Technische Universität Berlin

FAKULTÄT II
MATHEMATIK UND
NATURWISSENSCHAFTEN
Institut für Mathematik

SELECTED TOPICS IN COLUMN GENERATION

by

MARCO E. LÜBBECKE JACQUES DESROSIERS

No. 2004/08

Selected Topics in Column Generation

Marco E. Lübbecke
Technische Universität Berlin
Institut für Mathematik, Sekr. MA 6-1
Straße des 17. Juni 136, D-10623 Berlin, Germany
m.luebbecke@math.tu-berlin.de

Jacques Desrosiers
HEC Montréal and GERAD
3000, chemin de la Côte-Sainte-Catherine
Montréal, Québec, Canada H3T 2A7
jacques.desrosiers@hec.ca

December 2, 2002
Revised March 9, 2004

Abstract

Dantzig-Wolfe decomposition and column generation, devised for linear programs, is a success story in large scale integer programming. We outline and relate the approaches, and survey mainly recent contributions, not yet found in textbooks. We emphasize the growing understanding of the dual point of view, which has brought considerable progress to the column generation theory and practice. It stimulated careful initializations, sophisticated solution techniques for the restricted master problem and subproblem, as well as better overall performance. Thus, the dual perspective is an ever recurring concept in our "selected topics."

OR/MS Subject Classification: Integer Programming: Column generation, Dantzig-Wolfe decomposition, Lagrangian relaxation, branch-and-bound; Linear programming: large scale systems

1 Introduction

Almost five decades have passed since Ford and Fulkerson (1958) suggested dealing only implicitly with the variables of a multicommodity flow problem. Dantzig and Wolfe (1960) pioneered this fundamental idea, developing a strategy to extend a linear program columnwise as needed in the solution process. This technique was first put to actual use by Gilmore and Gomory (1961, 1963) as part of an efficient heuristic algorithm for solving the cutting stock problem. Column generation is nowadays a prominent method to cope with a huge number of variables. The embedding of column generation techniques within a linear programming based branch-and-bound framework, introduced by Desrosiers, Soumis, and Desrochers (1984) for solving a vehicle routing problem under time window constraints, was the key step in the design of exact algorithms for a large class of integer programs.

This paper is a survey on column generation biased toward solving integer programs. Numerous integer programming column generation applications are described in the literature, as can be seen from Table 1. Generic algorithms for solving problems by integer programming column generation were presented by Barnhart et al. (1998b) and Vanderbeck and Wolsey (1996). Algorithmic efficacy is considered by Desaulniers, Desrosiers, and Solomon (2001b). Some dissertations (Ben Amor, 2002; Sol, 1994; Vanderbeck, 1994; Villeneuve, 1999) are a rich source of computational testing. Previous general reviews include those by Desrosiers et al. (1995); Soumis (1997); Wilhelm (2001).

We merge promising contemporary research works with more classical solution strategies in order to cover the whole integer programming column generation solution process. On the theoretical side, we give a primer on decomposition and column generation, pointing out that in general one cannot simply impose integrality constraints on the generated variables. On the algorithmic side, we emphasize the bounding role of the column generation algorithm and the importance of dual solutions for achieving a good overall performance. The paper is divided in two major parts. The first part covers the theory that is needed to expose integer programming column generation algorithms. In §2, we recall the classical decomposition principle in linear programming, which leads to the formulation of linear programs with a huge number of variables. In §3, we present both convexification and discretization approaches for extending the decomposition principle in order to handle integrality constraints. The second part is the algorithmic counterpart of the first. §§ 4, 5 and 6 cover the solution of linear programming column generation, expanding respectively on the strategies developed for getting dual solutions to the restricted master problems, generating new variables and compensating for bad convergence behaviors. Section 7 integrates integer programming considerations, putting the preceding algorithms in perspective. Our conclusion brings attention to the strongest and most promising ideas that are presented, in the hope that ever more complex column generation applications could be successfully solved.

2 Column Generation and Dantzig-Wolfe Decomposition

In applications, constraint matrices of (integer) linear programs are typically sparse and well-structured. Subsystems of variables and constraints appear in independent groups, linked by a distinct set of constraints and/or variables. Multicommodity flow formulations for vehicle routing and crew scheduling problems are well known examples (Desaulniers et al., 1998; Desrosiers et al., 1995).

The general idea behind the decomposition paradigm is to treat the linking structure at a superior, coordinating, level and to independently address the subsystem(s) at a subordinated level, exploiting their special structure algorithmically. We are concerned with linking constraints, or price directive

Reference(s)	Application(s)
Agarwal et al. (1989); Desaulniers et al. (2001b); Desrochers et al. (1992); Löbel (1997, 1998); Ribeiro and Soumis (1994)	various vehicle routing problems
Borndörfer et al. (2003); Desaulniers et al. (2001b); Desrochers and Soumis (1989)	crew scheduling
Desrosiers et al. (1984)	multiple traveling salesman problem with time windows
Krumke et al. (2002)	real-time dispatching of automobile service units
Lübbecke and Zimmermann (2003); Sol (1994)	multiple pickup and delivery problem with time windows
Anbil et al. (1998); Crainic and Rousseau (1987)	airline crew pairing
Barnhart and Schneur (1996)	air network design for express shipment service
Erdmann et al. (2001)	airline schedule generation
Barnhart et al. (1998a); Desaulniers et al. (1997); Ioachim et al. (1999)	fleet assignment and aircraft routing and scheduling
Crama and Oerlemans (1994)	job grouping for flexible manufacturing systems
Eben-Chaime et al. (1996)	grouping and packaging of electronic circuits
Park et al. (1996)	bandwidth packing in telecommunication networks
Ribeiro et al. (1989)	traffic assignment in satellite communication systems
Sankaran (1995)	course registration at a business school
Vanderbeck (1994)	graph partitioning e.g., in VLSI, compiler design
Vanderbeck (1994)	single-machine multi-item lot-sizing
Hurkens et al. (1997); Valério de Carvalho (1999, 2000, 2002b); Vance (1998); Vance et al. (1994); Vanderbeck (1999)	bin packing and cutting stock problems
Alvelos and Valério de Carvalho (2000); Barnhart et al. (1997, 2000)	integer multicommodity flows
Bourjolly et al. (1997)	maximum stable set problem
Hansen et al. (1998)	probabilistic maximum satisfiability problem
Johnson et al. (1993)	minimum cut clustering
Mehrotra and Trick (1996)	graph coloring
Savelsbergh (1997)	generalized assignment problem

Table 1: Some applications of integer programming column generation.

decomposition only; see for example Benders (1962), Lasdon (1970), and Van Roy (1983) for different points of view.

2.1 Column Generation

Let us call the following linear program the *master problem* (MP).

$$\begin{array}{rrcl} z^\star := \min & \sum_{j\in J} c_j\lambda_j & & \\ \text{subject to} & \sum_{j\in J} \mathbf{a}_j\lambda_j & \geq & \mathbf{b} \\ & \lambda_j & \geq & 0, \;\; j\in J \end{array} \tag{1}$$

In each iteration of the simplex method we look for a non-basic variable to price out and enter the basis. That is, in the *pricing step*, given the vector $\mathbf{u} \geq \mathbf{0}$ of dual variables, we wish to find

$$\arg\min\left\{\bar{c}_j := c_j - \mathbf{u}^T\mathbf{a}_j \mid j \in J\right\} \;\; . \tag{2}$$

An *explicit* search of J may be computationally impossible when $|J|$ is huge. Instead, we work with a reasonably small subset $J' \subseteq J$ of columns, with a *restricted master problem* (RMP). Assuming that we have a feasible solution, let $\bar{\boldsymbol{\lambda}}$ and $\bar{\mathbf{u}}$ be primal and dual optimal solutions of the RMP, respectively. When columns $\mathbf{a}_j$, $j \in J$, are *implicitly* given as elements of a set $\mathcal{A} \neq \emptyset$, and the cost coefficient c_j can be computed from $\mathbf{a}_j$ then the *subproblem* or *oracle*

$$\bar{c}^\star := \min\left\{c(\mathbf{a}) - \bar{\mathbf{u}}^T\mathbf{a} \mid \mathbf{a} \in \mathcal{A}\right\} \tag{3}$$

returns an answer to the pricing. If $\bar{c}^\star \geq 0$, no reduced cost coefficient $\bar{c}_j$ is negative and $\bar{\boldsymbol{\lambda}}$ (embedded in $\mathbb{R}^{|J|}$) optimally solves the master problem as well. Otherwise, we add to the RMP a column derived from the oracle's answer, and repeat with re-optimizing the RMP. For its role in the algorithm, (3) is also called the *generation problem*, or the *column generator*.

The advantage of solving an optimization problem in (3) instead of an enumeration in (2) becomes even more apparent when we remember that vectors $\mathbf{a} \in \mathcal{A}$ often encode combinatorial objects like paths, sets, or permutations. Then, $\mathcal{A}$ and the interpretation of cost are naturally defined on these structures, and we are provided with valuable information about what possible columns "look like."

Consider the one-dimensional cutting stock problem, the classical example in column generation introduced by Gilmore and Gomory (1961). Given are paper rolls of width W, and m demands b_i, $i = 1, \ldots, m$, for orders of width w_i. The goal is to minimize the number of rolls to be cut into orders, such that the demand is satisfied. A standard formulation is

$$\min\{\mathbf{1}^T\boldsymbol{\lambda} \mid A\boldsymbol{\lambda} \geq \mathbf{b},\; \boldsymbol{\lambda} \in \mathbb{Z}_+^{|J|}\} \;\; , \tag{4}$$

where A encodes the set of $|J|$ feasible cutting patterns, i.e., $a_{ij} \in \mathbb{Z}_+$ denotes how often order i is obtained when cutting a roll according to $j \in J$. From the definition of feasible patterns, the condition $\sum_{i=1}^m a_{ij}w_i \leq W$ must hold for every $j \in J$, and λ_j determines how often the cutting pattern $j \in J$ is used. The linear relaxation of (4) is then solved via column generation, where the pricing problem is a *knapsack problem*.

With the usual precautions against cycling of the simplex method, column generation is finite and exact. In addition, we have a knowledge about the solution quality during the process. Let $\bar{z}$ denote the optimal objective function value to the RMP. Note that by duality we have $\bar{z} = \bar{\mathbf{u}}^T\mathbf{b}$. Interestingly, when an upper bound $\kappa \geq \sum_{j \in J} \lambda_j$ holds for an optimal solution of the master problem, we establish not only an upper bound on $z^\star$ in each iteration, but also a lower bound: We cannot reduce $\bar{z}$ by more than κ times the smallest reduced cost $\bar{c}^\star$, hence

$$\bar{z} + \bar{c}^\star \kappa \leq z^\star \leq \bar{z} \ . \tag{5}$$

In the optimum of (1), $\bar{c}^\star = 0$ for the basic variables, and the bounds close. The lower bound in (5) is computationally cheap and readily available when (3) is solved to optimality. When the objective already is a sum of the variables, that is, $\mathbf{c} \equiv \mathbf{1}$, we use $z^\star$ instead of κ and obtain the improved lower bound $\bar{z}/(1-\bar{c}^\star) \leq z^\star$. For $\mathbf{c} \geq \mathbf{0}$ Farley (1990) proposes a more general lower bound at the expense of a slightly increased computational effort. Let $j' \in \arg\min_{j \in J}\{c_j/\bar{\mathbf{u}}^T\mathbf{a}_j \mid \bar{\mathbf{u}}^T\mathbf{a}_j > 0\}$. Then

$$\bar{z} \cdot c_{j'}/\bar{\mathbf{u}}^T\mathbf{a}_{j'} \leq z^\star \leq \bar{z} \ . \tag{6}$$

Valério de Carvalho (2002b); Vance (1998), and Vance et al. (1994) tailor this to the cutting stock problem with $\mathbf{c} \equiv \mathbf{1}$. See Hurkens et al. (1997) for an implementation of both bounds.

2.2 The Decomposition Principle in Linear Programming

We briefly review the classical decomposition principle in linear programming, due to Dantzig and Wolfe (1960). Consider a linear program (the *original* or *compact formulation*)

$$\begin{array}{rrcl} z^\star := \min & \mathbf{c}^T\mathbf{x} & & \\ \text{subject to} & A\mathbf{x} & \geq & \mathbf{b} \\ & D\mathbf{x} & \geq & \mathbf{d} \\ & \mathbf{x} & \geq & \mathbf{0} \ . \end{array} \tag{7}$$

Let $P = \{\mathbf{x} \in \mathbb{R}^n_+ \mid D\mathbf{x} \geq \mathbf{d}\} \neq \emptyset$. It is well known (Schrijver, 1986) that we can write each $\mathbf{x} \in P$ as convex combination of extreme points $\{\mathbf{p}_q\}_{q \in Q}$ plus non-negative combination of extreme rays $\{\mathbf{p}_r\}_{r \in R}$ of P, i.e.,

$$\mathbf{x} = \sum_{q \in Q} \mathbf{p}_q \lambda_q + \sum_{r \in R} \mathbf{p}_r \lambda_r, \quad \sum_{q \in Q} \lambda_q = 1, \quad \boldsymbol{\lambda} \in \mathbb{R}^{|Q|+|R|}_+ \tag{8}$$

where the index sets Q and R are finite. Substituting for $\mathbf{x}$ in (7) and applying the linear transformations $c_j = \mathbf{c}^T\mathbf{p}_j$ and $\mathbf{a}_j = A\mathbf{p}_j$, $j \in Q \cup R$ we obtain an equivalent *extensive formulation*

$$\begin{array}{rrcl} z^\star := \min & \displaystyle\sum_{q \in Q} c_q \lambda_q + \sum_{r \in R} c_r \lambda_r & & \\ \text{subject to} & \displaystyle\sum_{q \in Q} \mathbf{a}_q \lambda_q + \sum_{r \in R} \mathbf{a}_r \lambda_r & \geq & \mathbf{b} \\ & \displaystyle\sum_{q \in Q} \lambda_q & = & 1 \\ & \boldsymbol{\lambda} & \geq & \mathbf{0} \ . \end{array} \tag{9}$$

It typically has a large number $|Q| + |R|$ of variables, but possibly substantially fewer rows than (7). The equation $\sum_{q \in Q} \lambda_q = 1$ is referred to as the *convexity constraint*. If $\mathbf{x} \equiv \mathbf{0}$ is feasible for P in (7) at zero cost it may be omitted in Q. The convexity constraint is then replaced by $\sum_{q \in Q} \lambda_q \leq 1$.

