544 LECTURE NOTES IN ECONOMICS AND MATHEMATICAL SYSTEMS

Bernhard Fleischmann
Andreas Klose
Editors

Distribution Logistics

Advanced Solutions to Practical Problems

Springer

Lecture Notes in Economics and Mathematical Systems 544

B. Fleischmann
A. Klose (Eds.)

Distribution Logistics

Advanced Solutions to Practical Problems

Springer

Editors

Prof. Dr. Bernhard Fleischmann
University of Augsburg
Department of Production and Logistics
Universitätsstraße 16
86159 Augsburg, Germany

Priv.-Doz. Dr. Andreas Klose
University of St. Gallen
Bodanstraße 6
9000 St.Gallen, Switzerland

Library of Congress Control Number: 2004107326

ISSN 0075-8442
ISBN 978-3-540-22100-5 ISBN 978-3-642-17020-1 (eBook)
DOI 10.1007/978-3-642-17020-1

springeronline.com

Originally published by Springer-Verlag Berlin Heidelberg New York in 2005

Typesetting: Camera ready by author
Cover design: *Erich Kirchner*, Heidelberg

Printed on acid-free paper 42/3130Di 5 4 3 2 1 0

Editorial

This book is the fourth volume on applied research in Distribution Logistics that results from the work of a group of mainly European researchers meeting regularly at the "IWDL" (International Workshop on Distribution Logistics). The book contains a selection of research papers, some of which have been presented at the IWDL 7 in Grainau, Germany, in October 2002. It continues the tradition of the previous volumes, all appeared as Springer Lecture Notes in Economics and Mathematical Systems (no. 460, 1998; no. 480, 1999; no. 519, 2002), which have found a favourable acceptance by the logistics community. Recently, the second volume has appeared in a Chinese translation.

Distribution Logistics make up a major part of Supply Chain Management and concern all flows of goods and information between the production sites and the customers. Various trends contribute to a considerable increase in complexity of distribution systems, such as the globalization of the business of most manufacturers and the increasing dynamics of the customer demands. It is therefore not surprising, that the interest in "advanced" planning methods, based on quantitative optimization, increases in practice, in particular for the design of distribution systems and the control of the various interrelated transportation and warehousing processes. This development is favoured by the advances in the information and communication technology, which has enabled a nearly unlimited availability of data at any place at any time. On this background, a new kind of supply chain planning software for manufacturing industries, the "Advanced Planning Systems", have emerged. Their usefulness for Distribution Logistics however is restricted, as the manufacturers have outsourced major tasks of it to logistics service providers (LSP), who combine the distribution processes for several supply chains. Therefore, Distribution Logistics is mainly the responsibility of the LSP. But Advanced Planning Systems for an LSP do not exist yet. The papers in this book deal with some of these developments and challenges.

This book, like its predecessors, includes papers on a recent branch of Logistics, Reverse Logistics. The reason for this is threefold: First, Reverse Logistics are closely related to Distribution Logistics, as the reverse flows link again customers and production sites. Second, the practical importance of Reverse Logistics is fast-growing, and third, a number of members in the IWDL group have become experts in this field.

The 13 papers of this volume have been arranged in four chapters, the first three on the traditional subjects: design of distribution systems, tactical and operational vehicle routing and warehousing operations. The last chapter is dedicated to Reverse Logistics.

Chapter 1 addresses theory and application of the distribution network design and distribution concepts for E-commerce. *Görtz and Klose* consider the general Capacitated Facility Location Problem (CFLP), which is a basic model for various network design problems. They review the existing solution methods for this hard combinatorial problem and suggest a new type of lower bounds based on column generation. The lower bound is used in a branch & bound algorithm and tested in a computational experiment. *Bauer* stresses the need for considering cost and capacities of warehouse processes when designing a distribution network. She introduces a "Modular Node Model" for that purpose and shows the impact of the warehouse processes on the resulting network. *Bloemhof, Smeets and van Nunen* report an interesting practical case of supply chain design for the Dutch pig husbandry. The network consisting of farmers, slaughter houses, wholesalers and retailers is optimized using a mixed integer programming model. *Daduna and Lenz* deal with the often neglected aspects of physical distribution caused by "Online-Shopping". They investigate its impact on the traffic for the "last mile", both for the commercial freight deliveries and the private shopping trips.

Chapter 2 deals with different cases of vehicle routing in various practical environments. *Mansini, Speranza and Angelelli* address on-line routing problems occurring in the multi-depot network of an LSP. They analyse the particular planning situation and present a model and new algorithms. *Archetti and Speranza* report on the solution of a case of waste collection in the county of Brescia, Italy. It contains a pickup and delivery problem for the transport of empty and full containers, where various side constraints have to be considered. *Bieding* deals with planning time-critical standard-tours, as they occur in the daily distribution of newspapers. A focus is on the improvement of data on travel times using modern measuring systems such as GPS and RFID. *Schönberger and Kopfer* consider a general pickup and delivery problem, where a distinction is made between own vehicles, which are to be routed, and subcontracted vehicles, which are paid per load.

The papers in **Chapter 3**, again, concern routing problems, but for order-picking within a warehouse. *Le Anh and de Koster* investigate the performance of various dispatching rules for the on-line control of the vehicles. They report a simulation study in two real-life environments. *Le Duc and de Koster* present models for estimating the length of an order-picking tour, depending on the storage strategy. These approximations allow the optimization of the boundaries between the warehouse zones and of the warehouse layout.

Finally, **Chapter 4** consists of contributions to the field of Reverse Logistics. For the case of reusing components of products after their end of live, *Geyer and Van Wassenhove* study the effect of time constraints, such as a limited component durability or a finite product life cycle. In the same context, *Krikke, van Nunen, Zuidwijk and Kuik* consider the management of return flows, using information on the "installed base" of product placements in the market. *De Brito, Dekker and Flapper* provide a comprehensive review of case studies in Reverse Logistics, classifying the content of more than 60 publications.

The editors are indebted to all authors for their valuable contributions and to the referees whose work was essential to ensure a high quality level of this book.

Bernhard Fleischmann, University of Augsburg, Germany
Andreas Klose, University of Zurich, Switzerland

July 2004

Contents

Chapter 1: Networks

Chapter 2: Vehicle Routing

Chapter 3: Warehousing

Chapter 4: Reverse Logistics

Appendix

Chapter 1

Networks

An Exact Column Generation Approach to the Capacitated Facility Location Problem

Andreas Klose[1] and Simon Görtz[2]

[1] University of St. Gallen and Institute for Operations Research, University of Zurich, Switzerland
[2] Faculty of Economics and Social Sciences, University of Wuppertal, Germany

Abstract The Capacitated Facility Location Problem (CFLP) is a well-known combinatorial optimization problem with applications in distribution and production planning. It consists in selecting plant sites from a finite set of potential sites and in allocating customer demands in such a way as to minimize operating and transportation costs. A variety of lower bounds based on Lagrangean relaxation and subgradient optimization has been proposed for this problem. However, in order to solve large or difficult problem instances information about a primal (fractional) solution is important. Therefore, we employ column generation in order to solve a corresponding master problem exactly. The algorithm uses different strategies for stabilizing the column generation process. Furthermore, the column generation method is employed within a branch-and-price procedure for computing optimal solutions to the CFLP. Computational results are reported for a set of larger and difficult problem instances. The results are compared with computational results obtained from a branch-and-bound procedure based on Lagrangean relaxation and subgradient optimization and a branch-and-bound method that uses the LP relaxation and polyhedral cuts.

