

Klose, Andreas; Drexler, Andreas

**Working Paper — Digitized Version**

## Lower bounds for the capacitated facility location problem based on column generation

Manuskripte aus den Instituten für Betriebswirtschaftslehre der Universität Kiel, No. 544

**Provided in Cooperation with:**

Christian-Albrechts-University of Kiel, Institute of Business Administration

*Suggested Citation:* Klose, Andreas; Drexler, Andreas (2001) : Lower bounds for the capacitated facility location problem based on column generation, Manuskripte aus den Instituten für Betriebswirtschaftslehre der Universität Kiel, No. 544, Universität Kiel, Institut für Betriebswirtschaftslehre, Kiel

This Version is available at:

<https://hdl.handle.net/10419/147622>

**Standard-Nutzungsbedingungen:**

Die Dokumente auf EconStor dürfen zu eigenen wissenschaftlichen Zwecken und zum Privatgebrauch gespeichert und kopiert werden.

Sie dürfen die Dokumente nicht für öffentliche oder kommerzielle Zwecke vervielfältigen, öffentlich ausstellen, öffentlich zugänglich machen, vertreiben oder anderweitig nutzen.

Sofern die Verfasser die Dokumente unter Open-Content-Lizenzen (insbesondere CC-Lizenzen) zur Verfügung gestellt haben sollten, gelten abweichend von diesen Nutzungsbedingungen die in der dort genannten Lizenz gewährten Nutzungsrechte.

**Terms of use:**

*Documents in EconStor may be saved and copied for your personal and scholarly purposes.*

*You are not to copy documents for public or commercial purposes, to exhibit the documents publicly, to make them publicly available on the internet, or to distribute or otherwise use the documents in public.*

*If the documents have been made available under an Open Content Licence (especially Creative Commons Licences), you may exercise further usage rights as specified in the indicated licence.*

Manuskripte  
aus den  
Instituten für Betriebswirtschaftslehre  
der Universität Kiel

No. 544

**Lower Bounds for the  
Capacitated Facility Location Problem  
Based on Column Generation**

Andreas Klose, Andreas Drexl



## Abstract

The Capacitated Facility Location Problem (CFLP) is a well-known combinatorial optimization problem with applications in distribution and production planning. 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 can be important. Therefore, we study various approaches for solving the master problems exactly. The algorithms employ different strategies for stabilizing the column generation process. Furthermore, a new lower bound for the CFLP based on partitioning the plant set and employing column generation is proposed. Computational results are reported for a set of large problem instances.

Keywords: Capacitated Facility Location Problem; Integer Programming; Lagrangean Relaxation; Column Generation

## 1 Introduction

The Capacitated Facility Location Problem (CFLP) 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).

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

$$Z = \min \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} + \sum_{j \in J} f_j y_j$$

$$\text{s. t. } \sum_{j \in J} x_{ij} = 1, \quad \forall i \in I \quad (\text{D})$$

$$\sum_{i \in I} d_i x_{ij} \leq s_j y_j, \quad \forall j \in J \quad (\text{C})$$

$$0 \leq x_{ij} \leq 1, \quad 0 \leq y_j \leq 1, \quad \forall i \in I, \forall j \in J \quad (\text{N})$$

$$y_j \in \{0, 1\}, \quad \forall j \in J \quad (\text{I})$$

where  $I$  is the set of customers and  $J$  the set of potential plant locations;  $c_{ij}$  is the cost of supplying all of customer  $i$ 's demand  $d_i$  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,  $x_{ij}$  denotes the fraction of customer  $i$ 's demand met from facility  $j$ . The constraints (D) are the demand constraints and constraints (C) are the capacity constraints. Without loss of generality it is assumed that  $c_{ij} \geq 0 \forall i, j$ ,  $f_j \geq 0 \forall j$ ,  $s_j > 0 \forall j$ ,  $d_i \geq 0 \forall i$ , and  $\sum_{j \in J} s_j > d(I) = \sum_{i \in I} d_i$ .

Numerous heuristic and exact algorithms for the CFLP have been proposed in the literature, and most solution approaches are based on Lagrangean relaxation (see Cornuejols et al. (1991) and Sridharan (1995) for a comprehensive survey). 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. For solving larger and/or more difficult instances of the CFLP, however, 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. The aim of this paper is, therefore, to investigate decomposition

methods for solving the master problems exactly. To this end, different strategies for stabilizing the column generation process are employed.

In §2 a new lower bound for the CFLP is proposed. Methods for stabilizing column generation are briefly summarized in §3 and used in §4 in order to compute important bounds for the CFLP exactly. Extensive computational experiments are presented and discussed in §5. Finally, the findings are summarized in §6.

## 2 A New Lower Bound for the CFLP

A common way to obtain lower bounds for the CFLP is to relax constraints (C) and/or (D) in a Lagrangean manner and to add some additional inequalities which are implied by the relaxed constraints and some of the other constraints. The valid inequalities which are usually considered for these purposes are the variable upper bound or trivial clique constraints

$$x_{ij} \leq y_j \quad \forall i \in I, \forall j \in J \quad (\text{B})$$

and the aggregate capacity constraint

$$\sum_{j \in J} s_j y_j \geq d(I). \quad (\text{T})$$

Besides the two additional constraints (B) and (T), one may devise a number of valid inequalities which can be useful to sharpen a relaxation, provided that the resulting subproblem is manageable. One group of redundant constraints is easily constructed as follows. Let  $\{J_q : q \in Q\}$ ,  $J_q \cap J_h = \emptyset \forall q \neq h$ , denote a given partitioning of the set  $J$  of potential plant locations. Then the “clique constraints”

$$\sum_{j \in J_q} x_{ij} \leq 1 \quad \forall i \in I, \forall q \in Q \quad (\text{U})$$

are implied by (D); however, they can be useful if constraints (D) are relaxed.