Although the compact and the extensive formulations are equivalent in that they give the same optimal objective function value $z^\star$, the respective polyhedra are not combinatorially equivalent (Adler and Ülkücü, 1973; Nazareth, 1987). As (8) suggests, $\mathbf{x}$ uniquely reconstructs from a given $\boldsymbol{\lambda}$, but not vice versa.

Given a dual optimal solution $\bar{\mathbf{u}}, \bar{v}$ to the RMP obtained from (9), where variable v corresponds to the convexity constraint, the subproblem (3) in Dantzig-Wolfe decomposition is to determine $\min_{j\in Q\cup R}\{c_j - \bar{\mathbf{u}}^T\mathbf{a}_j - \bar{v}\}$. By our previous linear transformation this results in

$$\bar{c}^\star := \min\left\{(\mathbf{c}^T - \bar{\mathbf{u}}^T A)\mathbf{x} - \bar{v} \mid D\mathbf{x} \geq \mathbf{d}, \mathbf{x} \geq \mathbf{0}\right\} \ . \tag{10}$$

This is a linear program again. We assumed $P \neq \emptyset$. When $\bar{c}^\star \geq 0$, no negative reduced cost column exists, and the algorithm terminates. When $\bar{c}^\star < 0$ and finite, the optimal solution to (10) is an extreme point $\mathbf{p}_q$ of P, and we add the column $[\mathbf{c}^T\mathbf{p}_q, (A\mathbf{p}_q)^T, 1]^T$ to the RMP. When $\bar{c}^\star = -\infty$ we identify an extreme ray $\mathbf{p}_r$ of P as a homogeneous solution to (10), and we add the column $[\mathbf{c}^T\mathbf{p}_r, (A\mathbf{p}_r)^T, 0]^T$ to the RMP. From (5) together with the convexity constraint we obtain at each iteration

$$\bar{z} + \bar{c}^\star \leq z^\star \leq \bar{z} \ , \tag{11}$$

where $\bar{z} = \bar{\mathbf{u}}^T\mathbf{b} + \bar{v}$ is again the optimal objective function value of the RMP. Note that the lower bound is also valid in the case the subproblem generates an extreme ray, that is, when $\bar{c}^\star = -\infty$. Dantzig-Wolfe type approximation algorithms with guaranteed convergence rates have been proposed for certain linear programs, see Klein and Young (1999), and the references given therein.

2.3 Block Diagonal Structure

The decomposition principle has an interpretation as decentralized planning without complete information at the center, see Chvátal (1983) and Lasdon (1970). In that context many applications have a block diagonal structure of D, i.e.,

$$D = \begin{pmatrix} D^1 & & & \\ & D^2 & & \\ & & \ddots & \\ & & & D^\kappa \end{pmatrix} \qquad \mathbf{d} = \begin{pmatrix} \mathbf{d}^1 \\ \mathbf{d}^2 \\ \vdots \\ \mathbf{d}^\kappa \end{pmatrix} ,$$

where D^k and $\mathbf{d}^k$ are of compatible size. This special structure can be exploited. Each $P^k = \{\mathbf{x} \mid D^k\mathbf{x} \geq \mathbf{d}^k,\ \mathbf{x} \geq \mathbf{0}\}$, $k = 1, \ldots, \kappa$ is independently represented in the sense of (8). A superscript k to the entities c_j^k, $\mathbf{a}_j^k$, and λ_j^k for $j \in Q^k \cup R^k$, indicates the respective subsystem $k \in K := \{1, \ldots, \kappa\}$. In analogy to (9) the master problem reads

$$\begin{aligned} z^\star := \min \quad & \sum_{k\in K}\left(\sum_{q\in Q^k} c_q^k\lambda_q^k + \sum_{r\in R^k} c_r^k\lambda_r^k\right) & & \\ \text{subject to} \quad & \sum_{k\in K}\left(\sum_{q\in Q^k} \mathbf{a}_q^k\lambda_q^k + \sum_{r\in R^k} \mathbf{a}_r^k\lambda_r^k\right) & \geq \ & \mathbf{b} \\ & \sum_{q\in Q^k} \lambda_q^k & = \ & 1, \quad k \in K \\ & \boldsymbol{\lambda}^k & \geq \ & \mathbf{0}, \quad k \in K \ . \end{aligned} \tag{12}$$

Denoting by v^k the dual variable associated with the k^{th} convexity constraint, the κ subproblems are analogues of (10)

$$\bar{c}^{k\star} := \min\{(\mathbf{c}^{kT} - \bar{\mathbf{u}}^T A^k)\mathbf{x}^k - \bar{v}^k \mid D^k\mathbf{x}^k \geq \mathbf{d}^k,\ \mathbf{x}^k \geq \mathbf{0}\}, \quad k \in K \tag{13}$$

and the algorithm terminates when $\bar{c}^{k\star} \geq 0$, for all $k \in K$. Otherwise, extreme points and rays identified in (13) give rise to new columns to be added to the RMP. By linear programming duality, $\bar{z} = \bar{\mathbf{u}}^T\mathbf{b} + \sum_{k\in K} \bar{v}^k$, and we obtain the bounds

$$\bar{z} + \sum_{k\in K} \bar{c}^{k\star} \leq z^\star \leq \bar{z}\ . \tag{14}$$

3 Decomposition of Integer Programs

In almost every application we are interested in optimizing over a discrete set X, that is,

$$\begin{array}{rlcl} z^\star := \min & \mathbf{c}^T\mathbf{x} & & \\ \text{subject to} & A\mathbf{x} & \geq & \mathbf{b} \\ & \mathbf{x} & \in & X\ . \end{array} \tag{15}$$

We consider the case of integer programming where $X = P \cap \mathbb{Z}_+$ and $P \subseteq \mathbb{R}^n$ is a polyhedron. However, X could have a much more complicated non-linear definition (Desaulniers et al., 1998). We assume that $z^\star$ be finite.

3.1 Lagrangian Relaxation

A popular approach to solving (15) is *Lagrangian relaxation*. Relaxing constraints $A\mathbf{x} \geq \mathbf{b}$ and penalizing their violation in the objective function via Lagrangian multipliers $\mathbf{u} \geq \mathbf{0}$ results in the following *Lagrangian subproblem*

$$L(\mathbf{u}) := \min_{\mathbf{x}\in X}\ \mathbf{c}^T\mathbf{x} - \mathbf{u}^T(A\mathbf{x} - \mathbf{b})\ . \tag{16}$$

$L(\mathbf{u})$ is a lower bound on $z^\star$, because $L(\mathbf{u}) \leq \min\{\mathbf{c}^T\mathbf{x} - \mathbf{u}^T(A\mathbf{x} - \mathbf{b}) \mid A\mathbf{x} \geq \mathbf{b}, \mathbf{x} \in X\} \leq z^\star$. The best such bound on $z^\star$ is computed in the *Lagrangian dual problem*

$$\mathcal{L} := \max_{\mathbf{u}\geq\mathbf{0}}\ L(\mathbf{u})\ . \tag{17}$$

Assume we are given optimal multipliers $\mathbf{u}^\star$ for (17). By solving (16), we ensure that $\mathbf{x} \in X$, we can show that $\mathbf{u}^{\star T}(A\mathbf{x} - \mathbf{b}) = 0$ (complementary slackness), but $A\mathbf{x} \geq \mathbf{b}$ (feasibility) has to be verified to prove optimality. If this condition is violated, the primal-dual pair $(\mathbf{x}, \mathbf{u}^\star)$ is not optimal.

The Lagrangian function $L(\mathbf{u}), \mathbf{u} \geq \mathbf{0}$ is the lower envelope of a family of functions linear in $\mathbf{u}$, and therefore is concave. It is piecewise linear and only subdifferentiable in its breakpoints. The most popular, since very easy to implement, and well documented choice to obtain (near) optimal multipliers are subgradient algorithms (Wolsey, 1998). An alternative way is by linear programming. Replace X by $\operatorname{conv}(X)$ in (15); this does not change $z^\star$. Changing the Lagrangian subproblem and the dual accordingly, we are enabled to write (16) and (17) in terms of extreme points and rays of $\operatorname{conv}(X)$. This turns the Lagrangian dual into a linear program, and for a given vector $\bar{\mathbf{u}}$ of multipliers

$$L(\bar{\mathbf{u}}) \;=\; (\bar{\mathbf{u}}^T\mathbf{b} + v) + \min_{\mathbf{x}\in\operatorname{conv}(X)} (\mathbf{c}^T - \bar{\mathbf{u}}^T A)\mathbf{x} - v \;=\; \bar{z} + \bar{c}^\star\ , \tag{18}$$

that is, the lower bound obtained from the RMP in Dantzig-Wolfe decomposition in (11) is the same as the Lagrangian bound.

3.2 Convexification

The strong relation of Dantzig-Wolfe decomposition and Lagrangian relaxation is now investigated (see Nemhauser and Wolsey, 1988). When $X = \emptyset$, which may happen during branch-and-bound, then $\mathcal{L} = \infty$ in (17). Otherwise, let again Q and R denote the index sets of extreme points and extreme rays, respectively, of $\mathrm{conv}(X)$. For given multipliers $\mathbf{u}$, the Lagrangian bound is

$$L(\mathbf{u}) = \begin{cases} -\infty & \text{if } (\mathbf{c}^T - \mathbf{u}^T A)\mathbf{p}_r < 0 \text{ for some } r \in R \\ \mathbf{c}^T \mathbf{p}_q - \mathbf{u}^T(A\mathbf{p}_q - \mathbf{b}) & \text{for some } q \in Q \text{ otherwise.} \end{cases} \tag{19}$$

Since we assumed $z^\star$ to be finite, we wish to avoid $L(\mathbf{u}) = -\infty$. We state this as follows

$$\begin{array}{lll} \mathcal{L} = & \max\limits_{\mathbf{u}\geq\mathbf{0}} \min\limits_{q\in Q} & \mathbf{c}^T\mathbf{p}_q - \mathbf{u}^T(A\mathbf{p}_q - \mathbf{b}) \\ \text{subject to} & & (\mathbf{c}^T - \mathbf{u}^T A)\mathbf{p}_r \geq 0, \quad r \in R \end{array} \tag{20}$$

or as a linear program with many constraints

$$\begin{array}{llcll} \mathcal{L} = \max & v & & & \\ \text{subject to} & \mathbf{u}^T(A\mathbf{p}_q - \mathbf{b}) + v & \leq & \mathbf{c}^T\mathbf{p}_q, & q \in Q \\ & \mathbf{u}^T A\mathbf{p}_r & \leq & \mathbf{c}^T\mathbf{p}_r, & r \in R \\ & \mathbf{u} & \geq & \mathbf{0} \ . & \end{array} \tag{21}$$

The dual linear program of (21), composed of many variables, reads

$$\begin{array}{llcl} \mathcal{L} = \min & \sum\limits_{q\in Q} \mathbf{c}^T\mathbf{p}_q\lambda_q + \sum\limits_{r\in R} \mathbf{c}^T\mathbf{p}_r\lambda_r & & \\ \text{subject to} & \sum\limits_{q\in Q} A\mathbf{p}_q\lambda_q + \sum\limits_{r\in R} A\mathbf{p}_r\lambda_r & \geq & \mathbf{b}\sum\limits_{q\in Q}\lambda_q \\ & \sum\limits_{q\in Q}\lambda_q & = & 1 \\ & \boldsymbol{\lambda} & \geq & \mathbf{0} \ . \end{array} \tag{22}$$

Consequently, we can solve (17) by either (21) or (22); the former gives us the multipliers, but the latter provides us with an $\mathbf{x}$ feasible to (15) via the substitution (8). Moreover, in (22) we get complementary slackness and feasibility of $A\mathbf{x} \geq \mathbf{b}$ for free, which is *not* the case in subgradient algorithms. Also $\mathbf{x} \in \mathrm{conv}(X)$; therefore only one issue remains to be checked: The integrality of $\mathbf{x}$, cf. §7.