1 Introduction

The Capacitated Facility Location Problem (CFLP) is a well-known and well studied combinatorial optimization problem. It consists in deciding which plants to open from a given set of potential plant locations and how to assign customers to those plants. The objective is to minimize total fixed and shipping costs. Constraints are that each customer's demand must be satisfied and that each plant cannot supply more than its capacity if it is open. Applications of the CFLP include location and distribution planning, lot sizing in production planning (Pochet and Wolsey, 1988), and telecommunication network design (Kochmann and McCallum, 1981; Mirzaian, 1985; Boffey, 1989; Chardaire, 1999).

Numerous heuristic and exact algorithms for the CFLP have been proposed in the literature. Classical heuristics apply ADD, DROP and interchange moves in conjunction with dominance criteria and approximations of the cost change caused by a move (Kuehn and Hamburger, 1963; Khumawala, 1974; Jacobsen, 1983; Domschke and Drexl, 1985; Mateus and Bornstein, 1991). Tabu search procedures have been developed for related problems like the p-median problem (Rolland et al., 1996) and the CFLP with single sourcing (Delmaire et al., 1999). Based on a rounding and filtering technique proposed by Lin and Vitter (1992), Shmoys et al. (1997) derive new approximation results for the metric CFLP. Korupolu et al. (1998) show that simple local search heuristics also give polynomial constant factor approximation algorithms for this problem. Magnanti and Wong (1981) and Wentges (1996) apply Benders' decomposition and show how to derive strong Benders' cuts for the CFLP. Polyhedral results for the CFLP have been obtained by Leung and Magnanti (1989), Aardal et al. (1995) and Aardal (1998). Aardal (1998) uses these results in a branch-and-cut algorithm for the CFLP. A variety of solution approaches for the CFLP, however, use Lagrangean relaxation. Lagrangean relaxation of the demand constraints with or without addition of an aggregate capacity constraint or another surrogate constraint is considered in Geoffrion and McBride (1978), Nauss (1978), Christofides and Beasley (1983), Shetty (1990), and Ryu and Guignard (1992). Beasley (1988) relaxes both the demand and capacity constraints; the resulting relaxation is, therefore, no stronger than the LP relaxation. Van Roy (1986) employs Lagrangean relaxation of the capacity constraints and cross decomposition in order to solve the resulting Lagrangean dual.

With the exception of Van Roy's (1986) cross decomposition algorithm, Lagrangean relaxation approaches for the CFLP generally use subgradient optimization in order to obtain an approximate solution to the Lagrangean dual. However, for solving larger and/or more difficult instances of the CFLP the knowledge of an exact solution of the corresponding master problem can be advantageous. Firstly, this gives an improved lower bound and, secondly, the knowledge of a fractional optimal solution of the primal master problem can be exploited to devise (better) branching decisions in the framework of a branch-and-price algorithm. In this paper, column generation is, therefore, employed in order to obtain exact solutions to the master problem. The approach is based on relaxing the demand constraints in a Lagrangean manner, and a hybrid mixture of subgradient optimization and a "weighted" decomposition method is applied for solving the master problem. Furthermore, the column generation procedure is embedded in a branch-and-price algorithm for computing optimal solutions to the CFLP.

This paper is organized as follows. The next section summarizes important Lagrangean bounds for the CFLP. Section 3 describes the column generation method, that is the master problem and the pricing subproblem, the employed method for stabilizing the decomposition as well as the branch-and-price procedure. Section 4 gives computational results which have been obtained for a set of problem instances with up to 200 potential plant locations and 500 customers. Finally, the findings are summarized in section 5.

2 Lagrangean Relaxation

Mathematically, the CFLP can be stated as the following linear mixed-integer program:

$$
\begin{aligned}
Z = \min \ & \sum_{k\in K}\sum_{j\in J} c_{kj}x_{kj} + \sum_{j\in J} f_j y_j && && \text{(O)}\\
\text{s.t.} \ & \sum_{j\in J} x_{kj} = 1\,, && \forall\, k \in K && \text{(D)}\\
& \sum_{k\in K} d_k x_{kj} \le s_j y_j\,, && \forall\, j \in J && \text{(C)}\\
& \sum_{j\in J} s_j y_j \ge d(K) && && \text{(T)}\\
& x_{kj} - y_j \le 0, && \forall\, k \in K\,,\ \forall\, j \in J && \text{(B)}\\
& 0 \le x_{kj} \le 1\,,\ 0 \le y_j \le 1\,, && \forall\, k \in K\,,\ \forall\, j \in J && \text{(N)}\\
& y_j \in \{0,1\}\,, && \forall\, j \in J\,, && \text{(I)}
\end{aligned}
$$

where K is the set of customers and J the set of potential plant locations; c_{kj} is the cost of supplying all of customer k's demand d_k from location j, f_j is the fixed cost of operating facility j and s_j its capacity if it is open; the binary variable y_j is equal to 1 if facility j is open and 0 otherwise; finally, x_{kj} denotes the fraction of customer k's demand met from facility j. The constraints (D) are the demand constraints and constraints (C) are the capacity constraints. The aggregate capacity constraint (T) and the implied bounds (B) are superfluous; they are, however, usually added in order to sharpen the bound if Lagrangean relaxation of constraints (C) and/or (D) is applied. Without loss of generality it is assumed that $c_{kj} \ge 0 \ \forall\ k, j$, $f_j \ge 0 \,\forall\, j$, $s_j > 0 \,\forall\, j$, $d_k \ge 0 \,\forall\, k$, and $\sum_{j\in J} s_j > d(K) = \sum_{k\in K} d_k$.

Lagrangean relaxation approaches for the CFLP relax at least one the constraint sets (D) or (C); otherwise the Lagrangean subproblem has the same complexity as the CFLP itself. Cornuejols et al. (1991) examine all possible ways of applying Lagrangean relaxations or Lagrangean decompositions to the problem consisting of constraints (D), (C), (T), (B), (N), and (I). Following their notation, let

- Z_R^S denote the resulting lower bound if constraint set S is ignored and constraints R are relaxed in a Lagrangean fashion, and let
- Z_{R_1/R_2} denote the bound which results if Lagrangean decomposition is applied in such a way that constraints R_2 do not appear in the first subproblem and constraints R_1 do not appear in the second subproblem.

Regarding Lagrangean relaxation, Cornuejols et al. (1991, Theorem 1) show that

$$Z^{BI} \leq Z^{I} \leq Z_C^T \leq Z_C \leq Z\,, \tag{1}$$
$$Z^{I} \leq Z_D \leq Z_C\,, \tag{2}$$
$$Z^{BI} \leq Z_C^B \leq Z_D\,. \tag{3}$$

Furthermore, they provide instances showing that all the inequalities above can be strict. The subproblem corresponding to Z_D can be converted to a knapsack problem and is solvable in pseudo-polynomial time. Therefore, bounds inferior to Z_D seem not to be interesting. Furthermore, as computational experiments show, Z_C^T is usually not stronger than Z_D. This leaves Z_D and Z_C as candidate bounds.

With respect to Lagrangean decomposition, Cornuejols et al. (1991, Theorem 2) proof that

$$Z_{C/D} = Z_{C/DB} = Z_{C/DT} = Z_{C/DBT} = Z_C\,, \tag{4}$$
$$\max\{Z_C^T, Z_D\} \leq Z_{D/TC} \leq Z_C\,, \tag{5}$$
$$Z_{D/BC} = Z_{D/TBC} = Z_{TD/BC} = Z_D\,. \tag{6}$$

Since Lagrangean decomposition requires to solve two subproblems in each iteration and to optimize a large number of multipliers, Lagrangean decomposition should give a bound which is at least as strong as Z_D. The only remaining interesting bound is, therefore, $Z_{D/TC}$. As shown by Chen and Guignard (1998), the bound $Z_{D/TC}$ is also obtainable by means of a technique called Lagrangean substitution, which substitutes the copy constraints $x = x'$ by $\sum_k d_k x_{kj} = \sum_k d_k x'_{kj}$. Compared to the Lagrangean decomposition, this reduces the number of dual variables from $|K| \cdot |J| + |J|$ to $2|J|$.