Without taking constraints (U) into account, Cornuejols et al. (1991) examine all possible ways of applying Lagrangean relaxation/decomposition to the CFLP. 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  $R_1$  and  $R_2$  are split into two subproblems.

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

$$Z^{BIU} \leq Z^{IU} \leq Z_C^{TU} \leq Z_C^U \leq Z, \quad Z^{IU} \leq Z_D^U \leq Z_C^U, \quad \text{and} \quad Z^{BIU} \leq Z_C^{BU} \leq Z_D^U.$$

Furthermore, they provide instances showing that all the inequalities above can be strict. The subproblem corresponding to  $Z_D^U$  can be converted to a knapsack problem and is solvable in pseudo-polynomial time. Therefore, bounds inferior to  $Z_D^U$  seem not to be interesting. Furthermore, as computational experiments show,  $Z_C^{TU} = Z_C^T$  is usually not stronger than  $Z_D^U$ . This leaves  $Z_D^U$  and  $Z_C^U = Z_C$  as candidate bounds. Since constraints (U) are implied by (D), constraints (U) can only be helpful if constraints (D) are relaxed. If the aggregate capacity constraint (T) is relaxed as well, the resulting Lagrangean subproblem decomposes into  $|Q|$  smaller CFLPs as will be made clear in §4. Obviously,

$$Z_D^T = \begin{cases} Z_D^{TU} = Z_{DU}^T = Z^{IU} = Z^I & , \text{ if } |Q| = |J| \\ Z & , \text{ if } |Q| = 1. \end{cases}$$

For  $1 < |Q| < |J|$ , however, the bound  $Z_D^T$  can be anywhere between the (strong) LP-bound  $Z^{IU} = Z^I$  and the optimum value  $Z$  of the CFLP, i.e.  $Z^I \leq Z_D^T \leq Z$ . Although the subproblem corresponding to  $Z_D^T$  has the same structure as the CFLP, the bound  $Z_D^T$  may be advantageous, if the set of potential plant locations is large and if the capacity constraints are not very tight.

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

$$Z_{C/D}^U = Z_{C/DB}^U = Z_{C/DT}^U = Z_{C/DBT}^U = Z_C^U, \max\{Z_C^{TU}, Z_D^U\} \leq Z_{D/TC}^U \leq Z_C^U, \\ \text{and } Z_{D/BC}^U = Z_{D/TBC}^U = Z_{TD/BC}^U = Z_D^U.$$

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  $Z_D^U$ . The only remaining interesting bound is, therefore,  $Z_{D/TC}^U$ . As shown by Chen and Guignard (1998), the bound  $Z_{D/TC}^U$  is also obtainable by means of a technique called Lagrangean substitution, which substitutes the copy constraints  $x = x'$  by  $\sum_i d_i x_{ij} = \sum_i d_i x'_{ij}$ . Compared to the Lagrangean decomposition, this reduces the number of dual variables from  $|I| \cdot |J| + |J|$  to  $2|J|$ .

In summary, interesting Lagrangean bounds for the CFLP are  $Z_D^U$ ,  $Z_C$ ,  $Z_{D/TC}^U$  and, possibly the new lower bound  $Z_D^T$ . Compared to  $Z_C$ , the computation of the bound  $Z_{D/TC}^U$  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  $Z_{D/TC}^U$  is no stronger than  $Z_C$  and since an APLP is often not much harder to solve than an UFLP, the bound  $Z_{D/TC}^U$  is not considered further in this paper. The computation of the other bounds by means of column generation, however, is described in detail in §4.

### 3 Methods for Stabilizing Column Generation

For larger instances of the CFLP, the computation of the above mentioned bounds requires the use of decomposition methods exhibiting good convergence behaviour. Furthermore, the choice and design of a decomposition method must also be guided by the degree of difficulty of the subproblem and (restricted) master problem resulting from the employed Lagrangean relaxation scheme. In the following, we summarize the main principles of the decomposition methods which we applied in pure or hybrid form in order to compute the bounds  $Z_D^U$ ,  $Z_C$  and  $Z_D^T$ .

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

$$\nu(u) = ub + \min\{(c - uA)x : x \in X\} = ub + \min\{(c - uA)x^t : t \in \mathcal{T}\}, \quad (1)$$

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  $\tilde{\mathcal{T}} \subset \mathcal{T}$  of columns, the function

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

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

$$\tilde{\nu}(u^h) \equiv \max_u \tilde{\nu}(u) = \max_{u_0, u} \{u_0 + ub : u_0 + uAx^t \leq cx^t \forall t \in \tilde{\mathcal{T}}\} \quad (2)$$

$$= \min_{\alpha \geq 0} \left\{ \sum_{t \in \tilde{\mathcal{T}}} (cx^t) \alpha_t : \sum_{t \in \tilde{\mathcal{T}}} (Ax^t) \alpha_t = b, \sum_{t \in \tilde{\mathcal{T}}} \alpha_t = 1 \right\}, \quad (3)$$

where  $u_0 \equiv \min\{(c - uA)x^t : t \in \tilde{\mathcal{T}}\}$  and  $\alpha_t$  is the dual variable corresponding to the dual cut  $u_0 + uAx^t \leq c^t$  for  $t \in \tilde{\mathcal{T}}$ . At each iteration of the standard column generation algorithm (Kelley 1960, Dantzig and Wolfe 1960), the restricted master problem (2) is solved and an optimal solution  $x^h$  of the Lagrangean/pricing subproblem (1) for fixed  $u = u^h$  is determined. The outer approximation  $\tilde{\nu}(u)$  is then refined by adding  $h$  to the set  $\tilde{\mathcal{T}}$  of columns. Since this algorithm suffers from bad convergence behaviour (Lemaréchal 1989), a variety of approaches for stabilizing column generation has 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

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

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

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

$$\begin{aligned} \max \quad & u_0 + ub - w^+ \pi^+ - w^- \pi^- \\ \text{s.t.} \quad & u_0 + uAx^t \leq cx^t, \quad \forall t \in \tilde{\mathcal{T}} \\ & \delta^- - w^- \leq u \leq \delta^+ + w^+, \\ & w^-, w^+ \geq 0. \end{aligned} \tag{4}$$

Du Merle et al. propose different strategies to initialize the parameters  $\pi^+$ ,  $\pi^-$ ,  $\delta^+$ ,  $\delta^-$  and to adapt them in case that an optimal solution  $u^h$  of (4) 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

$$P_1(u) = \sum_i \max\{0, \pi_i^+(u_i - \delta_i^+), \pi_i^-(\delta_i^- - u_i)\} \tag{5}$$

from the outer approximation  $\tilde{\nu}(u)$  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

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

where  $t > 0$  is a “trust” parameter and  $u^{h-1}$  the current point.