Applying Dantzig-Wolfe decomposition to (15) with X replaced by $\mathrm{conv}(X)$, we directly obtain the above correspondence. This explains the notion of *convexification* (Vanderbeck, 1994).

$$\begin{aligned}
z^\star := \min \quad & \sum_{q\in Q} c_q\lambda_q + \sum_{r\in R} c_r\lambda_r \\
\text{subject to} \quad & \sum_{q\in Q} \mathbf{a}_q\lambda_q + \sum_{r\in R} \mathbf{a}_r\lambda_r \quad \geq \quad \mathbf{b} \\
& \qquad\qquad\qquad \sum_{q\in Q} \lambda_q \quad = \quad 1 \\
& \qquad\qquad\qquad\qquad \boldsymbol{\lambda} \quad \geq \quad \mathbf{0} \\
& \qquad\qquad\qquad\qquad \mathbf{x} \quad = \quad \sum_{q\in Q} \mathbf{p}_q\lambda_q + \sum_{r\in R} \mathbf{p}_r\lambda_r \\
& \qquad\qquad\qquad\qquad \mathbf{x} \quad \in \quad \mathbb{Z}_+^n \ ,
\end{aligned} \tag{23}$$

where again $c_j = \mathbf{c}^T\mathbf{p}_j$ and $\mathbf{a}_j = A\mathbf{p}_j$, $j \in Q \cup R$. When we relax the integrality of $\mathbf{x}$, there is no need to link $\mathbf{x}$ and $\boldsymbol{\lambda}$ in (23), and we may also relax the coupling constraint, obtaining precisely (22). Still, in order to get integer solutions, we have to impose additional conditions on the $\mathbf{x}$ variables. These conditions will appear in the compact formulation (15), at the level of the master problem or of the subproblem, or both, and the decomposition process—such as Lagrangian relaxation or Dantzig-Wolfe decomposition—has to be repeated at every node of the search tree.

One objection against subgradient algorithms for solving the Lagrangian dual is that they exploit only local information for the iterative update of the dual multipliers. On the contrary, solving an RMP based on (22) is a more elaborate update strategy which makes use of all the information gathered during the solution process. Only, the partly occurring large linear programs could not be solved until recently. Still, when the number of rows, and thus dual multipliers is very large, subgradient algorithms may be the only practical alternative. Hybrid methods are good compromises (Barahona and Jensen, 1998; Kallehauge, Larsen, and Madsen, 2001; Kohl and Madsen, 1997), e.g., starting the multiplier adjustment with a subgradient algorithm and finishing the computation using a linear program.

Besides subgradients and simplex-based methods, the Lagrangian dual can be solved with more advanced (non-linear) alternatives with stronger convergence properties. Among them are the *bundle method* (Hiriart-Urruty and Lemaréchal, 1993) based on quadratic programming, and the *analytic center cutting plane method* (Goffin and Vial, 2003), an interior point solution approach. However, the performance of these alternatives is still to be evaluated in the context of integer programming.

3.3 Discretization

Requiring integrality of the variables $\boldsymbol{\lambda}$ of the master problems (22) or (23) does not lead to an integer program equivalent to (15), since the optimum integer solution of (15) may be an interior point of $\text{conv}(X)$. Alternatively, *discretization* (Johnson, 1989; Vanderbeck, 2000) is a true integer analogue to the decomposition principle. It is based on the following (see Nemhauser and Wolsey, 1988).

Theorem 1 *Let $P = \{\mathbf{x} \in \mathbb{R}^n \mid D\mathbf{x} \geq \mathbf{d}, \mathbf{x} \geq \mathbf{0}\} \neq \emptyset$ and $X = P \cap \mathbb{Z}^n$. Then there exists a finite set of integer points $\{\mathbf{p}_q\}_{q\in Q} \subseteq X$ and a finite set of integer rays $\{\mathbf{p}_r\}_{r\in R}$ of P such that*

$$X = \left\{ \mathbf{x} \in \mathbb{R}_+^n \mid \mathbf{x} = \sum_{q\in Q} \mathbf{p}_q\lambda_q + \sum_{r\in R} \mathbf{p}_r\lambda_r,\ \sum_{q\in Q} \lambda_q = 1,\ \boldsymbol{\lambda} \in \mathbb{Z}_+^{|Q|+|R|} \right\} \ . \tag{24}$$

Substitution for $\mathbf{x}$ in (15) as given by (24) yields an *integer* master problem

$$\begin{aligned} z^\star := \min \quad & \sum_{q\in Q} c_q\lambda_q + \sum_{r\in R} c_r\lambda_r \\ \text{subject to} \quad & \sum_{q\in Q} \mathbf{a}_q\lambda_q + \sum_{r\in R} \mathbf{a}_r\lambda_r \quad \geq \quad \mathbf{b} \\ & \qquad\qquad \sum_{q\in Q} \lambda_q \quad = \quad 1 \\ & \qquad\qquad\quad \lambda_j \quad \in \quad \mathbb{Z}_+, \quad j \in Q \cup R \ , \end{aligned} \tag{25}$$

where again $c_j = \mathbf{c}^T\mathbf{p}_j$ and $a_j = A\mathbf{p}_j$, $j \in Q \cup R$. It is interesting that, when X is bounded, (25) is a linear integer program even for arbitrary linear and non-linear cost functions $c(\mathbf{x})$ in (15). Because of the convexity constraint, variables $\lambda_q, q \in Q$ are restricted to binary values, thus

$$c(\mathbf{x}) = c\left(\sum_{q\in Q} \mathbf{p}_q\lambda_q\right) = c(\mathbf{p}_{q^\star}) = c_{q^\star} = \sum_{q\in Q} c_q\lambda_q$$

holds for precisely one $q^\star \in Q$. This cannot be generalized to the unbounded case, since there need not be a unique representation of $\mathbf{x}$ as a combination of the $\boldsymbol{\lambda}$'s. This is the case when we can combine a point in X by a point in Q and several extreme rays. Then the objective function value $c(\mathbf{x})$ depends on the actual combination.

Remark. In the important special case that $X \subseteq [0,1]^n$ convexification and discretization coincide. All integral points in the bounded set X are already vertices of $\mathrm{conv}(X)$, and only binary $\lambda_q, q \in Q$ in (24) make sense. That is, $\mathbf{x}$ is the trivial convex combination of only one vertex, and therefore $\mathbf{x} \in \{0,1\}^n \iff \boldsymbol{\lambda} \in \{0,1\}^{|Q|}$. Many large scale applications belong to this class, in particular, many decomposition procedures that give rise to set partitioning and set covering problems.

3.4 Column Generation for Integer Programs

Consider the following integer program (IP)

$$\begin{aligned} z^\star := \min \quad & \sum_{j\in J} c_j\lambda_j \\ \text{subject to} \quad & \sum_{j\in J} \mathbf{a}_j\lambda_j \quad \geq \quad \mathbf{b} \\ & \qquad \lambda_j \quad \in \quad \mathbb{Z}_+, \quad j \in J \end{aligned} \tag{26}$$

for which the linear relaxation is solved by column generation using a given oracle. Unlike the situation in §3.2 we do not have available an explicit compact formulation on which we can analyze the solution of the linear relaxation of (26) and decide about how to branch.

Villeneuve et al. (2003) constructively show that a compact formulation equivalent to (26) exists under very mild assumptions. Their proposal involves a duplication of the variables and the domain of the oracle in such a way that a block diagonal structure with identical subproblems results. Their general branching strategy works on the variables of the compact formulation. Multicommodity flow formulations for various applications of vehicle routing and crew scheduling proposed by Desaulniers et al. (1998) follow this scheme. For the classical cutting stock problem (4), the above procedure leads to Kantorovich's formulation (1960) where a commodity is defined for each (identical) available roll. Let K be the set of rolls of width W. Define $y^k, k \in K$ as a binary variable assuming value 1 if roll k is used and 0 otherwise, and $x_i^k, k \in K, i = 1, \ldots, m$ as a non-negative integer variable that denotes the number of times order i is cut in roll k. The compact formulation reads as follows:

$$
\begin{array}{lrcll}
\min & \sum_{k \in K} y^k & & & \\
\text{subject to} & \sum_{k \in K} x_i^k & \geq & b_i, & i = 1, \ldots, m \\
& \sum_{i=1,\ldots,m} w_i x_i^k & \leq & W y^k, & k \in K \\
& y^k & \in & \{0,1\} & k \in K \\
& x_i^k & \in & \mathbb{Z}_+ & k \in K, i = 1, \ldots, m.
\end{array}
\tag{27}
$$

There are alternative compact formulations which lead to the same linear relaxation when the decomposition principle of §3.2 is used. Valério de Carvalho (1999, 2002a) proposes a network-based compact formulation where the classical knapsack subproblem is solved as a particular minimum cost flow problem. Each subproblem path flow in that network gives a valid cutting pattern, and it corresponds to an extreme ray, except the null pattern which is the single extreme point. However, no master problem is used but each generated column is split into its arc components, i.e., in terms of the original arc flow variables. A *restricted compact problem* is solved on the current subset of arcs, and the remaining arcs have to be priced out in order to prove optimality or to identify arcs to be included in the formulation. The flow conservation constraints are activated only as needed, i.e., for the terminal nodes of the generated arcs. Similar techniques to solve large scale linear multicommodity flow problems are used by Löbel (1997, 1998) and Mamer and McBride (2000).

Some imagination is needed to find an appropriate compact formulation, but we strongly recommend to look for one. Once this is established, finding integer solutions to the compact formulation is theoretically no more difficult than for any integer program, only a lower bound is computed via column generation. A valid argument to prefer an extensive formulation over its compact counterpart is that the former may be stronger in the sense that its linear programming relaxation gives a tighter approximation to the convex hull of integer points, see §5.3.1 on the integrality property.

Let us finally stress again the need for generating columns in every node of the branch-and-bound tree. There are cases where the master problem has to be solved in integers, but it is well known that the linear program RMP may be integer infeasible (Barnhart et al., 1998b). Villeneuve et al. (2003) demonstrate the defect that even if feasibility could be ensured, without branching we may miss (probably all) optimal integer solutions. Consider the following example, if $c_2 > 1$, variable z_2 cannot be generated using Dantzig's rule:

$$
\begin{array}{rccccccc}
\min & z_1 & + & c_2 z_2 & + & z_3 & & \\
\text{subject to} & z_1 & + & 2z_2 & + & 3z_3 & = & 2 \\
& z_1, & & z_2, & & z_3 & \in & \mathbb{Z}_+ \ .
\end{array}
\tag{28}
$$

Given the dual variable $u \in \mathbb{R}$ associated with the equality constraint, z_2 is of minimum reduced cost if and only if $c_2 - 2u \leq 1 - u$ and $c_2 - 2u \leq 1 - 3u$, that is, if $c_2 \leq 1 - |u|$, in contradiction to $c_2 > 1$. If $1 < c_2 < 2$ the unique optimal integer solution is $(z_1, z_2, z_3) = (0, 1, 0)$ of value c_2 while the solution restricted to the variables that can be generated is $(z_1, z_3) = (2, 0)$ of cost $2 > c_2$.

4 The Restricted Master Problem

The purpose of the RMP (as is the purpose of subgradient algorithms in Lagrangian relaxation) is to provide dual variables: To be transferred to the subproblem, and to control our stopping criterion. In the end only, we have to recover from the RMP a primal feasible solution to the compact formulation.

Primal methods, like column generation, maintain primal feasibility and work towards dual feasibility. It is therefore only natural to monitor the dual solution in the course of the algorithm. In our opinion, the dual point of view reveals most valuable insight into the algorithm's functioning. We call the polyhedron associated with the dual of the RMP the *dual polyhedron*. A dual solution to the RMP needs not be unique, e.g., if the primal is degenerate. This is significant inasmuch the dual solution directly influences the selection of new columns. Since a dual basic solution corresponds to an extreme point of the optimal face, it may be a bad representative of all the dual solutions obtainable.

Solving the RMP by the simplex method leads to an optimal basis essentially chosen at random, whereas the application of an interior point method produces a solution in the relative interior of the optimal face (Bixby et al., 1992). Therefore, e.g., *analytic centers* (Elhedhli and Goffin, 2004; Goffin et al., 1993), *volumetric centers*, and *central path* methods (Kirkeby Martinson and Tind, 1999) have been proposed. However, even such a solution only refers to the current optimal linear combination of columns generated so far. One step further, dual variable values could represent, in a well defined sense, all information presently available. This is the idea of using *central prices*. In the context of Dantzig-Wolfe decomposition it has been proposed by Goffin et al. (1993). The computational use of various of these proposals for obtaining integer solutions if is evaluated by Briant et al. (2004).

Extreme point dual solutions are immediately available when using the simplex method, and because of their "random" nature they may result in different, even complementary kinds of columns (Vanderbeck, 1994). More elaborate suggestions in the spirit of the above may speed up computation times for very difficult RMPs, see also Anbil, Forrest, and Pulleyblank (1998).

4.1 Solution Methods

4.1.1 Initialization

No matter what method we use, we have to initialize the RMP. The well known simplex first phase carries over to column generation (Chvátal, 1983). Artificial variables, one for each constraint, penalized by a "big M" cost, are kept in the RMP to ensure feasibility in a branch-and-bound algorithm. A smaller M gives a tighter upper bound on the respective dual variables, and may reduce the *heading-in effect* (Vanderbeck, 2004) of initially producing irrelevant columns. Details on initialization, especially for mixed integer programs, are given by Vanderbeck (1994, 2004).