In summary, interesting Lagrangean bounds for the CFLP are Z_D, Z_C, and $Z_{D/TC}$. Compared to Z_C, the computation of the bound $Z_{D/TC}$ requires to optimize an increased number of dual variables. Furthermore, one of the subproblems corresponding to $Z_{D/TC}$ is an Uncapacitated Facility Location Problem (UFLP) while the subproblem corresponding to Z_C is an Aggregate Capacitated Plant Location Problem (APLP). Since the bound $Z_{D/TC}$ is no stronger than Z_C and since an APLP is often not much harder to solve than an UFLP, the bound $Z_{D/TC}$ seems not to be more attractive than Z_C. Compared to Z_D, the computation of the bound Z_C requires to repeatedly solve a strictly NP-hard subproblem, while the subproblem corresponding to Z_D is decomposable and solvable in pseudo-polynomial time. The column generation and branch-and-price procedure described in the following sections is, therefore, based on the Lagrangean relaxation of the demand constraints (D).

3 Column Generation Method

In the following subsections, the employed column generation scheme is described in detail. As shown in Section 3.1 the approach is based on relaxing the demand constraints (D) in a Lagrangean manner. Due to convergence problems of standard Dantzig-Wolfe decomposition, methods for improving convergence are required in order to solve the master problem efficiently. These methods are explained in Section 3.2. Finally, the branch-and-price algorithm for computing optimal solutions is outlined in Section 3.3.

3.1 Master Problem and Pricing Subproblem

Consider the mathematical formulation of the CFLP given by (O), (D), (C), (T), (B), (N) and (I). Dualizing constraints (D) with multipliers η_k, $k \in K$, gives the Lagrangean subproblem

$$Z_D(\eta) = \sum_{k \in K} \eta_k + \min_{x,y} \sum_{k \in K} \sum_{j \in J} (c_{kj} - \eta_k) x_{kj} + \sum_{j \in J} f_j y_j \qquad (7)$$
$$\text{s.t.: (C), (N), (I), (B), (T)}\,.$$

It is easy to show, that optimal Lagrangean multipliers η^{opt} can be found in the interval $[\eta^{\min}, \eta^{\max}]$, where

$$\eta_k^{\min} = \min\{c_{kj} : j \in J \setminus \{j(k)\}\} \geq 0\,, \; c_{kj(k)} = \min_{j \in J} c_{kj}\,, \eta_k^{\max} = \max_{j \in J} c_{kj}\,.$$

Furthermore, it is well-known that (7) can be reduced to a knapsack problem (see, e. ,g., Sridharan (1993)). To this end, define

$$v_j = \max_x \Big\{ \sum_{k \in K} (\eta_k - c_{kj}) x_{kj} : \sum_{k \in K} d_k x_{kj} \leq s_j\,,\; 0 \leq x_{kj} \leq 1 \;\forall\, k \in K \Big\} \qquad (8)$$

in order to obtain

$$Z_D(\eta) = \eta_0 + \sum_{k \in K} \eta_k\,,$$

where

$$\eta_0 = \min_y \Big\{ \sum_{j \in J} (f_j - v_j) y_j : \sum_{j \in J} s_j y_j \geq d(K)\,,\; y_j \in \{0,1\} \;\forall\, j \in J \Big\}\,. \qquad (9)$$

In (8) and (9) constraints (B) are taken into account by setting $x_{ij} = 0$ if $y_j = 0$ holds in an optimal solution to (9).

The Lagrangean dual of (7) is to maximize the Lagrangean function $Z_D(\eta)$ over the set $[\eta^{\min}, \eta^{\max}]$. Let $\{y^t : t \in \mathcal{T}^y\}$ denote the finite set of all feasible solutions to the knapsack problem (9) and let $\{x_j^t : t \in \mathcal{T}_j^x\}$ denote the vertices of the set of feasible solutions to (8). For $t \in \mathcal{T}^y$ and $t \in \mathcal{T}_j^x$, define

$$F_t = \sum_{j \in J} f_j y_j^t \quad \text{and} \quad C_{tj} = \sum_{k \in K} c_{kj} x_{kj}^t\,.$$

Using these definitions and (8) as well as (9), the Lagrangean dual can be written as the linear program:

$$Z_D = \max \eta_0 + \sum_{k\in K} \eta_k \tag{10}$$

$$\text{s.t. } \eta_0 + \sum_{j\in J} y_j^t v_j \le F_t\,, \qquad \forall\, t \in \mathcal{T}^y \tag{11}$$

$$\sum_{k\in K} x_{kj}^t \eta_k - v_j \le C_{tj}\,, \quad \forall\, j \in J\,,\ \forall\, t \in \mathcal{T}_j^x \tag{12}$$

$$v_j \ge 0\,, \qquad \forall\, j \in J \tag{13}$$

$$\eta_k^{\min} \le \eta_k \le \eta_k^{\max}\,, \qquad \forall\, k \in K \tag{14}$$

$$\eta_0 \in \mathbb{R}\,. \tag{15}$$

Taking the dual of the above "dual master program" one obtains the so-called "primal master program", which is given by:

$$Z_D = \min \sum_{t\in\mathcal{T}^y} F_t \alpha_t + \sum_{j\in J}\sum_{t\in\mathcal{T}_j^x} C_{tj}\beta_{tj} + \sum_{k\in K}\left(\eta_k^{\max}\overline{p}_k - \eta_k^{\min}\underline{p}_k\right) \tag{16}$$

$$\text{s.t. } \sum_{t\in\mathcal{T}^y} \alpha_t = 1\,, \tag{17}$$

$$\sum_{t\in\mathcal{T}^y} y_j^t \alpha_t - \sum_{t\in\mathcal{T}_j^x} \beta_{tj} \ge 0\,, \qquad \forall\, j \in J \tag{18}$$

$$\sum_{j\in J}\sum_{t\in\mathcal{T}_j^x} x_{kj}^t \beta_{tj} + \overline{p}_k - \underline{p}_k \ge 1\,, \quad \forall\, k \in K \tag{19}$$

$$\alpha_t \ge 0\,, \qquad \forall\, t \in \mathcal{T}^y \tag{20}$$

$$\beta_{tj} \ge 0\,, \qquad \forall\, j \in J\,,\ \forall\, t \in \mathcal{T}_j^x \tag{21}$$

$$\overline{p}_k, \underline{p}_k \ge 0\,, \qquad \forall\, k \in K\,, \tag{22}$$

where the variables α_t, β_{tj} as well as $\underline{p}_k$ and $\overline{p}_k$ are the dual variables of the constraints (11), (12) and (14), respectively.

If the constraints (20) and (21) are replaced by $\alpha_t \in \{0,1\}$ $\forall t \in \mathcal{T}^y$ and $\beta_{tj} \in \{0,1\}$ $\forall j \in J$ and $\forall t \in \mathcal{T}_j^x$, an equivalent formulation of the CFLP is obtained: The problem consists in selecting exactly one plant subset $S^t = \{j \in J : y_j^t = 1\}$, $t \in \mathcal{T}^y$, with sufficient capacity and in choosing feasible flows x_j^t, $t \in \mathcal{T}_j^x$, from plants to customers in such a way that total costs are minimized, each customer's demand is met (constraints (19)) and that there are no flows from closed plants to customers (constraints (18)). The primal master program (16)–(22) is the linear relaxation of this equivalent integer reformulation.

