPLEX, that managed to provide solutions of good quality within a couple of minutes. Even though CPLEX served it purpose well in the pilot study, it could be of interest to develop column generation or meta-heuristic methods for solving the studied nurse scheduling problem. Perhaps it could then even be possible to exploit ideas from Papers [I] and [II].

References

[1] E. Balas and M. W. Padberg. On the set-covering problem. *Operations Research*, 20:1152–1161, 1972.

[2] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. Branch-and-price: column generation for solving huge integer programs. *Operations Research*, 46:316–329, 1998.

[3] J. Hellstrand, T. Larsson, and A. Migdalas. A characterization of the uncapacitated network desing polytope. *Operations Research Letters*, 12:159–163, 1992.

[4] M. E. Lübbecke and J. Desrosiers. Selected topics in column generation. *Operations Research*, 53:1007–1023, 2005.

[5] K. G. Murty. *Linear programming.* John Wiley & Sons, New York, NY, 1983.

[6] D. M. Ryan and B. A Foster. An integer programming approach to scheduling. In A. Wren, editor, *Computer Scheduling of Public Transport: Urban Passenger Vehicle and Crew Scheduling*, pages 269–280, Amsterdam, 1981. North-Holland.

[7] T. Sastry. One and two facility network design revisited. *Annals of Operations Research*, 108:19–31, 2001.

[8] G. L. Thompson. An integral simplex algorithm for solving combinatorial optimization problems. *Computational Optimization and Applications*, 22:351–367, 2002.

[9] V. A. Trubin. On a method of solution of integer linear programming problems of a special kind. Translated by V. Hall. *Soviet Mathematics Doklady*, 10: 1544–1546, 1969.

[10] F. Vanderbeck and L. A. Wolsey. An exact algorithm for IP column generation. *Operations Research Letters*, 19:151–159, 1996.

[11] W. E. Wilhelm. A technical review of column generation in integer programming. *Optimization and Engineering*, 2:159–200, 2001.

[12] L. A. Wolsey. *Integer programming.* John Wiley & Sons, New York, NY, 1998.

[13] V. A. Yemelichev, M. M. Kovalev, and M. K. Kravtsov. *Polytopes, graphs and optimisation.* Translated by G. H. Lawden. Cambridge University Press, Cambridge, 1984.

[14] P. Zörnig. *Degeneracy graphs and simplex cycling.* Springer-Verlag, Berlin, 1991.