In some applications, the unit basis is already feasible. Then, an estimate of the actual cost coefficients should be used instead of M. Heuristic estimates of the optimal dual variable values are imposed as artificial upper bounds by Agarwal, Mathur, and Salkin (1989) by introducing unit columns with appropriate cost. The bounds are gradually relaxed until they are no longer binding. This relates to the stabilization approach, see §6.2. Similar techniques are proposed, e.g., by Vanderbeck (2004).

Poorly chosen initial columns lead the algorithm astray, when they do not resemble the structure of a possible optimal solution at all. They must then be interpreted as a misleading bound on an irrelevant linear combination of the dual variables. Even an excellent initial integer solution is detrimental to solving a linear program by column generation (Vanderbeck, 1994). On the other hand, bounds on meaningful linear combinations of dual variables give good experiences (Ben Amor, 2002; Valério de Carvalho, 2000), cf. §4.2.1. Another option is a warm start from primal solutions obtained in earlier, similar runs (Anbil, Forrest, and Pulleyblank, 1998). However, the best results are obtained when both estimates of the primal and the dual solutions are used (Ben Amor, 2002; du Merle et al., 1999).

4.1.2 Traditional Approaches: Simplex and Barrier

As is the case for general linear programs, depending on the application and the available solvers, we do not know beforehand which of the several traditional ways of solving the RMP will perform "best." Lasdon (1970) comments on the suitability of primal, dual, and primal-dual simplex methods. In presence of primal degeneracy, the dual simplex may be preferred to the primal. The *sifting* method can be a reasonable complement for large scale RMPs, see Anbil, Forrest, and Pulleyblank (1998); Bixby et al. (1992), and Chu, Gelman, and Johnson (1997). For some linear programs Barrier methods (Bixby, 2002) can prove most effective, although there is no possible warm start.

4.1.3 Subgradients and Volume Algorithm

The RMP may itself be solved by subgradient algorithms by relaxing all its constraints in the objective function (Caprara, Fischetti, and Toth, 1999, 2000; Wedelin, 1995). This approach has been applied to set covering applications for which these authors provide ways to compute a number of integer primal solutions at each iteration of the column generation process. The solution method may be dynamically switched during the solution process like e.g., by Bixby et al. (1992). Because of these alternatives, we do not even need to maintain a basis.

Rather than computing an exact solution to the RMP, an approximation may suffice. Assuming P is non-empty and bounded, the *volume algorithm* (Barahona and Anbil, 2000) is an extension of subgradient algorithms, and rapidly produces primal as well as dual approximate solutions to the RMP. It is named after a new way of looking at linear programming duality, using volumes below the active faces to compute the dual variable values and the direction of movement. The pricing subproblem is called with a dual solution "in a neighborhood" of an optimal dual solution. Then, one can compute the probability that a particular column (which induces a face of the dual polyhedron) is generated. The subgradient method is modified in order to furnish estimates of these probabilities, i.e., approximate primal values for $\lambda_q, q \in Q$ that sum to 1. Primal feasibility may be mildly violated.

When used in alternation with the simplex method, the volume algorithm produces dual solutions with a large number of non-zero variables (Anbil, Forrest, and Pulleyblank, 1998). This quality is claimed to accelerate column generation for reasons as discussed above. Promising computational experience is given for various combinatorial optimization problems (Barahona and Anbil, 2002; Barahona and Jensen, 1998). Advantages of the volume algorithm are a straight forward implementation with small memory requirements, numerical stability, and fast convergence.

Elaborate hybrids may evolve, see e.g., Anbil, Forrest, and Pulleyblank (1998); Bixby et al. (1992); Borndörfer et al. (2003); Desaulniers, Desrosiers, and Solomon (2001b). Heuristics are used to construct or improve dual variable values at any time in the algorithm. The choice of a method in general will also depend on how fast or how accurate a solution is needed, whether particular problem structures are present, and what implementation skills or solvers are available. One apparent trend is not to insist on optimality, but to attack even larger problems, for which still a guaranteed approximation quality can be obtained. This is specially true for integer programs, see §7.1.

4.2 A Dual Point of View

The dual of the RMP is the dual master problem with rows omitted, hence a relaxation. An optimal dual solution obtained from the RMP may still violate constraints of the full dual master problem. Thus, the pricing problem is a separation problem for the dual. Dantzig-Wolfe decomposition and related methods can be interpreted as special cases of Kelley's (1961) cutting plane method devised for solving convex programs. It is sometimes more instructive and revealing to adopt the dual perspective.

Remark. Consequences from the equivalence of separation and optimization (Grötschel, Lovász, and Schrijver, 1988) arise in this context. Exponential size RMP linear programs are polynomially solvable by column generation under the assumption that the pricing problem is solvable in polynomial time (Mehlhorn and Ziegelmann, 2000; Minoux, 1987). Conversely, solving an RMP is $\mathcal{NP}$-hard, if the pricing problem is (Johnson, Mehrotra, and Nemhauser, 1993).

4.2.1 Restriction of the Dual Master Problem

It is known that set partitioning type master problems can be converted to set covering type, preserving the optimal objective function value, when taking subsets does not increase cost. This constrains the dual space by restricting the dual variables in sign. Further, we are not restricted to only add columns from the master program. Using structural information we can do better (Ben Amor, 2002; Valério de Carvalho, 2000; see also Holmberg and Jörnsten, 1995). Consider a pair of feasible and bounded primal and dual master problems $\min\{\mathbf{c}^T\boldsymbol{\lambda} \mid A\boldsymbol{\lambda} = \mathbf{b}, \lambda \geq 0\}$ and $\max\{\mathbf{b}^T\mathbf{u} \mid A^T\mathbf{u} \leq \mathbf{c}\}$ and their extended counterparts of the form

$$\begin{array}{rrcl} \min & \mathbf{c}^T\boldsymbol{\lambda} + \mathbf{f}^T\mathbf{y} & & \\ \text{subject to} & A\boldsymbol{\lambda} + F\mathbf{y} & = & \mathbf{b} \\ & \boldsymbol{\lambda},\, \mathbf{y} & \geq & \mathbf{0} \end{array} \qquad\qquad \begin{array}{rrcl} \max & \mathbf{b}^T\mathbf{u} & & \\ \text{subject to} & A^T\mathbf{u} & \leq & \mathbf{c} \\ & F^T\mathbf{u} & \leq & \mathbf{f} \end{array},$$

where structural inequalities $F^T\mathbf{u} \leq \mathbf{f}$ are added to the dual *at initialization time*, i.e., before column generation starts. We assume that these inequalities do not change the optimal objective function value. These constraints correspond to additional variables $\mathbf{y} \geq \mathbf{0}$ in the primal, which are *not* present in the original master problem. From the primal perspective, we obtain a relaxation. The size of a primal basis is not affected. A *good* restriction of the dual polyhedron is sought, ideally to the optimal face.

Structural inequalities exploit specific problem knowledge. Consider the one-dimensional cutting stock problem (4). It can be easily shown that if the orders are ranked such that $w_1 < w_2 < \cdots < w_m$ then the dual multipliers satisfy $u_1 \leq u_2 \leq \cdots \leq u_m$. Hence we can impose $m-1$ dual constraints that must be satisfied *at each iteration* of the column generation process. These constraints can be generalized to larger sets. Let $S_i = \{s \mid w_s < w_i\}$. Then

$$\sum_{s \in S} w_s \leq w_i \;\Rightarrow\; \sum_{s \in S} u_s \leq u_i, \;\; S \subset S_i \;. \tag{29}$$

A primal interpretation of these constraints is given by Valério de Carvalho (2000); a direct proof of their validity in the dual space can be found in Ben Amor (2002). Using the above $m-1$ simplest dual constraints and, for each order i, at most one constraint of type (29) with $|S| = 2$, there is a considerable speedup for solving the linear relaxation of (4).

We also observe that in the case of the so-called "difficult" triplet-problems, where each roll is cut into exactly three orders without any waste, the optimal dual multipliers are known in advance and assume values $u_i = w_i/W$, $i = 1, \ldots, m$. Using this *a priori* perfect dual information the number of column generation iterations dramatically decreases for a number of test problems (Ben Amor, 2002).

Computational experiments conducted by Ben Amor (2002) on the multiple depot vehicle scheduling problem show that constraining the dual multipliers to a small interval around their optimal values, column generation can be accelerated by a factor of 100. This is because poor dual cutting planes, with respect to the interval, are not generated. Optimal multipliers are usually not available. Good estimates can be obtained from a relaxation of the problem, from tests of previous experiments, or derived by subgradient algorithms. As a further benefit, restricting the dual space partly removes primal degeneracy (Ben Amor, 2002; Valério de Carvalho, 2000). See also §6.2.

4.2.2 Row Aggregation for Set Partitioning Problems

When the master problem is a set partitioning problem, large instances are difficult to solve due to massive degeneracy, say, when the number of non-zero elements per column roughly exceeds ten. Then, the value of the dual variables are no meaningful measure for which column to adjoin to the RMP. As a remedy, Elhallaoui et al. (2003) propose a dynamic row aggregation technique. Their intuition is that in applications like vehicle routing and crew scheduling, some activity sequences are more likely to occur than others: In airline crew scheduling a pilot usually stays in the aircraft with which he starts his duty day. Since aircraft itineraries are known prior to solving the crew pairing problem, it is natural to "guess" some aggregation of the flights to cover. This allows for a considerable reduction of the size of the RMP in each iteration.

No aggregation is done at the level of the subproblem and the cost of a column remains unchanged, regardless of whether aggregated or not. Still, the aggregated RMP provides us with aggregated dual multipliers. These are split into estimates of the dual multipliers for the unaggregated RMP solving shortest path problems, based on the columns already generated and the reduced cost optimality criterion. The estimated duals are used in the subproblem to generate new columns. To ensure proper convergence and optimality, the aggregation is dynamically updated throughout the solution process.

Tests conducted on the linear relaxation of the simultaneous vehicle and bus driver scheduling problem in urban mass transit show that this solution approach significantly reduces the size of the master problem, the degeneracy, and the solution times, especially for larger problems: For an instance with 1600 set partitioning constraints, the RMP solution time is reduced by a factor of eight.

5 The Pricing Problem

We are free to choose a subset of non-basic variables, and a criterion according to which a column is selected from the chosen set. According to the classical Dantzig rule, one chooses among all columns the one with the most negative reduced cost. Various schemes are proposed in the literature like *full*, *partial*, or *multiple pricing* (Chvátal, 1983). Column generation is a pricing scheme for large scale linear programs. In Gamache et al. (1999) up to 300 $\mathcal{NP}$-hard subproblems arise, and partial column generation is used. That is, only a subset of subproblems is chosen at each iteration to generate new columns. This also avoids the generation of many similar columns.

The role of the pricing subproblem is to provide a column that prices out profitably or to prove that none exists. It is important to see that *any* column with negative reduced cost contributes to this aim. In particular, there is no need to solve (3) exactly; an approximation suffices until the last iteration. We may add many negative reduced cost columns from a subproblem, even positive ones are sometimes used. We may solve a temporary restriction of the subproblem, or a relaxation, which is the case for the vehicle routing problem with time windows (Desrochers, Desrosiers, and Solomon, 1992).

5.1 Dominance and Redundancy of Columns

Let us speak about strength of dual constraints. A column with reduced cost $\bar{c}$ is *dominated* if there exists another column with reduced cost no larger than $\bar{c}$ for all dual variables ranging within their respective domains (Vanderbeck, 1994). On the other hand, a column with reduced cost $\bar{c}$ is undominated, if for all other columns there exists a set of dual variables yielding reduced cost strictly larger than $\bar{c}$. If dominance is detected after the solution of the pricing problem, the column is replaced by the dominating column in a post-processing phase. For instance, let $\mathcal{A}$ be the collection of sets of a set covering problem. A column $\mathbf{a}_s$ corresponding to a set $s \subseteq \mathcal{A}$ is dominated, if adding to s an element $r \in \mathcal{A} \setminus \{s\}$ incurs no cost, since $c_s - \mathbf{u}^T\mathbf{a}_s \geq c_s - \mathbf{u}^T\mathbf{a}_s - u_r = c_{s\cup\{r\}} - \mathbf{u}^T\mathbf{a}_{s\cup\{r\}}$ for all $\mathbf{u} \geq \mathbf{0}$.

An even stronger concept is introduced by Sol (1994). By analogy with the search for strong cutting planes, ideally facets of the dual polyhedron, we ask for strong columns in the RMP. A column $\mathbf{a}_s$ is called *redundant* if the corresponding constraint is redundant for the dual problem. That is,

$$\mathbf{a}_s = \sum_{r \subset s} \mathbf{a}_r \lambda_r \quad \text{and} \quad c_s \geq \sum_{r \subset s} c_r \lambda_r \quad . \tag{30}$$

A column is *strictly* redundant if (30) holds with strict inequality. The pair $(\mathcal{A}, \mathbf{c})$ satisfies the *subcolumn property* if $c_r < c_s$, for all subsets $r \subset s \in \mathcal{A}$. In this case, a set partitioning problem can be solved as a set covering problem. If only $c_r \leq c_s$ holds, we modify the cost structure by $c_r := c_r + |r|$. This adds to $z^\star$ a constant term equal to the number of rows and does not change the problem. A characterization of redundant columns in the case of identical subproblems is given by Sol (1994), and a proof that no column is redundant in case of all distinct subproblems. For set partitioning problems with identical subproblems the generation of redundant columns can be avoided using an alternative pricing rule.