The primal master program (16)–(22) has to be solved by means of column generation. To this end consider the master problem *restricted* to known column subsets $\overline{\mathcal{T}}^y \subset \mathcal{T}^y$ and $\overline{\mathcal{T}}_j^x \subset \mathcal{T}_j^x$ for all $j \in J$. Furthermore let $\overline{\eta}_0$, $\overline{v}$, and $\overline{\eta}$ denote an optimal dual solution of the restricted master problem. New columns x_j^h and y^h price out, if

$$\overline{v}_j < \sum_{k\in K}(\overline{\eta}_k - c_{kj})x_{kj}^h \quad \Rightarrow \quad \overline{v}_j < v_j(\overline{\eta}) \stackrel{\text{def}}{=} \max\Big\{\sum_{k\in K}(\overline{\eta}_k - c_{kj})x_{kj}^t \,:\, t\in \mathcal{T}_j^x\Big\}$$

and

$$\overline{\eta}_0 > \sum_{j\in J}(f_j - \overline{v}_j)y_j^h \quad \Rightarrow \quad \overline{\eta}_0 > \min\Big\{\sum_{j\in J}(f_j - \overline{v}_j)y_j^t \,:\, t\in \mathcal{T}^y\Big\}.$$

Since $v_j(\overline{\eta}) \geq \overline{v}_j \ \forall\, j \in J$, using $v_j(\overline{\eta})$ instead of $\overline{v}$ in order to price out columns y^h is generally preferable; it leads to an earlier detection of required columns y^h.

For large instances of the CFLP even the restricted master problem is usually quite large, and the effort required for iteratively (re-)optimizing the restricted master problem can be tremendous. In addition it is well known that the conventional Danztig-Wolfe decomposition approach suffers from bad convergence behavior (see e. g. Lemaréchal (1989)). Methods for stabilizing the decomposition and reducing the computational effort required are, therefore, essential in order to solve the master problem (16)–(22) efficiently.

3.2 Stabilizing the Column Generation

When Lagrangean relaxation is applied to a general mixed-integer programming problem $\min\{cx \,:\, Ax = b,\ x \in X\}$, the Lagrangean dual is to maximize the piecewise linear and concave function

$$\nu(u) = ub + \min\{(c - uA)x \,:\, x \in X\} = ub + \min\{(c - uA)x^t \,:\, t \in \mathcal{T}\}\,, \quad (23)$$

where $\{x^t \,:\, t \in \mathcal{T}\}$ is the set of all vertices of the convex hull of X (for simplicity it is assumed that X is nonempty and bounded). For a given known subset $\overline{\mathcal{T}} \subset \mathcal{T}$ of columns, the function

$$\overline{\nu}(u) = ub + \min\{(c - uA)x^t \,:\, t \in \overline{\mathcal{T}}\}$$

is an outer approximation of $\nu(u)$. The restricted dual and primal master problem is then given by

$$\overline{\nu}(u^h) \stackrel{\text{def}}{=} \max_u \ \overline{\nu}(u) = \max_{u_0,u}\{u_0 + ub \,:\, u_0 + uAx^t \leq cx^t\ \forall\, t \in \overline{\mathcal{T}}\} \qquad (24)$$

$$= \min_{\alpha\geq 0}\Big\{\sum_{t\in\overline{\mathcal{T}}}(cx^t)\alpha_t \,:\, \sum_{t\in\overline{\mathcal{T}}}(Ax^t)\alpha_t = b\,,\ \sum_{t\in\overline{\mathcal{T}}}\alpha_t = 1\Big\}\,, \qquad (25)$$

where $u_0 \stackrel{\text{def}}{=} \min\{(c - uA)x^t \,:\, t \in \overline{\mathcal{T}}\}$ and α_t is the dual variable corresponding to the dual cut $u_0 + uAx^t \leq c^t$ for $t \in \overline{\mathcal{T}}$. At each iteration of the standard column generation algorithm (Kelley, 1960; Dantzig and Wolfe, 1960), the restricted master problem (24) is solved and an optimal solution x^h of the Lagrangean/pricing subproblem (23) for fixed $u = u^h$ is determined. The outer approximation $\overline{\nu}(u)$ is then refined by adding h to the set $\overline{\mathcal{T}}$ of columns. In order to improve the convergence behavior of this approach, a number of different methods have been proposed in the literature.

In order to avoid large oscillations of the dual variables u, Marsten et al. (1975) put a box centered at the current point, say u^{h-1}, around the dual variables u and solve

$$\overline{\nu}_\delta(\overline{u}^h) = \max_u \{\overline{\nu}(u) \,:\, u^{h-1} - \delta \leq u \leq u^{h-1} + \delta\}\,.$$

The next iterate u^h is then found by performing a line search into the direction $(\overline{u}^h - u^{h-1})$.

Du Merle et al. (1999) generalize the boxstep method of Marsten et al. They allow the next iterate to lie outside the current box, but penalize violations of the "box constraints". For these purposes they use the perturbed (restricted) dual master program

$$\begin{aligned} \max\ & u_0 + ub - w^+\pi^+ - w^-\pi^- \\ \text{s.t. } & u_0 + uAx^t \leq cx^t\,, \qquad\qquad \forall\, t \in \overline{\mathcal{T}} \\ & \delta^- - w^- \leq u \leq \delta^+ + w^+\,, \\ & w^-, w^+ \geq 0\,. \end{aligned} \tag{26}$$

Du Merle et al. propose different strategies to initialize the parameters π^+, π^-, δ^+, δ^- and to adapt them in case that an optimal solution u^h of (26) improves (not improves) the best dual solution found so far or in case that u^h is dual feasible.

As Neame et al. (1998) show, the method of du Merle et al. can be viewed as a penalty method which subtracts the penalty function

$$P_1(u) = \sum_i \max\{0,\, \pi_i^+(u_i - \delta_i^+),\, \pi_i^-(\delta_i^- - u_i)\}$$

from the outer approximation $\overline{\nu}(u)$ in order to determine the next point. The method of du Merle et al. is, therefore, closely related to bundle methods (Lemaréchal, 1989; Carraresi et al., 1995; Frangioni and Gallo, 1999) which use a quadratic penalty function

$$P_2(u) = \nu(u^{h-1}) + \frac{1}{2\tau}\left\|u - u^{h-1}\right\|^2,$$

where $\tau > 0$ is a "trust" parameter and u^{h-1} the current point. In this case, the master program or direction finding problem is a quadratic program.

Let $\nu(u^b) = \max\{\nu(u^t) \ : \ t \in \overline{\mathcal{T}}\}$ denote the best lower bound found so far. Optimal dual variables u are then located in the set

$$L = \big\{(u_0,\, u) \ : u_0 + ub - w_0 = \nu(u^b)\,,\ u_0 + u(Ax^t) + w_t = (cx^t)\ \forall\, t \in \overline{\mathcal{T}}\,,\\ w_0 \geq 0\,,\ w_t \geq 0\ \forall\, t \in \overline{\mathcal{T}}\big\}\,.$$

Select any point $(u_0^h,\, u^h) \in L$ with $w_0^h = u_0^h + u^h b - \nu(u^b) > 0$. If $(u_0^h,\, u^h)$ is dual feasible, that is $w_t^h = cx^t - u_0^h - u^h(Ax^t) \geq 0\ \forall\, t \in \mathcal{T}$, then

$$\begin{aligned}\nu(u^h) &= \min\{cx^t + u^h(b - Ax^t) \ : \ t \in \mathcal{T}\}\\ &= cx^{t^*} + u^h(b - Ax^{t^*})\ , \text{ for some } t^* \in \mathcal{T},\\ &= w_{t*}^h + u_0^h + u^h b \geq u_0^h + u^h b = w_0^h + \nu(u^b) > \nu(u^b)\,,\end{aligned}$$

and the best lower bound increases at least by w_0^h. Otherwise, the localization set L is reduced by adding a column $t^* \in \mathcal{T} \setminus \overline{\mathcal{T}}$ which prices out at the current point u^h. Thus, a method which selects in every iteration such a point $(u_0^h,\, u^h) \in L$ converges in a finite number of steps to an ϵ-optimal dual solution u.