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

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

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

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

and the best lower bound increases at least by  $w_0^h$ . Otherwise, the localization set  $L$  is reduced by adding a column  $k \in \mathcal{T} \setminus \tilde{\mathcal{T}}$  which prices out at the current proposition  $u^h$ . 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  $\epsilon$ -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 (Goffin et al. 1992, 1993) selects the point  $(u_0^h, u^h)$  which maximizes the (dual) potential function

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

over  $L$ . This requires to solve the system

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

where  $\tau = 1$ . 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 (2) and primal master (3), respectively. It is straightforward to show (Goffin et al. 1993) that the primal solution  $\alpha^h$  has objective function value

$$U^h = \sum_{t \in \tilde{\mathcal{T}}} (cx^t) \alpha_t^h = |\tilde{\mathcal{T}}| w_0^h + (u_0^h + u^h b).$$

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 (2). For these purposes a centrality parameter  $\tau > 0$  not necessarily equal to 1 is used and iteratively adjusted. Finally, Wentges (1997) simply proposes to select the point

$$(u_0^h, u^h) = \gamma(\tilde{u}_0^h, \tilde{u}^h) + (1 - \gamma)(u_0^b, u^b) \in L \quad (0 < \gamma \leq 1), \quad (6)$$

where  $(u_0^b = \nu(u^b) - u^b b, u^b)$  is the best dual solution found so far and  $(\tilde{u}_0^h, \tilde{u}^h)$  is an optimal solution of the restricted master program (2). The parameter  $\gamma$  is first set to 1 and declined to a given threshold value in subsequent iterations. The convex combination (6) generally does not lie in the vicinity of a central path; nevertheless, the method is somehow related to interior point methods.

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 (2) as next proposition 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  $\max_u \tilde{\nu}(u)$  as (improved) estimator of  $\max_u \nu(u)$  in a commonly used step length formula.

## 4 Computation of the Lower Bounds

### 4.1 Relaxation of Demand Constraints

Ignoring the artificial constraints (U) and dualizing constraints (D) with multipliers  $\eta_i$ , gives the Lagrangean subproblem

$$Z_D^U(\eta) = \sum_{i \in I} \eta_i + \min_{x,y} \left\{ \sum_{i \in I} \sum_{j \in J} (c_{ij} - \eta_i) x_{ij} + \sum_{j \in J} f_j y_j : (C), (N), (I), (B), (T) \right\}. \quad (7)$$

It is easy to show, that optimal multipliers  $\eta^{opt}$  can be found in the interval  $[\eta^{\min}, \eta^{\max}]$ , where  $\eta_i^{\min} = \min_{j \neq j(i)} \{c_{ij}\}$ ,  $j(i) = \arg \min_j \{c_{ij}\}$ , and  $\eta_i^{\max} = \max_j \{c_{ij}\}$ . Furthermore, it is well-known that (7) can be reduced to a knapsack problem. To this end, define

$$v_j = \max_x \left\{ \sum_{i \in I} (\eta_i - c_{ij}) x_{ij} : \sum_{i \in I} d_i x_{ij} \leq s_j, 0 \leq x_{ij} \leq 1 \forall i \in I \right\} \quad (8)$$

in order to obtain  $Z_D^U(\eta) = \eta_0 + \sum_{i \in I} \eta_i$ , where

$$\eta_0 = \min_y \left\{ \sum_{j \in J} (f_j - v_j) y_j : \sum_{j \in J} s_j y_j \geq d(I), y_j \in \{0, 1\} \forall j \in J \right\}. \quad (9)$$

Let  $\{y^t : t \in \mathcal{T}^y\}$  denote the set of 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  $F_t = \sum_{j \in J} f_j y_j^t$  and  $C_{tj} = \sum_{i \in I} c_{ij} x_{ij}^t$ . If  $\tilde{\mathcal{T}}^y \subset \mathcal{T}^y$  and  $\tilde{\mathcal{T}}_j^x \subset \mathcal{T}_j^x$ ,  $j \in J$ , are sets of already generated columns, the restricted dual and primal master problem can be written as

$$\tilde{Z}_D^U = \max \eta_0 + \sum_{i \in I} \eta_i \quad (10)$$

$$\text{s.t. } \eta_0 + \sum_{j \in J} y_j^t v_j \leq F_t, \quad \forall t \in \tilde{\mathcal{T}}^y \quad (11)$$

$$\sum_{i \in I} x_{ij}^t \eta_i - v_j \leq C_{tj}, \quad \forall j \in J, \forall t \in \tilde{\mathcal{T}}_j^x \quad (12)$$

$$\eta_i^{\min} \leq \eta_i \leq \eta_i^{\max}, \quad \forall i \in I \quad (13)$$

$$\eta_0 \in \mathbb{R}, v_j \geq 0, \quad \forall j \in J \quad (14)$$

and

$$\tilde{Z}_D^U = \min \sum_{t \in \tilde{\mathcal{T}}^y} F_t \alpha_t + \sum_{j \in J} \sum_{t \in \tilde{\mathcal{T}}_j^x} C_{tj} \beta_{tj} + \sum_{i \in I} (\eta_i^{\max} \bar{p}_i - \eta_i^{\min} \underline{p}_i) \quad (15)$$