Proposition 2 *Let $(\mathcal{A}, \mathbf{c})$ satisfy the subcolumn property for a set partitioning problem and $\bar{\mathbf{u}}$ be a vector of dual multipliers. If $\mathbf{a}_s \in \mathcal{A}$ is a strictly redundant column, then it cannot be an optimal solution to the pricing problem*

$$\min \left\{ \frac{c(\mathbf{a}) - \bar{\mathbf{u}}^T\mathbf{a}}{\mathbf{1}^T\mathbf{a}} \mid \mathbf{a} \in \mathcal{A} \right\} \quad . \tag{31}$$

5.2 Alternative Pricing Rules

Proposition 2 indicates that not using the Dantzig rule may be theoretically advantageous. *Steepest-edge pricing* (Forrest and Goldfarb, 1992; Goldfarb and Reid, 1977) and the practical `Devex` variant (Harris, 1973) are reported to perform particularly well for set partitioning RMPs (Sol, 1994). The rationale behind the dual pendant *deepest-cut* (Vanderbeck, 1994) is to cut away as much of the dual space as possible. While steepest-edge is inherently based on the simplex method, deepest-cut is more independent from a particular solution method. This latter property leads to the *lambda pricing* rule

(Bixby et al., 1992). Assume that $c_j \geq 0, j \in J$. Clearly, the reduced cost $c_j - \bar{\mathbf{u}}^T \mathbf{a}_j$ are non-negative for all $j \in J$ if and only if

$$\min_{j \in J} \left\{ \frac{c_j}{\bar{\mathbf{u}}^T \mathbf{a}_j} \mid \bar{\mathbf{u}}^T \mathbf{a}_j > 0 \right\} \quad \geq 1 \ . \tag{32}$$

At first glance, this is just a reformulation. However, (32) takes advantage of structural properties of (particular) set partitioning problems: Picking columns with a small ratio accounts for smaller cost coefficients as well as for more non-zero entries in $\mathbf{a}_j$.

Many publications on simplex pricing are available, only a few of which have been related to column generation. One proposal is to generate columns which maximally improve the objective function value (Swoveland, 1974). The computational usefulness of such comparably aged proposals needs assessment from a modern implementation point of view. The natural question for which columns serve our goals best is not consistently to answer owing to the multiple, sometimes contrary, evaluation criteria. Computational efforts may cancel theoretical benefits. When the subproblem is solved by dynamic programming, many negative reduced cost columns are available. Among these, *a posteriori* choosing columns according to the alternative proposals is a practicable compromise in view of possible difficulties in efficiently implementing alternative pricing rules.

Still, pricing rules are sensitive to the dual variable values, in case of non-unique dual solutions. Although primal as well as dual information went into pricing strategies, complementary slackness conditions have not been satisfactorily exploited or applied.

5.3 Pricing Integer Programs

When the compact formulation (15) is an integer program, so is the pricing problem. It may be difficult to solve, and the efficiency of decomposition hinges on the question whether repeatedly solving the subproblems is “easier” than solving (15) at once.

5.3.1 Integrality Property

When $\mathrm{conv}(X)$ is an integral polyhedron already the linear program (18) gives an integer solution. This is called the *integrality property* of the subproblem. Of course, it intimately depends on the subproblem formulation. For subproblems whose natural formulation gives integral optimal solutions anyway, e.g., shortest path problems, the lower bound on the optimal integral $z^\star$ in (15) obtained from the linear relaxation of the extensive formulation is no better than the one obtained from the compact formulation (Geoffrion, 1974). On the other hand, when this property does not hold, one has potential to improve the lower bound, but one has to work harder to obtain integral subproblem solutions in the first place. Thus, when the integrality gap is large, one would prefer a subproblem without the integrality property; this also holds for Lagrangian relaxation (Geoffrion, 1974). On the other hand, in presence of the integrality property, the computation time gained by fast combinatorial algorithms and their ease to implement may outmatch the disadvantage of a large gap.

When the polyhedron $\mathrm{conv}(X)$ is well studied like e.g., a knapsack polytope, the relevant literature can be exploited to strengthen the linear relaxation of the pricing integer programs (Vanderbeck, 1994). When problems are small or cuts are of insufficient strength plain branch-and-bound may be the faster alternative. A more thorough investigation of the subsystem polytope can give encouraging results when adding strong valid inequalities to the pricing problem before using branch-and-bound (Johnson, Mehrotra, and Nemhauser, 1993). A combinatorial algorithm to solve the subproblem, if available, may be even more efficient. This is the case for constrained shortest path problems. Dynamic programming algorithms usually provide many columns per iteration. This is to be preferred over adding only one from a linear program.

5.3.2 Structured Sets of Columns

What columns are good for integer solutions? Even columns that are part of an (optimal) integer solution may interfere with solving the linear RMP by influencing its "guide," namely the dual variable values (Vanderbeck, 1994). The RMP may require longer solution times due to the enlarged problem size. Conversely, columns which are of no use for the linear relaxation may be required for the integer feasibility of the RMP. The concept of adjoining *partial solutions* seems to offer significant advantages as for obtaining integer solutions. Desrochers, Desrosiers, and Solomon (1992) remark that adding columns to a set partitioning RMP that are orthogonal sets replicate the structure of the optimal integer solution. Savelsbergh and Sol (1998) observe less similar columns generated when the dual solution is far from optimal.

Lagrangian pricing exploits the problem information provided by the original formulation (Löbel, 1997, 1998). Based on a dual solution to the RMP, one obtains primal solutions from several Lagrangian relaxations and deduces the columns to be adjoined to the RMP. Considerable customization is necessary, e.g., deciding which Lagrangian subproblems to use. Furthermore, the deduction of columns can be non-trivial. Nonetheless, the charm of using Lagrangian relaxations of the compact formulation rests upon controlling the structure of added columns. Very large scale linear programs emerging from practical vehicle routing problems are optimally solved using this pricing scheme.

6 The Tailing Off Effect

Simplex-based column generation is known for its poor convergence. While usually a near optimal solution is approached considerably fast, only little progress per iteration is made close to the optimum. Also, it may be relatively time consuming to prove optimality of a degenerate optimal solution. Figuratively speaking, the solution process exhibits a long tail (Gilmore and Gomory, 1963), hence this phenomenon is called the *tailing off effect*. There is an intuitive assessment of the phenomenon, but a theoretical understanding has only been partly achieved to date; the monographs by Lasdon (1970) and Nazareth (1987) make notable contributions.

6.1 Computational Difficulties

In general, the trajectory of solutions to (7) as constructed from solutions $\boldsymbol{\lambda}$ to the RMP passes through the interior of $\{\mathbf{x} \in \mathbb{R}^n \mid D\mathbf{x} \geq \mathbf{d},\ \mathbf{x} \geq \mathbf{0}\}$. In the optimum, complementary slackness conditions have to be satisfied. Since changes of primal and dual variables are applied iteratively, not simultaneously, a certain amount of "cleaning up" (Nazareth, 1987) is to be expected. Very small adjustments may be necessary close to optimum.

Finding an appropriate combination in (8) might be hindered by the possibly "complex combinatorial structure" of faces of the polyhedron defined by the subproblem (Kim and Nazareth, 1991). This complication may be worse in presence of several subproblems. A problem specific reformulation of the pricing problem in order to *a priori* restrict attention to a set of known simple columns may help (Hurkens, De Jong, and Chen, 1997), see also Barnhart et al. (1998b). The diameter of the polyhedron associated with (9) is not smaller than that of the polyhedron corresponding to (7) (Adler and Ülkücü, 1973). This is interesting, considering that the diameter is the maximal number of steps an ideal vertex following algorithm takes.

In finite precision arithmetic one has to cope with numerical instability. There are examples for bad numerical characteristics of the master problem in contrast to a well behaving compact formulation (Nazareth, 1984, 1987). Then, our stopping criterion may not work correctly (Ho, 1984; Minoux, 1986). We remark, however, that tailing off also occurs when columns are computed exactly, e.g., by use of combinatorial algorithms.

6.2 Stabilized Column Generation

It has been observed that the dual variable values do not smoothly converge to their respective optima, but vehemently oscillate, seemingly following no regular pattern. This behavior is regarded as a major efficiency issue, and its absence is seen as a (possibly *the*) desirable property.

A simple idea was mentioned in §4.1.1, *viz.* bounding the dual variable values (Agarwal, Mathur, and Salkin, 1989). A more sophisticated control of the dual variables is as follows. Let again $\bar{\mathbf{u}}$ denote an optimal solution to the current restricted dual RMP. By imposing lower and upper bounds, respectively, dual variables are constrained to lie in a "box around $\bar{\mathbf{u}}$." The such restricted RMP is re-optimized. If the new dual optimum is attained on the boundary of the box, we have a direction towards which the box should be relocated. Otherwise, the optimum is attained in the box's interior, producing the sought global optimum. This is the principle of the `Boxtep` method introduced by Marsten (1975) and Marsten, Hogan, and Blankenship (1975). Several other stabilization approaches have been proposed, see e.g., Senne and Lorena (2002), where a so-called Langrangian/surrogate relaxation is used for this purpose. In the sequel we describe three of these.

6.2.1 Weighted Dantzig-Wolfe Decomposition

In the computation of the Lagrangian dual (17), one can search "for good Lagrangian multipliers in the neighborhood of the best multipliers found so far" (Wentges, 1997). In lieu of pricing with the optimal dual variables $\bar{\mathbf{u}}^{k+1}$ in the $k+1^{\text{st}}$ iteration, a convex combination is used

$$\bar{\mathbf{u}}^{k+1} := \frac{1}{\omega_k}\bar{\mathbf{u}}^k + \frac{\omega_k - 1}{\omega_k}\bar{\mathbf{u}}^{best,k} \quad , \qquad \bar{\mathbf{u}}^{best,k} \in \arg\max\left\{L(\bar{\mathbf{u}}^i) \mid i = 1,\ldots,k\right\} \quad , \tag{33}$$

where $L(\mathbf{u})$ is defined in (16), and

$$\omega_k := \min\{const,\ (k + \text{ number of improvements of } L(\bar{\mathbf{u}}^{best,\cdot}))/2\}$$

with $const \geq 2$. Obviously, (33) is biased towards the dual solution, which produced the respective best Lagrangian lower bound in the column generation process. This emphasis becomes even stronger as the algorithms proceeds, and grows with the number of improvements of the lower bound. This can be seen as a stabilization of heuristically good multipliers. The constant $const$ is instrumental in ensuring the consideration of enough fresh information from the current RMP.

Rewriting (33) as $\bar{\mathbf{u}}^{k+1} := \bar{\mathbf{u}}^{best,k} + \omega_k^{-1}(\bar{\mathbf{u}}^k - \bar{\mathbf{u}}^{best,k})$ the method is interpreted as feasible direction search, emerging from $\bar{\mathbf{u}}^{best,k}$ in the direction of the current dual solution $\bar{\mathbf{u}}^k$ with step length ω_k^{-1}. Finiteness of this weighted version is proven. In computational experience with capacitated facility location problems, the method delivers better Lagrangian lower bounds, when termination is guided by a small size of the duality gap. On the other hand, the same experiments indicate that the primal objective function value of the RMP decreases more slowly when using this method.

This is an example where ideas from proposals for multiplier adjustment in subgradient methods can be transferred to the column generation context. In fact, the two approaches are combined (Barahona and Jensen, 1998), i.e., every few iterations some or all of the dual variables obtained from the RMP are improved by some iterations of a subgradient algorithm before passing them to the subproblem. In early iterations this produces good multipliers, later on improves the lower bound. Considerably reduced computation times are reported for their particular application. A similar observation is made by Mahey (1986). The voluminous fund of "Lagrangian literature" may further provide stimulation in this direction.

6.2.2 Trust Region Method

It is desirable not having to customize the stabilization device. For a direct control of dual variables (Kallehauge, Larsen, and Madsen, 2001) consider the dual RMP with additional box constraints centered around the current dual optimal solution $\hat{\mathbf{u}}$, i.e., $\hat{u}_i - \delta \leq u_i \leq \hat{u}_i + \delta$. The method is related to the work of Madsen (1975) in the sense that these bounds are adjusted automatically, depending on how well the dual restricted master problem approximates the Lagrangian dual problem. This type of method is called a *trust region method*. The trust region parameter δ is updated in each iteration according to the original update scheme by Marquardt (1963). Only iterations yielding primal progress are actually performed, and Kelley's (1961) cutting plane method is applied to generate rows of the dual RMP, i.e., columns of the primal. When the duality gap closes (up to a preset accuracy) for a dual solution in the interior of the current box, optimality is reached, and the algorithm terminates.