Interior point decomposition methods choose a point $(u_0^h,\, u^h) \in L$ obeying some centrality property. The analytic center cutting plane method (ACCPM) (Goffin et al., 1992, 1993) selects the point $(u_0^h,\, u^h)$ which maximizes the (dual) potential function

$$\Psi(w) = \sum_{t \in \overline{\mathcal{T}}} \ln w_t + \ln w_0$$

over L. This requires to solve the non-linear system

$$\begin{aligned}&\mu_0 w_0 = \zeta\,,\ \mu_t w_t = \zeta\ \ \forall\, t \in \overline{\mathcal{T}}\,,\ \sum_{t \in \overline{\mathcal{T}}} \mu_t = \mu_0\,,\ \sum_{t \in \overline{\mathcal{T}}} \mu_t (Ax^t) = \mu_0 b\,,\\ &w_0 = u_0 + ub - \nu(u^b) > 0\,,\ w_t = cx^t - u_0 - u(Ax^t) > 0\ \ \forall\, t \in \overline{\mathcal{T}}\,,\end{aligned} \tag{27}$$

where $\zeta = 1$. If $(w_0^h,\, w^h,\, u_0^h,\, u^h,\, \mu_0^h,\, \mu^h)$ is a solution to the system above, then $(u_0^h,\, u^h)$ and $\alpha^h = \mu^h/\mu_0^h$ gives a feasible solution to the restricted dual master (24) and primal master (25), respectively.

Instead of computing the analytic center, Gondzio and Sarkissian (1996) as well as Martinson and Tind (1999) propose to use points on the central path between the analytic center and an optimal solution of the restricted master program (24). For these purposes a centrality parameter $\zeta > 0$ not necessarily equal to 1 is used and iteratively adjusted. Finally, Wentges (1997) simply proposes to select the point

$$(u_0^h,\, u^h) = \gamma(\overline{u}_0^h, \overline{u}^h) + (1 - \gamma)(u_0^b, u^b) \in L \qquad (0 < \gamma \leq 1)\,, \tag{28}$$

where $(u_0^b = \nu(u^b) - u^b b,\, u^b)$ is the best dual solution found so far and $(\overline{u}_0^h,\, \overline{u}^h)$ is an optimal solution of the restricted dual master program (24).

The parameter γ is first set to 1 and declined to a given threshold value in subsequent iterations. The convex combination (28) generally does not lie in the vicinity of a central path; nevertheless, the method is somehow related to interior point methods.

Van Roy (1983) introduces the cross decomposition method which combines Dantzig-Wolfe decomposition and Benders decomposition, and Van Roy (1986) uses this method to compute the bound Z_C for the CFLP. In order to avoid the need of solving a master problem in every iteration, the cross decomposition procedure obtains new dual iterates u^h from dual solutions to the primal subproblem, where integer variables are kept fixed, and generates new primal solutions x by solving the Lagrangean subproblem. When a convergence test indicates that convergence can no longer be expected, a master problem has to be solved in order to enforce convergence. Since subproblems are often easier to solve than master problems, a reduction in computational efforts is expected. If the primal subproblem does, however, not produce good dual information, the cross decomposition procedure may neither yield a reduced number of calls to the oracle nor a reduced number of master problems solved compared to a (stabilized) Dantzig-Wolfe decomposition method.

Last but not least, subgradient optimization and Dantzig-Wolfe decomposition can be combined in various ways in order to improve convergence. Guignard and Zhu (1994) use a two-phase method, which takes an optimal solution of the restricted dual master program (24) as next iterate only if subgradient steps fail to generate new columns for a given number of subsequent iterations. The restricted master is solved in every iteration in order to use the objective value $\max_u \overline{\nu}(u)$ as (improved) estimator of $\max_u \nu(u)$ in a commonly used step length formula.

Compared to Dantzig-Wolfe decomposition, bundle methods and interior point decomposition methods like ACCPM usually succeed in significantly decreasing the required number of calls to the oracle, that is the pricing subproblem. However, in case of large master programs and the addition of multiple columns in each iteration, the computational effort required for computing the next dual iterates can be substantial (in case of bundle methods a quadratic program needs to be solved; interior point methods (re-)solve the nonlinear system (27) by means of Newton methods). Such methods seem, therefore, to be suitable in case of a difficult subproblem and a relatively small master program. In case of the CFLP and the bound Z_D, however, even the restricted master program is usually quite large, while the pricing subproblem is relatively easy to solve. In order to solve the master program (16)–(22) it seems, therefore, more adequate to use a method which reduces the necessary number of calls to the master program even at the expense of an increased number of calls to the oracle. This can be achieved by means of mixing subgradient optimization and the weighted Dantzig-Wolfe decomposition approach:

- Good approximations of optimal multipliers η are quickly found by means of subgradient optimization. Furthermore, subsets of different columns y^t ($t \in \mathcal{T}^y$) and x_j^t ($t \in \mathcal{T}_j^x$) generated during subgradient optimization can be stored and used to initialize the restricted master program.
- The weighted decomposition scheme usually significantly improves convergence of Dantzig-Wolfe decomposition. Furthermore, the method is easy to use and "just" requires to reoptimize a linear program. In case that the dual prices u^h obtained from the convex combination (28) are dual feasible, the best lower bound $\nu(u^b)$ found so far increases by $\gamma\big(\overline{u}_0^h - \nu(u^b)\big)$. This gives the chance for further improvements in the current best lower bound if the parameter γ is increased in small steps until new columns price out at the current dual prices given by the optimal dual solution $(\overline{u}_0^h, \overline{u}^h)$ of the master program. Such a line search into the direction of $\overline{u}^h$ is feasible from a computational point of view only if the pricing subproblem is relatively easy to solve.
- Since the weighted decomposition as well as interior point methods may give feasible dual points, convergence can slow down at the end of the procedure if no additional columns are generated and the lower bound is already close to the optimum. Some intermediate additional subgradient steps may, however, help to overcome this situation. In our implementation, a limited number of intermediate subgradient steps are, therefore, performed in case that the next dual iterate obtained by the convex combination (28) is not dual feasible. During this intermediate subgradient phase, columns which price out at the dual prices given by the optimal dual solution to the master program are added.
- Furthermore, simple heuristics are used during and at the end of the column generation for determining (improved) feasible solutions for the CFLP. Columns from improved feasible solutions are added the master program. This guarantees that the objective function value of the restricted master program is no larger than the objective function value of a feasible solution for the CFLP. Finally, simple Lagrangean probing methods are employed in order to fix binary variables y_j if possible.

The column generation procedure employed for solving the master program (16)–(22) can be summarized as follows:

Column generation procedure

Phase 1 (subgradient phase)

Step 0: Set $h = 0$, $\overline{\mathcal{T}}^y = \overline{\mathcal{T}}_j^x = \emptyset\ \forall j \in J$, $LB = 0$, $UB = \infty$, $\overline{Z}_D = \infty$, $\sigma_h = 2$, and $\eta^h = \eta^{\min}$.

Step 1: Solve the pricing subproblem (7) for $\eta = \eta^h$. Let (y^h, x^h) denote the corresponding solution and let v^h, $\eta_0^h = Z_D(\eta^h) - \sum_{k \in K} \eta_k^h$ denote the values of v and η_0 corresponding to η^h. Set $O = \{j \in J : y_j^h = 1\}$. If $Z_D(\eta^h) > LB$, then set:

$$LB = Z_D(\eta^h),\ (y^b, x^b) = (y^h, x^h),\ \eta^b = \eta^h,\ v^b = v^h,\ \text{and}\ \eta_0^b = \eta_0^h\,.$$

Otherwise, half σ_h if the lower bound has not improved for a given number H^* of subsequent steps (e. g. $H^* = 10$). If $\big(\min\{\overline{Z}_D, UB\} - LB\big)/LB \leq \varepsilon$, go to Step 11.