$$\text{s.t. } \sum_{t \in \tilde{\mathcal{T}}^y} \alpha_t = 1, \quad (16)$$

$$\sum_{t \in \tilde{\mathcal{T}}^y} y_j^t \alpha_t - \sum_{t \in \tilde{\mathcal{T}}_j^x} \beta_{tj} \geq 0, \quad \forall j \in J \quad (17)$$

$$\sum_{j \in J} \sum_{t \in \tilde{\mathcal{T}}_j^x} x_{ij}^t \beta_{tj} + \bar{p}_i - \underline{p}_i \geq 1, \quad \forall i \in I \quad (18)$$

$$\alpha_t \geq 0, \quad \forall t \in \tilde{\mathcal{T}}^y \quad (19)$$

$$\beta_{tj} \geq 0, \quad \forall j \in J, \forall t \in \tilde{\mathcal{T}}_j^x \quad (20)$$

$$\bar{p}_i, \underline{p}_i \geq 0 \quad \forall i \in I. \quad (21)$$

If  $(\tilde{\eta}_0, \tilde{\eta}, \tilde{v})$  denotes an optimal dual solution of the master problem, new columns  $x_{ij}^h$  and  $y^h$  price out, if

$$\tilde{v}_j < \sum_{i \in I} (\tilde{\eta}_i - c_{ij}) x_{ij}^h \Rightarrow \tilde{v}_j < \bar{v}_j \equiv \max \left\{ \sum_{i \in I} (\tilde{\eta}_i - c_{ij}) x_{ij}^t : t \in \mathcal{T}_j^x \right\}$$

and

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

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

Even the restricted master problem (15)–(21) is quite large, and the effort required for iteratively (re-)optimizing the restricted master problem can be tremendous. On the other hand, the subproblem (7) is generally relatively easy to solve. Stabilization methods which may save calls to the oracle at the expense of an increased effort for determining new propositions  $\eta$  from the localization set are, therefore, not adequate in this case. Good approximations of optimal multipliers  $\eta$  are, however, easily found by means of subgradient optimization. Thus, we combined subgradient optimization and “weighted” Dantzig-Wolfe decomposition in the following way in order to solve the full master problem:

### Procedure for computing $Z_D^U$

#### Phase 1 (subgradient phase)

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

*Step 1:* Solve (7) for  $\eta = \eta^h$ . Let  $(y^h, x^h)$  denote the corresponding solution and let  $v^h, \eta_0^h = Z_D^U(\eta^h) - \sum_{i \in I} \eta_i^h$  denote the values of  $v$  and  $\eta_0$  corresponding to  $\eta^h$ . Set  $O = \{j \in J : y_j^h = 1\}$ . If  $Z_D^U(\eta^h) > LB$ , then set:

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

Otherwise, half  $\sigma_h$  if the lower bound has not improved for a given number  $H^*$  of subsequent steps (e.g.  $H^* = 10$ ). If  $(\min\{\tilde{Z}_D^U, UB\} - LB)/LB \leq \varepsilon$ , then terminate.

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

*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  $\eta_i^{h+1} = \max\{\eta_i^{\min}, \min\{\eta_i^{\max}, \eta_i^h + \theta_h g^h\}\} \forall i \in I$ , where

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

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

$$\rho_j = \min_y \left\{ \sum_{l \in J} (f_l - v_l^b) y_l : \sum_{l \in J} s_l y_l \geq d(I), y_j = 1 - y_j^b, y_l \in \{0, 1\} \forall l \in J \right\}. \quad (22)$$

If  $UB \leq \sum_{i \in I} \eta_i^b + \rho_j$ , 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.

#### Phase 2 (column generation phase)

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

$$(1 - \epsilon)F_t \leq \eta_0^b + \sum_{j \in J} v_j^b y_j^t \quad \text{and} \quad (1 - \epsilon)C_{tj} \leq \sum_{i \in I} \eta_i^b x_{ij}^t - v_j^b \quad (23)$$

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

*Step 7:* Solve the primal master problem (15)–(21) and obtain an optimal dual solution  $(\tilde{\eta}_0, \tilde{\eta}, \tilde{v})$  with objective value  $\tilde{Z}_D^U$ . Remove all columns from the master which have been nonbasic for a

certain number of subsequent iterations. If  $(\tilde{Z}_D^U - LB)/LB \leq \varepsilon$ , then terminate. Otherwise, set  $h := h + 1$ ,  $\eta^h = \gamma\tilde{\eta} + (1 - \gamma)\eta^b$ , 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  $(\tilde{\eta}_0, \tilde{\eta}, \tilde{v})$ , add them to the master problem and go to Step 9. Otherwise, go to Step 10.

*Step 9:* Apply a limited number  $\Delta H$  of additional subgradient steps, that is repeat Step 3, Step 4 and Step 1  $\Delta H$  times in this order. During this intermediate subgradient phase, add all columns which price out at the current dual prices  $(\tilde{\eta}_0, \tilde{\eta}, \tilde{v})$  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  $(\tilde{\eta}_0, \tilde{\eta}, \tilde{v})$ , increase  $\gamma$  in small steps and repeat Step 1 and Step 3 with

$$\eta^h = \gamma\tilde{\eta} + (1 - \gamma)\eta^b. \quad (24)$$

During this “line search” also apply Step 5 whenever an improved feasible solution is found. Afterwards, go to Step 7.