6.2.3 A Stabilization Approach Using Primal and Dual Strategies

Stabilized column generation (Ben Amor, 2002; du Merle et al., 1999) involves a more flexible, linear programming concept of a box, together with an ε-perturbation of the right hand side. Consider the following linear program.

$$\begin{array}{lrcl} \min & \mathbf{c}^T\boldsymbol{\lambda} - \boldsymbol{\delta}_-^T\mathbf{y}_- + \boldsymbol{\delta}_+^T\mathbf{y}_+ & & \\ \text{subject to} & A\boldsymbol{\lambda} - \mathbf{y}_- + \mathbf{y}_+ & = & \mathbf{b} \\ & \mathbf{y}_- & \leq & \boldsymbol{\varepsilon}_- \\ & \mathbf{y}_+ & \leq & \boldsymbol{\varepsilon}_+ \\ & \boldsymbol{\lambda}, \mathbf{y}_-, \mathbf{y}_+ & \geq & \mathbf{0} \end{array} \tag{34}$$

After changing the sign of $\mathbf{w}_-, \mathbf{w}_+$, respectively, its dual reads

$$\begin{array}{lrcl} \max & \mathbf{b}^T\mathbf{u} - \boldsymbol{\varepsilon}_-^T\mathbf{w}_- - \boldsymbol{\varepsilon}_+^T\mathbf{w}_+ & & \\ \text{subject to} & A^T\mathbf{u} & \leq & \mathbf{c} \\ & -\mathbf{u} - \mathbf{w}_- & \leq & -\boldsymbol{\delta}_- \\ & \mathbf{u} - \mathbf{w}_+ & \leq & \boldsymbol{\delta}_+ \\ & \mathbf{w}_-, \mathbf{w}_+ & \geq & \mathbf{0}\ . \end{array} \tag{35}$$

In (34), surplus and slack variables $\mathbf{y}_-$ and $\mathbf{y}_+$, respectively, account for a perturbation of $\mathbf{b}$ by $\boldsymbol{\varepsilon} \in [-\boldsymbol{\varepsilon}_-, \boldsymbol{\varepsilon}_+]$, helping to reduce degeneracy. Their usage is penalized via $\boldsymbol{\delta}_-, \boldsymbol{\delta}_+$, respectively. The interpretation of (35) is more interesting. The original dual variables $\mathbf{u}$ are restricted to the interval

$[\boldsymbol{\delta}_- - \mathbf{w}_-, \boldsymbol{\delta}_+ + \mathbf{w}_+]$, that is, deviation of $\mathbf{u}$ from the interval $[\boldsymbol{\delta}_-, \boldsymbol{\delta}_+]$ is penalized by an amount of $\boldsymbol{\varepsilon}_-, \boldsymbol{\varepsilon}_+$ per unit, respectively. The motivation is to steer $\mathbf{u}$ towards a hopefully good estimate of an optimal solution $\mathbf{u}^\star$ to the unperturbed problem $\min\{\mathbf{c}^T\boldsymbol{\lambda} | A\boldsymbol{\lambda} = \mathbf{b}, \boldsymbol{\lambda} \geq \mathbf{0}\}$. When does (34) yield an optimal solution this problem? Sufficient is (a) $\boldsymbol{\varepsilon}_- = \boldsymbol{\varepsilon}_+ = \mathbf{0}$ or (b) $\boldsymbol{\delta}_- < \hat{\mathbf{u}} < \boldsymbol{\delta}_+$, where $\hat{\mathbf{u}}$ is an optimal solution to (35); (a) for the obvious reason, and (b) by complementary slackness conditions and $\boldsymbol{\varepsilon}_\pm \geq \mathbf{0}$. Therefore the stopping criteria of a column generation algorithm become $\bar{c}^\star = 0$ and $\mathbf{y}_- = \mathbf{y}_+ = \mathbf{0}$.

The parameters are updated dynamically so as to make greatest use of the respective latest information. With intent to reduce the dual variables' variation select $\boldsymbol{\delta}_\pm$ to form a small box containing the (in the beginning estimated) current dual solution, and solve the linear program (34). If the new $\hat{\mathbf{u}}$ lies in the box described by $\boldsymbol{\delta}_\pm$, reduce its width and augment the penalty given by $\boldsymbol{\varepsilon}_\pm$. Otherwise, enlarge the box and decrease the penalty. This allows for fresh dual solutions when our estimate was bad. The update could be performed in each iteration, or alternatively, each time a dual solution of currently best quality is obtained. This latter method proves most effective in implementations. An update of the dual estimates can be seen as a serious step in the *bundle method*.

It is clear that a problem specific adaptation of the parameter choice is needed. The experiments by Ben Amor (2002) and du Merle et al. (1999) indicate considerable speedup of up to a factor of ten or a growth in the size of manageable problems when using the stabilization approach. A related proposal (Agarwal, Mathur, and Salkin, 1989) gives similar experiences. Vanderbeck (2004) concludes that among more sophisticated implementation techniques, stabilization may promise the largest performance gains. It is important to note that (34) is a relaxation of the unperturbed problem which is faster computed, and can be used in a branch-and-bound algorithm.

7 Integer Solutions

Having introduced decomposition techniques for integer programs in §3, we still need ideas on how to actually obtain integer solutions. The literature is rich on that subject, see Table 1. In fact, X may as well contain non-linear aspects other than discreteness. Various notions have been coined for the synthesis of column generation and branch-and-bound, like *branch-and-price* (Barnhart et al., 1998b) and *IP column generation* (Vanderbeck and Wolsey, 1996). Our point is that—besides the decomposition principles—essentially it all boils down to branch-and-bound. Consequently, this section is about lower bounds and branching decisions.

7.1 Lower Bounds and Early Termination

In each node of a branch-and-bound tree we derive lower bounds on the best possible integer solution in the respective branch from solving the RMP linear relaxation by column generation. One would expect that the tailing off effect be amplified by the multitude of linear programs to solve. However, the contrary is true. The need for integer solutions provides us with a very simple amendment: Stop generating columns when tailing off occurs and take a branching decision. This *early termination* is based on the following. Assuming $c_j \in \mathbb{Z}$, $j \in J$, column generation can be terminated as soon as $\lceil LB \rceil = \lceil \bar{z} \rceil$, with LB e.g., one of the lower bounds of §2.1. For this purpose they have been widely used in the literature, e.g., Sol (1994); Vanderbeck (1994); Vanderbeck and Wolsey (1996). With the incumbent integral objective function value $\hat{z}$, a node can be pruned as soon as $\lceil LB \rceil \geq \hat{z}$.

Early termination makes the algorithm effective for integer programs in contrast to linear programs. Of course, we need not wait until $\lceil LB \rceil = \lceil \bar{z} \rceil$, but terminate heuristically even earlier. Here is a tradeoff between computational efforts and the quality of the obtained lower bound upon premature termination. We remind the reader that this is the usual way to stop subgradient algorithms in Lagrangian relaxation. Note that monitoring the relative decrease of the objective function value over a predefined number of iterations (Gilmore and Gomory, 1963) is not robust against temporary stalls.

Integrality helps us also in other places. For a single subproblem the computation of an *a priori* upper bound on $\bar{c}^\star$ is given in (Vanderbeck and Wolsey, 1996, Prop. 4). When the subproblem is solved as an integer program, an initial upper cutoff can be applied. When the pricing problem is detected to be infeasible we terminate.

7.2 Pricing Out the Original $\mathbf{x}$ Variables

Assume that in (15) we have a linear subproblem $X = \{\mathbf{x} \in \mathbb{R}^n_+ \mid D\mathbf{x} \geq \mathbf{d}\} \neq \emptyset$. Column generation then essentially solves the linear program

$$\min\ \mathbf{c}^T\mathbf{x} \quad \text{subject to} \quad A\mathbf{x} \geq \mathbf{b},\ D\mathbf{x} \geq \mathbf{d},\ \mathbf{x} \geq \mathbf{0}\ .$$

We obtain an optimal primal solution $\mathbf{x}^\star$ but only the dual multipliers $\mathbf{u}^\star$ associated with the constraint set $A\mathbf{x} \geq \mathbf{b}$. However, Walker (1969) describes how to retrieve the dual variables $\mathbf{w}^\star$ associated with $D\mathbf{x} \geq \mathbf{d}$: Take the dual vector obtained from solving the linear subproblem in the last iteration of the column generation process. Knowing the full dual information allows for a pricing of the original variables, and therefore a possible elimination of some of them. Together with an integral solution this can be exploited to discard original binary variables with reduced cost larger than the optimality gap. Hadjar, Marcotte, and Soumis (2001) apply this technique to remove more than 90% of the flow variables in multiple depot vehicle scheduling problems.

In the general case of a linear integer or even non-linear pricing subproblem, the above procedure does not work. Poggi de Aragão and Uchoa (2003) suggest to directly use the extensive formulation: If we keep the coupling constraint $\mathbf{x} = \sum_{q\in Q} \mathbf{p}_q\lambda_q + \sum_{r\in R} \mathbf{p}_r\lambda_r$ in the master problem (23), it suffices to impose $\mathbf{x} \geq \boldsymbol{\varepsilon}$, for a small $\boldsymbol{\varepsilon} > \mathbf{0}$, at the end of the process. The shadow prices of these constraints are the reduced costs of the $\mathbf{x}$ vector of original variables. Note that there is no need to impose the additional constraints on already positive variables. Computational experiments underline the benefits of this procedure.

7.3 Branching and Cutting Decisions

A valid branching scheme divides, desirably partitions, the solution space in such a way that the current fractional solution is excluded, integer solutions remain intact, and finiteness of the algorithm is ensured. Moreover, some general rules of thumb prove useful, such as to produce branches of possibly equal size, sometimes referred to as balancing the search tree. Important decisions should be made early in the tree. In the case that the master problem has to be solved integrally, a compatible branching scheme is needed which prevents columns that have been branched on from being regenerated without a significant complication of the pricing problem (Johnson, 1989; Savelsbergh, 1997; Vance, 1998). This, in general, would lead to finding the k^{th} best subproblem solution instead of the optimal one (Ribeiro, Minoux, and Penna, 1989). Aside from the conceptual complication, this modified or destroyed a possibly well exploited structure. This is all the more important when e.g., combinatorial algorithms are used for the subproblem solution.

When integer solution are sought for the master problem, general branching schemes are given by Vanderbeck (2000) and Vanderbeck and Wolsey (1996). The most common scheme in conjunction with column generation is Ryan and Foster's (1981) designed for set partitioning problems.

Proposition 3 *Given $A \in \{0,1\}^{m\times|J'|}$ and a fractional basic solution to $A\boldsymbol{\lambda} = \mathbf{1},\ \boldsymbol{\lambda} \geq \mathbf{0}$, i.e., $\boldsymbol{\lambda} \notin \{0,1\}^m$. Then there exist $r, s \in \{1,\ldots,m\}$ such that $0 < \sum_{j\in J'} a_{rj}a_{sj}\lambda_j < 1$.*

When such two rows are identified, we obtain one branch in which these rows must be covered by the same column, i.e., $\sum_{j\in J'} a_{rj}a_{sj}\lambda_j = 1$, and one branch in which they must be covered by two distinct columns, i.e., $\sum_{j\in J'} a_{rj}a_{sj}\lambda_j = 0$. Note, that this information can be easily transferred to and obeyed by the pricing problem.

This excellent scheme already hints to a powerful insight which is used already in standard branch-and-bound, *viz.* to branch on meaningful sets of variables. Our most valuable source of information are the *original* variables of the compact formulation; they must be integer, and they are what we branch and cut on, see e.g., Desaulniers et al. (1998); Gamache et al. (1998); Sol (1994). To this end, let us assume that we have a compact formulation on hand, see §3.4. Branching and cutting decisions both involve the addition of constraints, at least implicitly. One could require integrality of $\mathbf{x}$ at any node of a branch-and-bound tree (Holm and Tind, 1988), but this is not efficient. A problem specific penalty function method is proposed by Hurkens, De Jong, and Chen (1997). Alternatively, given an added set of constraints, these restrictions on the compact formulation (15) can be incorporated in $A\mathbf{x} \geq \mathbf{b}$, in $\mathbf{x} \in X$, or partially in both structures. In any case, the new problem is of the general form of (15) to which we apply a decomposition. The new RMP is still a linear program, and as long as the possible modification of the subproblem's structure is tractable we face no severe complications.

Consider for example the vehicle routing problem with time windows (Desaulniers et al., 2001a; Desrochers et al., 1992): Decisions can be taken on the network flow variables as well as on the starting time of the service at a customer. Additionally, we may impose subtour elimination constraints, or more generally, κ-path cuts where κ is a lower bound on the number of vehicles needed to service a certain set of customers (Kohl et al., 1999). Also possible are the trivial cuts on the total number of vehicles and on the value of the objective function, decisions on a single arc, or on a linear combination of arc flow and time values.

Ioachim et al. (1999) perform branching on time variables that are already integer, but obtained as a convex combination of several time values. Gamache et al. (1998) impose a very deep cut into the subproblem structure. It does not only cut off the current infeasible solution but at the same time it also removes a number of non-optimal *integer* solutions from the subproblem structure. Generally speaking, a decision imposed on the pricing problem is preferable to one imposed on the master problem as it directly controls the generated columns. An example of that is the 2-cycle elimination for constrained shortest paths with time windows (Desrochers, Desrosiers, and Solomon, 1992), generalized by Irnich and Villeneuve (2003). The choice of the structure on which to impose decisions is a matter of algorithmic efficiency and performance. We remark that adding cutting planes in conjunction with column generation in a branch-and-bound search is usually called *branch-and-price-and-cut*, see e.g., Barnhart et al. (1998a); Barnhart, Hane, and Vance (2000), and Park, Kang, and Park (1996).