Step 2: Set $\overline{T}^y := \overline{T}^y \cup \{h\}$ and $\overline{T}_j^x := \overline{T}_j^x \cup \{h\}\ \forall\, j \in O$.

Step 3: Solve the transportation problem with plant set O. If this gives a solution improving UB, update UB and record the solution in (y^B, x^B).

Step 4: Set $\eta_k^{h+1} = \max\{\eta_k^{\min}, \min\{\eta_k^{\max}, \eta_k^h + \theta_h g^h\}\}\ \forall\, k \in K$, where

$$g_k^h = 1 - \sum_{j\in J} x_{kj}^h \quad \text{and} \quad \theta_h = \sigma_h\big(UB - Z_D(\eta^h)\big)/\|g^h\|^2\,.$$

Set $h := h+1$. If h exceeds the iteration limit H (e. g. $H = 100$), go to Step 5, else go to Step 1.

Step 5: For each $j \in J$ compute

$$\rho_j = \min_y\Big\{\sum_{l\in J}(f_l - v_l^b)y_l\ :\ \sum_{l\in J} s_l y_l \geq d(K)\,,\ y_j = 1 - y_j^b,\ y_l \in \{0,1\}\ \forall\, l \in J\Big\}\,.$$

If $UB \leq \sum_{k\in K}\eta_k^b + \rho_j$, then fix variable y_j to value y_j^b. If any (additional) binary variable could be fixed this way, recompute $\eta^{\min}$, $\eta^{\max}$ and perform some additional subgradient steps, that is set e. g. $H := H + 5$ and go to step 1. Otherwise, continue with Step 6.

<u>Phase 2 (column generation phase)</u>

Step 6: Initialize the primal master problem with columns $\{y^t\ :\ t \in \overline{T}^y\}$ and $\{x_j^t\ :\ t \in \overline{T}_j^x\}$ for which

$$(1-\epsilon)F_t \leq \eta_0^b + \sum_{j\in J} v_j^b y_j^t \quad \text{and} \quad (1-\epsilon)C_{tj} \leq \sum_{k\in K}\eta_k^b x_{kj}^t - v_j^b \tag{29}$$

holds, using e. g., $\epsilon = 0.01$. Add columns $\{y^B\}$ and $\{x_j^B\ :\ y_j^B = 1\}$ from the feasible solution to the master problem.

Step 7: Solve the primal master problem (16)–(22) and obtain an optimal dual solution $(\overline{\eta}_0, \overline{\eta}, \overline{v})$ with objective value $\overline{Z}_D$. Remove all columns from the master which have been nonbasic for a certain number of subsequent iterations. If $\big(\overline{Z}_D - LB\big)/LB \leq \varepsilon$, go to Step 11. Otherwise, set $h := h + 1$, $\eta^h = \gamma\overline{\eta} + (1-\gamma)\eta^b$, where $0 < \gamma < 1$, and go to Step 8.

Step 8: Solve the subproblem as in Step 1. If columns $\{y^h\}$ or $\{x_j^h\}$ price out at the current dual prices $(\overline{\eta}_0, \overline{\eta}, \overline{v})$, add them to the master problem and go to Step 9. Otherwise, go to Step 10.

Step 9: Apply a limited number ΔH of additional subgradient steps, that is repeat Step 3, Step 4 and Step 1 ΔH times in this order. During this intermediate subgradient phase, add all columns which price out at the current dual prices $(\overline{\eta}_0, \overline{\eta}, \overline{v})$ to the master problem. Furthermore, apply Step 5 whenever an improved feasible solution (y^B, x^B) is found. Return to Step 7 after completion of this intermediate subgradient phase.

Step 10: As long as no column prices out at the current dual prices $(\overline{\eta}_0, \overline{\eta}, \overline{v})$, increase γ in small steps and repeat Step 1 and Step 3 with $\eta^h = \gamma\overline{\eta}+(1-\gamma)\eta^b$. During this "line search" also apply Step 5 whenever an improved feasible solution is found. Afterwards, go to Step 7.

Step 11: Let $(\overline{y}, \overline{x})$ denote the computed optimal solution to the master problem in terms of the original variables. Sort $\overline{y}$ in decreasing order and open plants j in this order as long as total capacity is insufficient or $\overline{y}_j$ exceeds a given threshold value, e. g. 0.75. If this way a solution improving *UB* is found, update *UB* and record the solution in (y^B, x^B).

In order to further explain some of the above steps, it is appropriate to comment on the following points:

- The knapsack problems (9) were solved by means of the COMBO algorithm of Martello et al. (1999).
- The step size strategy employed in phase 1 is proposed in Ryu and Guignard (1992).
- The tolerance ε in Step 1 and Step 7 was set equal to $1/(2^{15} - 1)$.
- The restricted master becomes too large, if all different columns generated during the subgradient phase are added. Since (η_0^b, η^b, v^b) approximates an optimal solution of the dual master, it is expected that columns not meeting the selection criterion (29) will be nonbasic.
- In order to limit the size of the master problem, inactive columns have to be removed (Step 7). This reduces the computation time required for each master problem and generally increases the number of master problems to be solved. In our implementation, columns are removed, if they are inactive for 5 subsequent iterations.
- The parameter γ was set to a value of 0.2 for smaller test problems and to a value of 0.05 for larger problem instances. In Step 10, the parameter γ is incremented in steps of 0.05.
- The number ΔH of intermediate subgradient steps in Step 9 was set to 10. Compared to an application of the procedure without the use of intermediate subgradient steps, this halfed the required computation time for some of the larger test problems, although the number of calls to the oracle increased.
- Whenever an improved feasible solution (y^B, x^B) is found during the column generation phase, columns x_j^B $(j \in J)$ which price out at the current dual prices $(\overline{\eta}_0, \overline{\eta}, \overline{v})$ are added to the master program.

3.3 Branch-and-Price Procedure

Let $(\overline{\alpha}, \overline{\beta})$ denote the optimal solution to the master problem (16)–(22) computed by means of the column generation procedure described above. The corresponding solution $(\overline{y}, \overline{x})$ in terms of the variables of the original problem formulation is then given by

$$\overline{y} = \sum_{t \in \mathcal{T}^y} \overline{\alpha}_t y^t \quad \text{and} \quad \overline{x}_{kj} = \sum_{t \in \mathcal{T}_j^x} \overline{\beta}_{tj} x_{kj}^t .$$

If $\overline{y}$ is integral (which means that $\overline{\alpha}$ is integral), an optimal solution to the CFLP is reached. Otherwise, the column generation procedure has to be combined with an implicit enumeration method in order to obtain optimal solutions.

If at least one $\overline{y}_j$ is fractional, possible branching strategies are to branch on single variables y_j with $\overline{y}_j \in (0,1)$ or to branch on more complex constraints involving variables y_j. The simplest branching strategy is to require $y_j = 0$ on the "left" branch and to fix y_j to 1 on the "right" branch. This branching rule is relatively easy to implement. If y_j is fixed to a value $\delta_j \in \{0,1\}$, all present columns y^t with $y_j^t = 1 - \delta_j$ have to be removed from the master problem. Furthermore, it is easy to enforce the branching constraint $y_j = \delta_j$ in the pricing subproblem (7).