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

- The step size strategy employed in phase 1 is proposed in Ryu and Guignard (1992).
- The tolerance  $\varepsilon$  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  $(\eta_0^b, \eta^b, v^b)$  approximates an optimal solution of the dual master, it is expected that columns not meeting the selection criterion (23) 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.
- As shown in §3 the dual prices  $\eta^h$  determined by the convex combination (24) may be feasible for the full dual master problem. In this case,  $Z_D^U(\eta^h)$  must improve the best lower bound  $LB$  at least by  $\gamma(\tilde{Z}_D^U - LB)$ . This gives the chance for further (small) improvements if  $\gamma$  is increased in small steps until a new column prices out. In our implementation we used a value  $\gamma = 0.2$  for smaller test problems and  $\gamma = 0.05$  for larger problems. In Step 10, the parameter  $\gamma$  is incremented in steps of 0.05.
- Weighted decomposition and interior point methods may give dual feasible propositions. This can slow down convergence at the very end of the procedure if no additional information is generated and the lower bound is already close to the optimum. The use of some additional subgradient steps in Step 9 has helped to overcome this situation. The number  $\Delta H$  of intermediate subgradient steps was set to 10. This way, the required computation time could be halved for some of the larger test problems, although the number of calls to the oracle is increased.
- The computation times were out of the scope if standard Dantzig-Wolfe decomposition was employed instead of weighted decomposition in the above procedure.
- 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 procedure described above.
- In addition to the simple heuristic of Step 3, we employed the following simple rounding heuristic at the very end of the overall procedure: Let  $(\tilde{y}, \tilde{x})$  denote an optimal solution of the last primal master problem. Sort  $\tilde{y}$  in decreasing order and open plants  $j$  in this order as long as total capacity is insufficient or  $\tilde{y}_j$  exceeds a given threshold value, e. g. 0.75.

## 4.2 Relaxation of Capacity Constraints

If the capacity constraints (C) are dualized with nonnegative multipliers  $u_j$ , the Lagrangean subproblem is

$$Z_C(u) = \min \left\{ \sum_{i \in I} \sum_{j \in J} (c_{ij} + u_j d_i) x_{ij} + \sum_{j \in J} (f_j - s_j u_j) y_j : (\text{D}), (\text{N}), (\text{I}), (\text{B}), (\text{T}) \right\}. \quad (25)$$

In this case, the artificial constraints (U) are redundant. Let  $\{(y^t, x^t) : t \in \mathcal{T}\}$  denote the vertices of the convex hull of all feasible solutions to problem (25). Since  $Z_C(u) \geq 0$ , the dual and primal master problem, restricted to a subset  $\tilde{\mathcal{T}} \subset \mathcal{T}$  of columns, is given by

$$\begin{aligned} \tilde{Z}_C &= \max u_0 \\ \text{s.t. } & u_0 + \sum_{j \in J} \left( s_j y_j^t - \sum_{i \in I} d_i x_{ij}^t \right) u_j \leq C_t, \quad \forall t \in \tilde{\mathcal{T}} \\ & u_0, u \geq 0 \end{aligned} \quad (26)$$

and

$$\begin{aligned} \tilde{Z}_C &= \min \sum_{t \in \tilde{\mathcal{T}}} C_t \alpha_t \\ \text{s.t. } & \sum_{t \in \tilde{\mathcal{T}}} \alpha_t \geq 1, \\ & \sum_{t \in \tilde{\mathcal{T}}} \left( s_j y_j^t - \sum_{i \in I} d_i x_{ij}^t \right) \alpha_t \geq 0, \quad \forall j \in J \\ & \alpha_t \geq 0 \quad \forall t \in \tilde{\mathcal{T}}, \end{aligned} \quad (27)$$

where

$$C_t = \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij}^t + \sum_{j \in J} f_j y_j^t.$$

Regarding  $Z_C$ , the situation is contrary to that found for  $Z_D^U$ . Now the subproblem is a difficult, strongly NP-hard problem, while the (restricted) master problem is generally small. This allows to use decomposition methods which investigate more effort in the computation of good propositions from the localization set in order to reduce the required number of calls to the oracle. Therefore, we employed in this case the analytic center cutting plane method. The procedure to compute  $Z_C$  can be summarized as follows:

### Procedure for computing $Z_C$

*Step 0:* Apply the subgradient phase of the algorithm for computing  $Z_D^U$  in order to obtain a feasible solution  $(x^B, y^B)$  with objective value  $UB$  and to possibly reduce the problem by means of the simple reduction test (22). Let  $\bar{\eta}$  denote optimal dual prices corresponding to the demand constraints (D) of the transportation problem with plant set  $O = \{j \in J : y_j^B = 1\}$ . Set  $h = 0$ ,  $LB = 0$ ,  $\tilde{\mathcal{T}} = \emptyset$ , and  $U^h = \infty$ . For each  $j \in J$  solve the linear program

$$\begin{aligned} s_j u_j^0 + \sum_{i \in I} \omega_{ij}^0 &= \min s_j u_j + \sum_{i \in I} \omega_{ij} \\ \text{s.t. } & d_i u_j + \omega_{ij} \geq \bar{\eta}_i - c_{ij}, \quad \forall i \in I \\ & u_j, \omega_{ij} \geq 0, \quad \forall i \in I \end{aligned} \quad (28)$$

in order to obtain initial multipliers  $u^0$ .

*Step 1:* Solve subproblem (25) with  $u = u^h$  and let  $(x^h, y^h)$  denote the corresponding solution. Set  $\tilde{T} := \tilde{T} \cup \{h\}$  and  $LB := \max\{LB, Z_C(u^h)\}$ . If  $(\min\{U^h, UB\} - LB)/LB \leq \varepsilon$ , then terminate. Otherwise go to Step 2.

*Step 2:* Solve the transportation problem with plant set  $O = \{j \in J : y_j^h = 1\}$ . If this gives a solution improving  $UB$ , update  $UB$ , store the solution in  $(x^B, y^B)$  and try to fix further variables  $y_j$  using the bounds  $\rho_j$  defined by (22).