Finally, one should be aware that even if a new decision set goes into the master problem structure, the pricing problem may change. Ioachim et al. (1999), in the routing and scheduling area, have linear combinations of time variables which appear in the master problem structure; this has the consequence that these time variables also appear in the objective function of the subproblem together with the flow variables. This changes the way to solve the constrained shortest path problem (Ioachim et al., 1998).

The implementation of a column generation based integer programming code still is an issue. Not so because of the complex interaction of components but because of the vast possibilities to tune each of them. All strategies from standard branch-and-bound apply, including depth first search for early integer solutions, heuristic fathoming of nodes, rounding and (temporary) fixing of variables, pre- and postprocessing, and many more (Desaulniers, Desrosiers, and Solomon, 2001b; Vanderbeck, 2004).

Concluding, two decades ago, Chvátal (1983) saw no efficient way of handling the difficulty of finding an optimal integer solution to a problem solved using a column generation scheme. Today, this is no longer true when a compact formulation is available and columns are generated at each node of the search tree. This fundamental and indeed extremely simple approach has been in use now for more than twenty years (Desrosiers, Soumis, and Desrochers, 1984), and is being refined ever since. The price we have to pay for this simplicity is that besides RMP, subproblem, and branch-and-bound also the compact formulation has to be represented in order to recover a solution in terms of the original variables $\mathbf{x}$.

8 Conclusion

The growing understanding of the dual point of view brought considerable progress to the column generation theory and practice. It stimulated careful initializations, sophisticated solution techniques for the restricted master problem and the subproblem, as well as better overall performance.

Due to well known computational defects of Dantzig-Wolfe decomposition we cannot recommend its classical implementation as the formerly called complementary pricing scheme for large scale linear programs. However, structural dual inequalities, primal and dual stabilization strategies as well as non-linear implementations turn column generation into a very promising approach to decomposable linear programs.

Column generation is clearly a success story in large scale *integer* programming. The linear programming bound obtained from an extensive reformulation is often stronger, the tailing off effect can be lessened or circumvented at all, and the knowledge of the original compact formulation provides us with a strong guide for branching and cutting decisions in the search tree. Today we are in a position that generic integer programming column generation codes solve many large scale problems of "industrial difficulty," no standard commercial MIP solver could cope with. This is all the more true since non-linearities occurring in practical problems can be taken care of in the subproblem.

For very hard problems the *best* algorithmic choice may not be obvious. Having identified the computational bottleneck, one should invest in implementing more sophisticated ideas which we discuss in this paper. The good news is: There are plenty of them. In addition, all components greatly benefit from customization and tailoring. Problem adequate heuristics are most vital ingredients in an actual implementation. Thus, ample room for research and experiments is left, and hopefully some directions have been pointed to.

References

I. Adler and A. Ülkücü. On the number of iterations in Dantzig-Wolfe decomposition. In D.M. Himmelblau, editor, *Decomposition of Large Scale Problems*, pages 181–187. North-Holland, Amsterdam, 1973.

Y. Agarwal, K. Mathur, and H.M. Salkin. A set-partitioning-based exact algorithm for the vehicle routing problem. *Networks*, 19:731–749, 1989.

F. Alvelos and J.M. Valério de Carvalho. Solving multicommodity flow problems with branch-and-price. Technical report, Dept. Produção e Sistemas, Universidade do Minho, Portugal, 2000.

R. Anbil, J.J. Forrest, and W.R. Pulleyblank. Column generation and the airline crew pairing problem. In *Proceedings of the International Congress of Mathematicians Berlin*, volume Extra Volume ICM 1998 of *Doc. Math. J. DMV*, pages III 677–686, August 1998.

F. Barahona and R. Anbil. The volume algorithm: Producing primal solutions with a subgradient method. *Math. Programming*, 87(3):385–399, 2000.

F. Barahona and R. Anbil. On some difficult linear programs coming from set partitioning. *Discrete Appl. Math.*, 118(1–2):3–11, 2002.

F. Barahona and D. Jensen. Plant location with minimum inventory. *Math. Programming*, 83:101–111, 1998.

C. Barnhart, N.L. Boland, L.W. Clarke, E.L. Johnson, G.L. Nemhauser, and R.G. Shenoi. Flight string models for aircraft fleeting and routing. *Transportation Sci.*, 32(3):208–220, 1998a.

C. Barnhart, C.A. Hane, and P.H. Vance. Integer multicommodity flow problems. *Lecture Notes in Economics and Mathematical Systems*, 450:17–31, 1997.

C. Barnhart, C.A. Hane, and P.H. Vance. Using branch-and-price-and-cut to solve origin-destination integer multicommodity network flow problems. *Oper. Res.*, 48(3):318–326, 2000.

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. *Oper. Res.*, 46(3):316–329, 1998b.

C. Barnhart and R.R. Schneur. Air network design for express shipment service. *Oper. Res.*, 44(6): 852–863, 1996.

H. Ben Amor. *Stabilisation de l'Algorithme de Génération de Colonnes*. PhD thesis, École Polytechnique de Montréal, 2002.

J.F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numer. Math.*, 4:238–252, 1962.

R.E. Bixby. Solving real-world linear programs: A decade and more of progress. *Oper. Res.*, 50(1): 3–15, 2002.

R.E. Bixby, J.W. Gregory, I.J. Lustig, R.E. Marsten, and D.F. Shanno. Very large-scale linear programming: A case study in combining interior point and simplex methods. *Oper. Res.*, 40(5): 885–897, 1992.

R. Borndörfer, M. Grötschel, and A. Löbel. Duty scheduling in public transit. In W. Jäger and H.-J. Krebs, editors, *Mathematics—Key Technology for the Future*, pages 653–674. Springer-Verlag, Berlin, 2003.

J.-M. Bourjolly, G. Laporte, and H. Mercure. A combinatorial column generation algorithm for the maximum stable set problem. *Oper. Res. Lett.*, 20(1):21–29, 1997.

O. Briant, C. Lemaréchal, K. Monneris, N. Perrot, C. Tadonki, F. Vanderbeck, J.-P. Vial, and C. Beltram. Comparison of Kelley's cutting plane, bundle, and proximal analytic center algorithms for use in branch-and-price. Technical report, 2004. In preparation.

A. Caprara, M. Fischetti, and P. Toth. A heuristic method for the set covering problem. *Oper. Res.*, 47:730–743, 1999.

A. Caprara, M. Fischetti, and P. Toth. Algorithms for the set covering problem. *Ann. Oper. Res.*, 98: 353–371, 2000.

H.D. Chu, E. Gelman, and E.L. Johnson. Solving large scale crew scheduling problems. *European J. Oper. Res.*, 97:260–268, 1997.

V. Chvátal. *Linear Programming*. W.H. Freeman and Company, New York, 1983.

T.G. Crainic and J.-M. Rousseau. The column generation principle and the airline crew pairing problem. *Infor*, 25:136–151, 1987.

Y. Crama and A.G. Oerlemans. A column generation approach to job grouping for flexible manufacturing systems. *European J. Oper. Res.*, 78(1):58–80, 1994.

G.B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Oper. Res.*, 8:101–111, 1960.

G. Desaulniers, J. Desrosiers, Y. Dumas, M.M. Solomon, and F. Soumis. Daily aircraft routing and scheduling. *Management Sci.*, 43(6):841–855, 1997.

G. Desaulniers, J. Desrosiers, A. Erdmann, M.M. Solomon, and F. Soumis. The VRP with pickup and delivery. In P. Toth and D. Vigo, editors, *The Vehicle Routing Problem*, SIAM Monographs on Discrete Mathematics and Applications, chapter 9. SIAM, Philadelphia, 2001a.

G. Desaulniers, J. Desrosiers, I. Ioachim, M.M. Solomon, F. Soumis, and D. Villeneuve. A unified framework for deterministic time constrained vehicle routing and crew scheduling problems. In T.G. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, pages 57–93. Kluwer, Norwell, MA, 1998.

G. Desaulniers, J. Desrosiers, and M.M. Solomon. Accelerating strategies in column generation methods for vehicle routing and crew scheduling problems. In C.C. Ribeiro and P. Hansen, editors, *Essays and Surveys in Metaheuristics*, pages 309–324, Boston, 2001b. Kluwer.

M. Desrochers, J. Desrosiers, and M.M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Oper. Res.*, 40(2):342–354, 1992.

M. Desrochers and F. Soumis. A column generation approach to urban transit crew scheduling. *Transportation Sci.*, 23:1–13, 1989.

J. Desrosiers, Y. Dumas, M.M. Solomon, and F. Soumis. Time constrained routing and scheduling. In M.O. Ball, T.L. Magnanti, C.L. Monma, and G.L. Nemhauser, editors, *Network Routing*, volume 8 of *Handbooks in Operations Research and Management Science*, pages 35–139. North-Holland, Amsterdam, 1995.

J. Desrosiers, F. Soumis, and M. Desrochers. Routing with time windows by column generation. *Networks*, 14:545–565, 1984.

O. du Merle, D. Villeneuve, J. Desrosiers, and P. Hansen. Stabilized column generation. *Discrete Math.*, 194:229–237, 1999.

M. Eben-Chaime, C.A. Tovey, and J.C. Ammons. Circuit partitioning via set partitoning and column generation. *Oper. Res.*, 44(1):65–76, 1996.

I. Elhallaoui, D. Villeneuve, F. Soumis, and G. Desaulniers. Dynamic aggregation of set partitioning constraints in column generation. Les Cahiers du GERAD G-2003-45, HEC Montréal, 2003. Under revision for *Oper. Res.*

S. Elhedhli and J.-L. Goffin. The integration of an interior-point cutting-plane method within a branch-and-price algorithm. *Math. Programming*, 2004. In press.

A. Erdmann, A. Nolte, A. Noltemeier, and R. Schrader. Modeling and solving an airline schedule generation problem. *Ann. Oper. Res.*, 107:117–142, 2001.

A.A. Farley. A note on bounding a class of linear programming problems, including cutting stock problems. *Oper. Res.*, 38(5):922–923, 1990.

L.R. Ford and D.R. Fulkerson. A suggested computation for maximal multicommodity network flows. *Management Sci.*, 5:97–101, 1958.

J.J. Forrest and D. Goldfarb. Steepest-edge simplex algorithms for linear programming. *Math. Programming*, 57:341–374, 1992.

M. Gamache, F. Soumis, G. Marquis, and J. Desrosiers. A column generation apporach for large-scale aircrew rostering problems. *Oper. Res.*, 47(2):247–263, 1999.

M. Gamache, F. Soumis, D. Villeneuve, J. Desrosiers, and E. Gélinas. The preferential bidding system at Air Canada. *Transportation Sci.*, 32(3):246–255, 1998.

A.M. Geoffrion. Lagrangean relaxation for integer programming. *Math. Programming Stud.*, 2:82–114, 1974.

P.C. Gilmore and R.E. Gomory. A linear programming approach to the cutting-stock problem. *Oper. Res.*, 9:849–859, 1961.

P.C. Gilmore and R.E. Gomory. A linear programming approach to the cutting stock problem—Part II. *Oper. Res.*, 11:863–888, 1963.

J.-L. Goffin, A. Haurie, J.-Ph. Vial, and D.L. Zhu. Using central prices in the decomposition of linear programs. *European J. Oper. Res.*, 64:393–409, 1993.

J.-L. Goffin and J.-Ph. Vial. Convex nondifferentiable optimization: A survey focused on the analytic center cutting plane method. *Optim. Methods Softw.*, 17:805–867, 2003.

D. Goldfarb and J.K. Reid. A practicable steepest-edge simplex algorithm. *Math. Programming*, 12: 361–371, 1977.

M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer, Berlin, 1988.

A. Hadjar, O. Marcotte, and F. Soumis. A branch-and-cut algorithm for the multiple depot vehicle scheduling problem. Les Cahiers du GERAD G-2001-25, HEC Montréal, 2001.

P. Hansen, B. Jaumard, and M. Poggi de Aragão. Mixed-integer column generation algorithms and the probabilistic maximum satisfiability problem. *European J. Oper. Res.*, 108:671–683, 1998.

P.M.J. Harris. Pivot selection methods of the Devex LP code. *Math. Programming*, 5:1–28, 1973.

J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex analysis and minimization algorithms, part 2: Advanced theory and bundle methods*, volume 306 of *Grundlehren der mathematischen Wissenschaften*. Springer, Berlin, 1993.

J.K. Ho. Convergence behavior of decomposition algorithms for linear programs. *Oper. Res. Lett.*, 3 (2):91–94, 1984.

S. Holm and J. Tind. A unified approach for price directive decomposition procedures in integer programming. *Discrete Appl. Math.*, 20:205–219, 1988.

K. Holmberg and K. Jörnsten. A simple modification of Dantzig-Wolfe decomposition. *Optimization*, 34(2):129–145, 1995.

C.A.J. Hurkens, J.L. De Jong, and Z. Chen. A branch-and-price algorithm for solving the cutting strips problem. *Appl. Math., Ser. B (Engl. Ed.)*, 12(2):215–224, 1997.