Branching on single variables y_j may have, however, disadvantages. The master problem (16)–(22) is usually degenerated and possesses multiple optimal solutions. It is, therefore, likely that there will be only a small progress in the lower bound if a branch on a single variable is performed. This may be avoided by means of branching on subsets of variables y_j. If $S \subset J$ is such that $0 < \sum_{j \in S} \overline{y}_j < 1$, it is possible to branch by introducing the pair of branching constraints

$$\sum_{j \in S} y_j \geq 1 \quad \text{and} \quad y_j = 0 \, \forall \, j \in S .$$

In our implementation the set S is determined by sorting $\overline{y}$ in non-decreasing order and including in this order plants $j \in J$ into the set S until $\sum_{j \in S} \overline{y}_j \approx 0.5$. The branching constraint $y_j = 0 \; \forall j \in J$ is easy to handle; it just requires to exclude plants $j \in S$ from consideration. However, the branching constraint

$$\sum_{j \in S} y_j \geq 1 \quad \Leftrightarrow \quad \sum_{t \in \mathcal{T}^y} \Big(\sum_{j \in S} y_j^t \Big) \alpha_t \geq 1 \tag{30}$$

has to be added explicitly to the master problem. Each branching constraint of the type (30) contributes then to an additional dual variable, and the reduced costs of columns as well as the objective function of the pricing subproblem has to be adjusted accordingly. The general structure of the pricing subproblem is, however, not changed.

The branching rule suggested above is unbalanced in the sense that the constraint $y_j = 0 \; \forall j \in S$ is usually more restrictive than the constraint $\sum_{j \in S} y_j \geq 1$. One can expect that a branch with y_j fixed to zero for all $j \in S$ can be pruned quickly. In some sense, the above branching rule aims at generating an inequality of the type (30) which cuts off the fractional solution $(\overline{y}, \overline{x})$ and cannot be violated by an optimal solution to the CFLP. A possible drawback of this approach is that after the addition of several such constraints, the progress in the lower bound gets smaller and smaller.

The algorithm keeps a list of generated and processed subproblems (nodes of the enumeration tree) that were not fathomed. If this list is empty, optimality of the best feasible solution found is proved. Otherwise, the pending node with the smallest corresponding lower bound is selected for branching purposes.

In order to generate a subproblem, the following information is extracted from the father node. First the solution to the current master problem gives all active columns, which are stored together with an optimal basis. Furthermore, it is necessary to keep track of the best dual solution for the master problem; this information is required for performing steps of the weighted decomposition. After a node of the enumeration tree is selected, the branching constraint is added by deletion of the infeasible columns respective generation and addition of the new row $\sum_{j \in S} y_j \geq 1$. The storage requirements of this enumeration strategy are large; however, a "best-first" search strategy does usually contribute to a faster increase in the global lower bound than, for example, a depth-first search strategy.

4 Computational Results

The proposed branch-and-price procedure was coded in Sun Pascal and run on a Sun Ultra (300 MHz) to solve several test problems, which were generated according to the proposal of Cornuejols et al. (1991). Test problems for the CFLP generated this way are usually harder to solve than other problems of the same size. The test problems are divided into three different sets of problems which differ according to their tightness (ratio $r = \sum_j s_j / d(K)$ of total capacity and total demand). We used capacity tightness indices r of 3, 5 and 10, respectively. In each problem set, there are 5 problem types of each of the following sizes: 100 × 100, 200 × 100, 200 × 200, 500 × 100, and 500 × 200 where the first number is the number of customers and the second is the number of potential plant locations. Five problem instances have been generated for each given size and tightness index r. The transportation problems and the linear master problems arising in the computations of the bounds were solved by means of the procedures *CPXnetopt()* and *CPXprimopt()* contained in CPLEX's (1997) callable library (version 5.0).

In a first set of computational experiments, the computational effort required by the suggested column generation procedure was compared to the

Table1. LP relaxation solved using CPLEX

Size	SLP%	LP%	UB%	T_{SLP}	T_{LP}	T_H
			$r = \sum_j s_j/d(K) = 3$			
100×100	0.33	0.11	0.33	0.6	2.3	0.8
200×100	0.34	0.27	0.16	3.0	5.9	1.2
200×200	0.12	0.08	0.69	5.8	13.9	2.6
500×100	0.44	0.41	0.70	27.7	63.4	13.5
500×200	0.18	0.17	0.32	28.0	38.7	6.7
max	0.70	0.61	2.01	31.1	128.5	33.3
mean	0.28	0.21	0.44	13.0	24.8	5.0
			$r = \sum_j s_j/d(K) = 5$			
100×100	0.65	0.43	0.34	0.8	3.9	0.8
200×100	0.56	0.46	0.56	7.0	23.0	4.2
200×200	0.18	0.17	0.38	8.1	12.4	1.6
500×100	0.55	0.48	0.64	99.3	162.8	13.9
500×200	0.42	0.40	0.85	108.3	163.0	10.9
max	1.05	0.70	1.40	130.0	203.8	19.6
mean	0.47	0.39	0.55	44.7	73.0	6.3
			$r = \sum_j s_j/d(K) = 10$			
100×100	1.02	0.62	0.14	1.5	5.5	0.5
200×100	0.53	0.41	0.55	30.7	65.0	4.5
200×200	0.54	0.40	0.76	12.0	34.6	2.2
500×100	0.25	0.22	0.15	330.0	486.6	18.1
500×200	0.47	0.43	0.85	607.4	935.1	27.7
max	1.45	1.14	2.16	657.8	1138.2	50.6
mean	0.56	0.42	0.49	196.3	305.4	10.6
			Total			
max	1.45	1.14	2.16	657.8	1138.2	50.6
mean	0.44	0.34	0.49	84.7	134.4	7.3

effort required to solve the LP relaxation of the original problem formulation by means of a common linear programming software (CPLEX). This LP relaxation was solved in the following way: In a first step, the weak LP relaxation given by (O), (D), (C) and (N) is solved. Afterwards, violated implied bounds (B) are added until the strong LP relaxation is solved. Finally, in a third step, the LP relaxation was further strengthened by means of adding lifted cover inequalities, odd hole inequalities, flow cover inequalities and other submodular inequalities proposed by Aardal et al. (1995). In addition, simple rounding heuristics were applied in order to compute feasible solutions for the CFLP. Table 1 shows the results obtained with this procedure (averages over the five instances of each problem type). In Table 1, SLP% and LP% denote the percentage gap between the optimum value Z of the CFLP and the strong LP-bound Z^I and the computed LP-bound, respectively; UB% is the percentage deviation of the solution computed by the rounding heuris-

Table2. Results column generation procedure

Size	LB%	UB%	$\mathsf{It_{LR}}$	$\mathsf{It_M}$	$\mathsf{Col_A}$	$\mathsf{Col_{Tot}}$	$\mathsf{T_{LR}}$	$\mathsf{T_H}$	$\mathsf{T_M}$	$\mathsf{T_{Tot}}$
			$\mathsf{r} = \sum_j \mathsf{s_j/d(K)} = 3$							
100×100	0.05	0.00	183	11	239	518	0.8	0.3	0.4	1.7
200×100	0.23	0.00	232	25	418	807	1.6	1.2	1.0	4.0
200×200	0.05	0.00	220	18	370	896	2.8	1.7	1.5	6.3
500×100	0.40	0.27	575	56	1115	2618	12.2	11.5	28.6	52.8
500×200	0.16	0.02	300	24	984	2139	8.9	17.0	9.9	36.7
max	0.65	0.70	636	62	1210	2749	13.0	23.6	33.6	58.9
mean	0.18	0.06	302	27	625	1396	5.3	6.3	8.3	20.3
			$\mathsf{r} = \sum_j \mathsf{s_j/d(K)} = 5$							
100×100	0.19	0.06	206	12	285	523	0.9	0.2	0.4	1.6
200×100	0.44	0.26	395	53	404	925	3.3	1.8	3.0	8.4
200×200	0.12	0.06	217	18	410	821	3.0	1.8	1.1	6.5
500×100	0.55	0.88	915	100	1062	3091	19.7	18.6	99.3	138.4
500×200	0.40	0.37	631	68	1087	2772	25.7	21.2	39.6	87.6
max	0.73	1.95	1109	129	1196	3314	30.5	28.9	146.7	180.2
mean	0.34	0.33	473	50	649	1627	10.5	8.7	28.7	48.5
			$\mathsf{r} = \sum_j \mathsf{s_j/d(K)} = 10$							
100×100	0.23	0.00	323	26	253	585	0.8	0.1	0.9	1.9
200×100	0.46	0.34	473	55	300	953	4.0	1.8	7.1	13.0
200×200	0.21	0.00	364	44	472	935	3.5	0.8	1.9	6.5
500×100	0.24	0.19	1003	105	948	3188	16.5	29.8	190.3	237.5
500×200	0.47	1.09	901	91	1123	3312	42.1	35.5	172.3	251.7
max	0.75	1.50	1450	171	1246	3961	56.1	50.3	271.2	321.3
mean	0.32	0.32	613	64	619	1795	13.4	13.6	74.5	102.1
					Total					
max	0.75	1.95	1450	171	1246	3961	56.1	50.3	271.2	321.3
mean	0.28	0.24	463	47	631	1606	9.7	9.6	37.2	57.0