*Step 3:* Set  $h := h + 1$ . Determine the analytic center  $(u_0^h, u^h)$  of the current localization set

$$L = \left\{ (u_0^h, u^h) \geq 0 : u_0 + \sum_{j \in J} \left( s_j y_j^t - \sum_{i \in I} d_i x_{ij}^t \right) u_j \leq C_t \quad \forall t \in \tilde{T}, u_0 \geq LB \right\}$$

and the corresponding primal feasible solution  $\alpha^h$  with objective function value  $U^h$ . If  $(U^h - LB)/LB \leq \varepsilon$ , terminate. Otherwise, return to Step 1.

The following remarks further explain the above procedure:

- In Step 0, Van Roy's (1986) method for sharpening Benders' cuts is used to determine initial multipliers  $u^0$ . A solution  $(\eta, u, \omega)$  which is feasible for the dual of the CFLP for fixed  $y$  defines a Bender's cut

$$Z \geq \sum_{i \in I} \eta_i - \sum_{j \in J} \left( s_j u_j + \sum_{i \in I} \omega_{ij} \right) y_j,$$

where the dual variables  $\omega$  correspond to the variable upper bounds (B). The above Benders' cut can be sharpened by means of solving the linear programs (28). The primal of (28) is a continuous knapsack problem which can be solved by sorting. Of course, a good starting solution  $u^0$  may also be obtained in some other way.

- As in the case of  $Z_D^U$ , the tolerance value  $\varepsilon$  in Step 1 and Step 3 is set to  $1/(2^{15} - 1)$ .

- In Step 3, the analytic center were computed by means of the C++-library ACCPM of Gondzio et al. (1996). To ensure existence of the analytic center, bounds on the multipliers  $u_j$  have to be added. For this purpose  $0 \leq u_j \leq \max_j f_j/s_j$  was chosen. The library ACCPM may change this bounds if necessary. Furthermore, ACCPM uses warm start procedures in order to recompute the analytic center. To this end, dual feasibility is first recovered and then the point recentered using damped primal Newton steps (Goffin and Vial 1999).

- As in the case of  $Z_D^U$ , the same rounding heuristic is applied at the very end of the overall procedure in order to possibly further improve the upper bound  $UB$ .

- Again we experimented with the stabilization method proposed by du Merle et al. (1999). In order to possibly further improve the initial multipliers, we first applied some steps of Van Roy's (1986) cross decomposition algorithm until the dual convergence test was not passed for the first time. Furthermore, we also added columns from primal feasible solutions. This is done by representing a solution  $\bar{x}$  of the transportation problem as convex combination of some vertices of  $\{x \geq 0 : \sum_{j \in J} x_{ij} = 1 \quad \forall i \in I\}$  (Van Roy 1986). For some of the test problems this method gave better results than ACCPM, while for other test problems convergence was much slower. Since we did not succeed in finding an updating strategy of the parameters (unit penalty cost  $\pi^-$ ,  $\pi^+$  and box size  $\delta^-$ ,  $\delta^+$ ) which produced uniformly good results for all test problems, we finally favored ACCPM.

### 4.3 Lower Bound Based on Partitioning the Plant Set

If constraint (T) is ignored instead of constraints (U) and the demand constraints are dualized with multipliers  $\lambda_i$ , the Lagrangean subproblem is

$$Z_D^T(\lambda) = \sum_{i \in I} \lambda_i + \min \left\{ \sum_{q \in Q} \left( \sum_{i \in I} \sum_{j \in J_q} (c_{ij} - \lambda_i) x_{ij} + \sum_{j \in J_q} f_j y_j \right) : (C), (N), (I), (B), (U) \right\}. \quad (29)$$

The subproblem (29) decomposes into the  $|Q|$  subproblems

$$\begin{aligned}
z_q = \min & \sum_{i \in I} \sum_{j \in J_q} (c_{ij} - \lambda_i) x_{ij} + \sum_{j \in J_q} f_j y_j \\
\text{s.t.} & \sum_{j \in J_q} x_{ij} \leq 1, \quad \forall i \in I \\
& \sum_{i \in I} d_i x_{ij} \leq s_j y_j, \quad \forall j \in J_q \\
& 0 \leq x_{ij} \leq y_j, \quad y_j \in \{0, 1\}, \quad \forall i \in I, \forall j \in J_q.
\end{aligned} \tag{30}$$

The subproblem (30) is an “Anti-CFLP” which can be converted to a CFLP by introducing an artificial plant with index  $0_q$ , fixed cost  $f_{0_q} = 0$ , supply costs  $c_{i0_q} = 0 \forall i \in I$  and capacity  $s_{0_q} = d(I)$ . In this sense, the above relaxation decomposes the CFLP into a set of smaller CFLPs which have to be solved repeatedly. Let  $\{(x^t, y^t) : t \in \mathcal{T}_q\}$  denote the set of all vertices of the convex hull of the feasible solutions to problem (30). Since  $z_q \leq 0 \forall q \in Q$ , the dual and primal master problem, restricted to subsets  $\tilde{\mathcal{T}}_q \subset \mathcal{T}_q$  of columns, may then be written as

$$\begin{aligned}
\tilde{Z}_D^T = \max & \sum_{q \in Q} z_q + \sum_{i \in I} \lambda_i \\
\text{s.t.} & z_q + \sum_{i \in I} \left( \sum_{j \in J_q} x_{ij}^t \right) \lambda_i \leq C_{qt}, \quad \forall q \in Q, \forall t \in \tilde{\mathcal{T}}_q \\
& \lambda_i \geq 0, \quad z_q \leq 0, \quad \forall i \in I, \forall q \in Q
\end{aligned} \tag{31}$$

and

$$\begin{aligned}
\tilde{Z}_D^T = \min & \sum_{q \in Q} \sum_{t \in \tilde{\mathcal{T}}_q} C_{qt} \alpha_{qt} \\
\text{s.t.} & \sum_{t \in \tilde{\mathcal{T}}_q} \alpha_{qt} \leq 1, \quad \forall q \in Q \\
& \sum_{q \in Q} \sum_{t \in \tilde{\mathcal{T}}_q} \left( \sum_{j \in J_q} x_{ij}^t \right) \alpha_{qt} \geq 1, \quad \forall i \in I \\
& \alpha_{qt} \geq 0, \quad \forall q \in Q, \forall t \in \tilde{\mathcal{T}}_q,
\end{aligned} \tag{32}$$

respectively, where

$$C_{qt} = \sum_{i \in I} \sum_{j \in J_q} c_{ij} x_{ij}^t + \sum_{j \in J_q} f_j y_j^t.$$