I. Ioachim, J. Desrosiers, F. Soumis, and N. Bélanger. Fleet assignment and routing with schedule synchronization constraints. *European J. Oper. Res.*, 119(1):75–90, 1999.

I. Ioachim, S. Gélinas, F. Soumis, and J. Desrosiers. A dynamic programming algorithm for the shortest path problem with time windows and linear node costs. *Networks*, 31:193–204, 1998.

S. Irnich and D. Villeneuve. The shortest path problem with resource constraints and k-cycle elimination for $k \geq 3$. Les Cahiers du GERAD G-2003-55, HEC Montréal, 2003.

E.L. Johnson. Modelling and strong linear programs for mixed integer programming. In S.W. Wallace, editor, *Algorithms and Model Formulations in Mathematical Programming*, pages 1–43, Berlin, 1989. Springer.

E.L. Johnson, A. Mehrotra, and G.L. Nemhauser. Min-cut clustering. *Math. Programming*, 62:133–151, 1993.

B. Kallehauge, J. Larsen, and O.B.G. Madsen. Lagrangean duality applied on vehicle routing with time windows. Technical Report IMM-TR-2001-9, Informatics and Mathematical Modelling, Technical University of Denmark, Kgs. Lyngby, Denmark, 2001.

L.V. Kantorovich. Mathematical methods of organising and planning production. *Management Sci.*, 6:366–422, 1960. Translation from the Russian original, dated 1939.

J.E. Kelley Jr. The cutting-plane method for solving convex programs. *J. Soc. Ind. Appl. Math.*, 8(4): 703–712, 1961.

K. Kim and J.L. Nazareth. The decomposition principle and algorithms for linear programming. *Linear Algebra Appl.*, 152:119–133, 1991.

R. Kirkeby Martinson and J. Tind. An interior point method in Dantzig-Wolfe decomposition. *Comput. Oper. Res.*, 26(12):1195–1216, 1999.

P. Klein and N. Young. On the number of iterations for Dantzig-Wolfe optimization and packing-covering approximation algorithms. In *Proceedings of the 7th International IPCO Conference, Graz, Austria*, number 1610 in Lecture Notes in Computer Science, pages 320–327, Berlin, 1999. Springer.

N. Kohl, J. Desrosiers, O.B.G. Madsen, M.M. Solomon, and F. Soumis. 2-Path cuts for the vehicle routing problem with time windows. *Transportation Sci.*, 33(1):101–116, 1999.

N. Kohl and O.B.G. Madsen. An optimization algorithm for the vehicle routing problem with time windows. *Oper. Res.*, 45:395–406, 1997.

S.O. Krumke, J. Rambau, and L.M. Torres. Real-time dispatching of guided and unguided automobile service units with soft time windows. In R.H. Möhring and R. Raman, editors, *Proc. 10th Annual European Symposium on Algorithms, Rome, Italy*, volume 2461 of *Lecture Notes in Computer Science*, Berlin, 2002. Springer.

L.S. Lasdon. *Optimization Theory for Large Systems*. Macmillan, London, 1970.

A. Löbel. *Optimal Vehicle Scheduling in Public Transit*. PhD thesis, Technische Universität Berlin, 1997.

A. Löbel. Vehicle scheduling in public transit and Lagrangean pricing. *Management Sci.*, 44(12): 1637–1649, 1998.

M.E. Lübbecke and U.T. Zimmermann. Engine routing and scheduling at industrial in-plant railroads. *Transportation Sci.*, 37(2):183–197, 2003.

K. Madsen. An algorithm for minimax solution of overdetermined systems of non-linear equations. *J. Inst. Math. Appl.*, 16:321–328, 1975.

P. Mahey. A subgradient algorithm for accelerating the Dantzig-Wolfe decomposition method. In *Proceedings of the X. Symposium on Operations Research, Part I: Sections 1–5*, volume 53 of *Methods Oper. Res.*, pages 697–707, Königstein/Ts., 1986. Athenäum/Hain/Scriptor/Hanstein.

J.W. Mamer and R.D. McBride. A decomposition-based pricing procedure for large-scale linear programs – An application to the linear multicommodity flow problem. *Management Sci.*, 46(5): 693–709, 2000.

D.W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM J. Appl. Math.*, 11:431–441, 1963.

R.E. Marsten. The use of the boxstep method in discrete optimization. *Math. Programming Stud.*, 3: 127–144, 1975.

R.E. Marsten, W.W. Hogan, and J.W. Blankenship. The BOXSTEP method for large-scale optimization. *Oper. Res.*, 23:389–405, 1975.

K. Mehlhorn and M. Ziegelmann. Resource constrained shortest paths. In *Proceedings of the 8th Annual European Symposium on Algorithms, Saarbrücken, Germany*, number 1879 in Lecture Notes in Computer Science, pages 326–337. Springer, 2000.

A. Mehrotra and M.A. Trick. A column generation approach for graph coloring. *INFORMS J. Comput.*, 8(4):344–354, 1996.

M. Minoux. *Mathematical Programming*. John Wiley & Sons, Chichester, 1986.

M. Minoux. A class of combinatorial problems with polynomially solvable large scale set covering/partitioning relaxations. *RAIRO Rech. Opér.*, 21:105–136, 1987.

J.L. Nazareth. Numerical behaviour of LP algorithms based upon the decomposition principle. *Linear Algebra Appl.*, 57:181–189, 1984.

J.L. Nazareth. *Computer Solution of Linear Programs*. Oxford University Press, Oxford, 1987.

G.L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization*. John Wiley & Sons, Chichester, 1988.

K. Park, S. Kang, and S. Park. An integer programing approach to the bandwidth packing problem. *Management Sci.*, 42(9):1277–1291, 1996.

M. Poggi de Aragão and E. Uchoa. Integer program reformulation for robust branch-and-cut-and-price algorithms. In *Proceedings of the Conference Mathematical Program in Rio: A Conference in Honour of Nelson Maculan*, pages 56–61, 2003.

C.C. Ribeiro, M. Minoux, and M.C. Penna. An optimal column-generation-with-ranking algorithm for very large scale set partitioning problems in traffic assignment. *European J. Oper. Res.*, 41: 232–239, 1989.

C.C. Ribeiro and F. Soumis. A column generation approach to the multiple-depot vehicle scheduling problem. *Oper. Res.*, 42(1):41–52, 1994.

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.

J.K. Sankaran. Column generation applied to linear programs in course registration. *European J. Oper. Res.*, 87(2):328–342, 1995.

M.W.P. Savelsbergh. A branch-and-price algorithm for the generalized assignment problem. *Oper. Res.*, 45(6):831–841, 1997.

M.W.P. Savelsbergh and M. Sol. DRIVE: Dynamic routing of independent vehicles. *Oper. Res.*, 46 (4):474–490, 1998.

A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, Chichester, 1986.

E.L.F. Senne and L.A.N. Lorena. Stabilizing column generation using Lagrangean/surrogate relaxation: An application to p-median location problems. *European J. Oper. Res.*, 2002.

M. Sol. *Column Generation Techniques for Pickup and Delivery Problems*. PhD thesis, Eindhoven University of Technology, 1994.

F. Soumis. Decomposition and column generation. In M. Dell'Amico, F. Maffioli, and S. Martello, editors, *Annotated Bibliographies in Combinatorial Optimization*, pages 115–126. John Wiley & Sons, Chichester, 1997.

C. Swoveland. A note on column generation in Dantzig-Wolfe decomposition algorithms. *Math. Programming*, 6:365–370, 1974.

J.M. Valério de Carvalho. Exact solution of bin-packing problems using column generation and branch-and-bound. *Ann. Oper. Res.*, 86:629–659, 1999.

J.M. Valério de Carvalho. Using extra dual cuts to accelerate column generation. Technical report, Dept. Produção e Sistemas, Universidade do Minho, Portugal, 2000. To appear in *INFORMS J. Computing*.

J.M. Valério de Carvalho. LP models for bin-packing and cutting stock problems. *European J. Oper. Res.*, 141(2):253–273, 2002a.

J.M. Valério de Carvalho. A note on branch-and-price algorithms for the one-dimensional cutting stock problem. *Comput. Optim. Appl.*, 21(3):339–340, 2002b.

T.J. Van Roy. Cross decomposition for mixed integer programming. *Math. Programming*, 25:46–63, 1983.

P.H. Vance. Branch-and-price algorithms for the one-dimensional cutting stock problem. *Comput. Optim. Appl.*, 9(3):211–228, 1998.

P.H. Vance, C. Barnhart, E.L. Johnson, and G.L. Nemhauser. Solving binary cutting stock problems by column generation and branch-and-bound. *Comput. Optim. Appl.*, 3(2):111–130, 1994.

F. Vanderbeck. *Decomposition and Column Generation for Integer Programs*. PhD thesis, Université catholique de Louvain, 1994.

F. Vanderbeck. Computational study of a column generation algorithm for bin packing and cutting stock problems. *Math. Programming*, 86(3):565–594, 1999.

F. Vanderbeck. On Dantzig-Wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm. *Oper. Res.*, 48(1):111–128, 2000.

F. Vanderbeck. Implementing mixed integer column generation. In G. Desaulniers, J. Desrosiers, and M.M. Solomon, editors, *Column Generation*. 2004. In preparation.

F. Vanderbeck and L.A. Wolsey. An exact algorithm for IP column generation. *Oper. Res. Lett.*, 19: 151–159, 1996.

D. Villeneuve. *Logiciel de Génération de Colonnes*. PhD thesis, École Polytechnique de Montréal, 1999.

D. Villeneuve, J. Desrosiers, M.E. Lübbecke, and F. Soumis. On compact formulations for integer programs solved by column generation. Les Cahiers du GERAD G-2003-06, HEC Montréal, 2003. To appear in *Ann. Oper. Res.*

W.E. Walker. A method for obtaining the optimal dual solution to a linear program using the Dantzig-Wolfe decomposition. *Oper. Res.*, 17:368–370, 1969.

D. Wedelin. An algorithm for large scale 0-1 integer programming with application to airline crew scheduling. *Ann. Oper. Res.*, 57:283–301, 1995.

P. Wentges. Weighted Dantzig-Wolfe decomposition of linear mixed-integer programming. *Int. Trans. Opl. Res.*, 4(2):151–162, 1997.

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

L.A. Wolsey. *Integer Programming*. John Wiley & Sons, Chichester, 1998.

Reports from the group

"Combinatorial Optimization and Graph Algorithms"

of the Department of Mathematics, TU Berlin

2004/08 *Marco E. Lübbecke and Jacques Desrosiers:* Selected Topics in Column Generation

2003/49 *Christian Liebchen and Rolf H. Möhring:* Information on MIPLIB's timetab-instances

2003/48 *Jacques Desrosiers and Marco E. Lübbecke:* A Primer in Column Generation

2003/47 *Thomas Erlebach, Vanessa Kääb, and Rolf H. Möhring:* Scheduling AND/OR-Networks on Identical Parallel Machines

2003/43 *Michael R. Bussieck, Thomas Lindner, and Marco E. Lübbecke:* A Fast Algorithm for Near Cost Optimal Line Plans

2003/42 *Marco E. Lübbecke:* Dual Variable Based Fathoming in Dynamic Programs for Column Generation

2003/37 *Sándor P. Fekete, Marco E. Lübbecke, and Henk Meijer:* Minimizing the Stabbing Number of Matchings, Trees, and Triangulations

2003/25 *Daniel Villeneuve, Jacques Desrosiers, Marco E. Lübbecke, and François Soumis:* On Compact Formulations for Integer Programs Solved by Column Generation

2003/24 *Alex Hall, Katharina Langkau, and Martin Skutella:* An FPTAS for Quickest Multicommodity Flows with Inflow-Dependent Transit Times

2003/23 *Sven O. Krumke, Nicole Megow, and Tjark Vredeveld:* How to Whack Moles

2003/22 *Nicole Megow and Andreas S. Schulz:* Scheduling to Minimize Average Completion Time Revisited: Deterministic On-Line Algorithms

2003/16 *Christian Liebchen:* Symmetry for Periodic Railway Timetables

2003/12 *Christian Liebchen:* Finding Short Integral Cycle Bases for Cyclic Timetabling

762/2002 *Ekkehard Köhler and Katharina Langkau and Martin Skutella:* Time-Expanded Graphs for Flow-Dependent Transit Times

761/2002 *Christian Liebchen and Leon Peeters:* On Cyclic Timetabling and Cycles in Graphs

752/2002 *Ekkehard Köhler and Rolf H. Möhring and Martin Skutella:* Traffic Networks and Flows Over Time

739/2002 *Georg Baier and Ekkehard Köhler and Martin Skutella:* On the k-splittable Flow Problem

736/2002 *Christian Liebchen and Rolf H. Möhring:* A Case Study in Periodic Timetabling

Reports may be requested from:

Sekretariat MA 6–1
Fakultät II – Institut für Mathematik
TU Berlin
Straße des 17. Juni 136
D-10623 Berlin – Germany

e-mail: klink@math.TU-Berlin.DE

Reports are also available in various formats from

`http://www.math.tu-berlin.de/coga/publications/techreports/`

and via anonymous ftp as

`ftp://ftp.math.tu-berlin.de/pub/Preprints/combi/Report-`*number*`-`*year*`.ps`