tic from optimality; $\mathsf{T_{SLP}}$, $\mathsf{T_{LP}}$, and $\mathsf{T_H}$ are the times in seconds required to compute the strong LP-bound, the LP-bound, and heuristic solutions.

Table 2 gives information about the computational effort required to solve the master problem (16)–(22) by means of the proposed column generation procedure. In this table, $\mathsf{It_{LR}}$ and $\mathsf{It_M}$ denotes the number of subproblems and master problems solved, respectively; $\mathsf{Col_A}$ is the number of columns in the last master problem, and $\mathsf{Col_{Tot}}$ is the total number of columns generated; $\mathsf{T_{LR}}$, $\mathsf{T_H}$, and $\mathsf{T_M}$ denote the computation times in seconds required for solving the subproblems, the transportation problems and the master problems, respectively; $\mathsf{T_{Tot}}$ is the total computation time in seconds; LB% and UB% denote the percentage deviation of the computed bound Z_D and heuristic solution from optimality, respectively. Table 2 shows that the number $\mathsf{It_{LR}}$ of pricing subproblems solved is large compared to the number $\mathsf{T_M}$ of master problems solved: The procedure tries to avoid the relatively large master problem at the expense of an increased number of calls to the oracle. The computation times were, however, out of the scope, if instead of a weighted decomposition approach standard Dantzig-Wolfe decomposition was used (which means to set the parameter γ in (28) to a value of 1). We also experimented with the stabilization method proposed by du Merle et al. (1999), using the same subgradient procedure in order to obtain a good guess for optimal multipliers. The computation times were, however, significantly larger than those obtained by means of the proposed column generation scheme. Compared to the lower bound produced by the linear programming approach (column LP% in Table 1), the bound Z_D is on average better than this LP-bound. Furthermore, the proposed column generation method consumes less computation time than the computation of a bound based on the linear relaxation and additional cutting planes (compare columns $\mathsf{T}_{\mathsf{Tot}}$ in Table 2 and T_{LP} in Table 1). For larger problems with a capacity tightness of 5 and 10 the column generation procedure even consumed less computation time than the computation of the strong LP-bound Z^I by means of a simplex algorithm (compare columns $\mathsf{T}_{\mathsf{Tot}}$ in Table 2 and $\mathsf{T}_{\mathsf{SLP}}$ in Table 1). This indicates that this bounding procedure should be useful in the framework of a branch-and-price procedure for solving larger problem instances; it provides strong bounds in relatively short computation times and, in contrast to subgradient optimization, also a fractional primal solution on which branching decisions can be based.

In a second set of computational experiments the branch-and-price procedure described in Section 3.3 was used to compute optimal solutions. The method was also compared to two other exact solution procedures. The code of the branch-and-price procedure is, however, still under development. Furthermore, comparing different exact optimization procedures on large problem instances is very time-consuming. We are, therefore, only able to show preliminary results on some selected test problems.

The first method used for computing optimal solutions is the CAPLOC algorithm of Ryu and Guignard (1992). CAPLOC is a depth-first search branch-and-bound procedure based on Z_D and subgradient optimization. Before branching at the top node, however, CAPLOC tries to fix as many y variables as possible by means of extensive Lagrangean probing. The second alternative exact solution approach just consists in solving the LP relaxation of the original formulation in the way described above and in passing the problem together with the generated cuts and the computed feasible solution to CPLEX's MIP optimizer. Table 3 compares the results obtained with CAPLOC and CPLEX for some selected test problems. In this table, the numbers in the column headed Problem show the number of customers, the number of potential plant sites, the capacity tightness index r and the number of the problem instance. Nodes is the number of nodes checked and $\mathsf{T}_{\mathsf{Tot}}$ the total CPU time in seconds. Furthermore, Iter and #TPs denotes the number of subgradient steps performed as well as the number of transportation problems solved by CAPLOC. As can be seen from Table 3, the CAPLOC algorithm clearly outperforms CPLEX for the test problems shown. (Although

Table3. Results obtained by means of CAPLOC and CPLEX

	CAPLOC				CPLEX	
Problem	Nodes	Iter	#TPs	T_{Tot}	Nodes	T_{Tot}
100×100-3-1	103	274	16	1.1	9	5.2
100×100-3-2	106	397	35	1.8	182	24.8
100×100-3-3	113	500	18	1.4	248	25.3
100×100-3-4	131	767	23	2.0	103	8.9
100×100-3-5	125	793	38	2.4	103	11.8
max	131	793	38	2.4	248	25.3
mean	116	546	26	1.7	129	15.2
100×100-5-1	133	703	32	1.9	548	55.6
100×100-5-2	106	398	24	1.4	33	10.5
100×100-5-3	154	1264	77	3.6	325	45.1
100×100-5-4	293	1643	101	4.9	750	121.4
100×100-5-5	252	1684	94	4.2	178	25.7
max	293	1684	101	4.9	750	121.4
mean	188	1138	66	3.2	367	51.7
100×100-10-1	103	257	10	1.1	9	10.6
100×100-10-2	102	248	13	1.1	89	12.9
100×100-10-3	118	498	22	1.5	214	31.0
100×100-10-4	139	800	31	1.9	120	23.1
100×100-10-5	102	247	12	1.1	6	10.6
max	139	800	31	1.9	214	31.0
mean	113	410	18	1.4	88	17.6
200×200-3-1	260	1123	42	14.1	1146	382.7
200×200-3-2	206	558	41	9.8	1559	779.0
200×200-3-3	3590	17704	731	190.4	3019	2066.9
200×200-3-4	275	1138	34	13.8	1915	1185.4
200×200-3-5	1158	6528	358	78.8	524	429.9
max	3590	17704	731	190.4	3019	2066.9
mean	1098	5410	241	61.4	1633	968.8
200×200-5-1	426	2294	212	29.1	1555	774.7
200×200-5-2	871	5069	595	65.1	2221	1208.3
200×200-5-3	351	1662	100	20.1	578	313.1
200×200-5-5	200	318	39	8.0	40	38.9
max	871	5069	595	65.1	2221	1208.3
mean	462	2336	237	30.6	1099	583.7
200×200-10-1	495	3132	225	29.4	1459	1055.4
200×200-10-2	205	463	37	8.3	850	1355.9
200×200-10-3	228	841	75	12.6	731	421.6
200×200-10-4	523	3085	194	28.0	1413	798.1
200×200-10-5	204	460	43	9.7	832	1428.4
max	523	3132	225	29.4	1459	1428.4
mean	331	1596	115	17.6	1057	1011.9
	Total				Total	
max	3590	17704	731	190.4	3019	2066.9
mean	384	1906	117	19.3	729	441.5