In the case of  $|Q| = |J|$ , that is  $|J_q| = 1 \forall q \in Q$ , and  $\tilde{\mathcal{T}}_q = \mathcal{T}_q \forall q \in Q$ , we have  $\tilde{Z}_D^T = Z_D^T = Z_D^{TU} = Z^{IU} = Z^I$ , which is the (strong) LP-bound. Assume that  $(\bar{x}, \bar{y})$  solves the LP relaxation and that  $\bar{\lambda}_i, i \in I$ , are corresponding optimal dual variables associated with the demand constraints (D). Let  $\{(x^t, y^t) : t \in \mathcal{T}_{\bar{\lambda}}\}$  be the set of all optimal solutions to the subproblem

$$\bar{\lambda}_0 = \min \left\{ \sum_{i \in I} \sum_{j \in J} (c_{ij} - \bar{\lambda}_i) x_{ij} + \sum_{j \in J} f_j y_j : (C), (N) (I), (B) \right\}. \tag{33}$$

Then

$$Z^I = Z_D^{TU} = \bar{\lambda}_0 + \sum_{i \in I} \bar{\lambda}_i \quad \text{and} \quad (\bar{x}, \bar{y}) = \sum_{t \in \mathcal{T}_{\bar{\lambda}}} \bar{\alpha}_t (x^t, y^t)$$

for some  $\bar{\alpha} \geq 0$  with  $\sum_{t \in \mathcal{T}_{\bar{\lambda}}} \bar{\alpha}_t = 1$ . A necessary condition for the bound  $Z_D^T$  with  $1 < |Q| < |J|$  to be stronger than the LP-bound  $Z^I$  is then, that

$$\sum_{j \in J_q} x_{ij}^t > 1$$

holds for at least one  $q \in Q$ ,  $t \in \mathcal{T}_{\bar{\lambda}}$  and  $i \in I$ . The plant set  $J$  should, therefore, be partitioned in such a way that the above condition has a good chance to be met. One possible way to achieve this is the following: Let  $\hat{\lambda}$  denote near optimal dual prices associated with constraints (D) in the linear relaxation of the CFLP. Such an approximation of  $\bar{\lambda}$  is usually quickly obtained using subgradient optimization. Furthermore, let  $(\hat{x}, \hat{y})$  be an optimal solution of the subproblem (33) if  $\bar{\lambda}$  is replaced with  $\hat{\lambda}$ . Perform the steps below in order to decompose the set  $J$  into subsets:

1. Set  $q = 0$  and define:  $\hat{r}_i = \sum_{j \in J} \hat{x}_{ij}$ ,  $\hat{I} = \{i \in I : \hat{r}_i > 1\}$ ,  $H_i = \{j \in J : \hat{x}_{ij} = 0\}$ , and  $\hat{I}_i = \{k \in \hat{I} : \sum_{j \in H_i} \hat{x}_{kj} > 1\}$ .

2. As long as  $\hat{I} \neq \emptyset$  perform the following steps: (a) Select customer  $i \in \hat{I}$  with largest value of  $|\hat{I}_i|$ . If  $|\hat{I}_i| = 0$ , select customer  $i \in \hat{I}$  with largest value of  $\hat{r}_i$ . (b) Set  $q := q + 1$ ,  $J_q = J \setminus H_i$ ,  $\hat{r}_k := \hat{r}_k - \sum_{j \in J_q} \hat{x}_{kj} \forall k \in \hat{I}$ , and  $\hat{I} := \{k \in \hat{I} : \hat{r}_k > 1\}$ .

3. If  $q = 1$ , set  $J_2 = J \setminus J_1$ . Otherwise assign each plant  $j$  with  $\hat{y}_j = 0$  to the subset  $J_q$  which minimizes  $\min_{l \in J_q} \sum_{i \in I} |c_{ij} - c_{il}|$ .

The above procedure first decomposes the set of plants which are open in the solution  $(\hat{x}, \hat{y})$ . In the first iteration, a constraint  $\sum_{j \in J} x_{ij} \leq 1$  violated by  $\hat{x}$  is selected and  $J_1$  determined as  $J_1 = \{j \in J : \hat{x}_{ij} > 0\}$ . The next subset  $J_2$  may then only be constructed from plants  $j \in H_i$  with  $\hat{y}_j = 1$ . Furthermore, the set of constraints  $\sum_{j \in J_2} x_{ij} \leq 1$  possibly violated by  $\hat{x}$  is reduced to the set  $\hat{I}_i$ . This motivates the choice of the customer  $i$  in the above procedure. Obviously, the plant set may be decomposed in many other plausible ways.

The relaxation (29) does not make use of the aggregate capacity constraint (T). Therefore, the relaxation should usually not be preferable in the case of tight capacity constraints. Furthermore, the approach makes only sense in the case of large problems with a large set of potential plant sites. For such problems it may be better to solve several smaller CFLPs than for e.g. several large APLPs in order to derive strong lower and upper bounds. Since in any case, solving the hard subproblems (30) causes the main effort, we selected again ACCPM to solve the master problem, although the size of the restricted master problems and the effort required for computing analytic centers can be substantial in this case. The overall procedure to compute  $Z_D^T$  then consists in the application of the steps below:

### Procedure for computing $Z_D^T$

*Step 0:* Apply the subgradient phase of the algorithm for computing  $Z_D^U$  in order to obtain a feasible solution  $(x^B, y^B)$  with objective value  $UB$  and to possibly reduce the problem by means of the simple reduction test (22). Furthermore, let  $\hat{Z}_D^U$  denote the computed lower bound and let  $\hat{\eta}$  denote the corresponding multipliers.

*Step 1:* Apply the same subgradient procedure without consideration of the aggregate capacity constraint (T). To this end, use  $\hat{Z}_D^U$  as target value in the step length formula. Let  $\hat{\lambda}$  denote the computed multipliers. Furthermore, let  $(\hat{x}, \hat{y})$  be the solution of subproblem (33) corresponding to  $\hat{\lambda}$ .

*Step 2:* Apply the above procedure for decomposing the plant set  $J$ . Set  $h = 0$ ,  $LB = 0$ ,  $\tilde{Z}_D^T = U^h = \infty$ ,  $\tilde{\mathcal{T}}_q = \emptyset \forall q \in Q$ , and  $\lambda^0 = \hat{\lambda}$ .

*Step 3:* Solve the subproblems (30) for each  $q \in Q$  in order to obtain  $Z_D^T(\lambda)$ . Let  $(x^{hq}, y^{hq})$  denote the corresponding solutions. Set  $\tilde{\mathcal{T}}_q := \tilde{\mathcal{T}}_q \cup \{h\} \forall q \in Q$ . Set  $LB = \max\{LB, Z_D^T(\lambda^h)\}$ . If  $(\min\{U^h, UB\} - LB)/LB \leq \varepsilon$ , then terminate.

*Step 4:* If  $\sum_{q \in Q} \sum_{j \in J_q} s_j y_j^{h_q} \geq d(I)$ , then solve the transportation problem with plant set  $O = \bigcup_{q \in Q} \{j \in J_q : y_j^{h_q} = 1\}$ . If this gives a solution improving  $UB$ , update  $UB$ , store the solution in  $(x^B, y^B)$  and try to fix further variables  $y_j$  using the bounds  $\rho_j$  defined by (22).

*Step 5:* Set  $h := h + 1$ . Determine the analytic center  $(z^h, \lambda^h)$  of the current localization set

$$L = \left\{ (z, \lambda) : z_q + \sum_{i \in I} \sum_{j \in J_q} x_{ij}^t \lambda_i \leq C_{qt} \quad \forall q \in Q, \forall t \in \tilde{T}_q; \sum_{q \in Q} z_q + \sum_{i \in I} \lambda_i \geq LB \right\}$$

and the corresponding primal feasible solution  $\alpha^h$  with objective function value  $U^h$ . If  $(U^h - LB)/LB \leq \varepsilon$ , terminate. Otherwise, return to Step 3.

In addition to the already mentioned simple rounding heuristic, we used two further heuristics at the very end of the overall procedure in order to possibly further improve the upper bound  $UB$ . Let  $\bar{\alpha}$  denote the solution of the last primal master problem (32) and set  $\bar{y}_j = \sum_{t \in \tilde{T}_q} \bar{\alpha}_{qt} y_j^t$  for each  $q \in Q$  and  $j \in J_q$ . The first heuristic consists in solving the knapsack problem (9) using the multipliers  $\hat{\eta}$  computed in Step 0, where in addition some of the variables  $y_j$  are fixed to zero or one if  $\bar{y}_j \leq 0.1$  or  $\bar{y}_j \geq 0.9$ . The second heuristic simply applies some interchange moves to the best solution found so far, where, however, a plant  $j$  with  $\bar{y}_j \leq 0.1$  must not be opened and a plant  $j$  with  $\bar{y}_j \geq 0.9$  must not be closed.

## 5 Computational Results

The proposed procedures for computing the bounds  $Z_D^U$ ,  $Z_C$  and  $Z_D^T$  were 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  $r = \sum_j s_j / d(I)$  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:  $100 \times 100$ ,  $200 \times 100$ ,  $200 \times 200$ ,  $500 \times 100$ , and  $500 \times 200$  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  $r$ . In order to solve the various subproblems arising in the computations of the bounds, the following procedures and codes were used:

- The knapsack problems (9) were solved by means of the COMBO algorithm of Martello et al. (1999).
- A branch-and-bound algorithm coded in Pascal and proposed in Klose (1998) was used to solve the APLP (25).
- The subproblem (29), which itself is a CFLP, was solved by means of a branch-and-bound algorithm called CAPLOC and proposed and coded in FORTRAN by Ryu and Guignard (1992).
- As already mentioned, analytic centers were computed with the help of the C++-library of Gondzio et al. (1996).
- The transportation problems and the linear master problems were solved by means of the procedures *CPXnetopt()* and *CPXprimopt()* contained in CPLEX's (1997) callable library (version 5.0).

For the purposes of comparison, optimal solutions were computed by means of two different exact solution procedures. The first exact solution procedure is the CAPLOC algorithm of Ryu and Guignard (1992). CAPLOC is a depth-first search branch-and-bound procedure which is based on  $Z_D^U$  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.

**Table 1 Results obtained by means of CAPLOC**

| Size                       | UB%  | Nodes  | Iter   | #TPs   | $T_{Tot}$ |
|----------------------------|------|--------|--------|--------|-----------|
| $r = \sum_j s_j/d(l) = 3$  |      |        |        |        |           |
| 100×100                    | 0.00 | 116    | 546    | 26     | 1.7       |
| 200×100                    | 0.00 | 510    | 3255   | 401    | 33.1      |
| 200×100                    | 0.00 | 1098   | 5410   | 241    | 61.4      |
| 500×100                    | 0.00 | 14526  | 72327  | 20778  | 4972.2    |
| 500×200                    | 0.00 | 27768  | 94686  | 15192  | 7680.3    |
| max                        | 0.00 | 51316  | 259059 | 75075  | 18173.7   |
| mean                       | 0.00 | 8803   | 35245  | 7328   | 2549.7    |
| $r = \sum_j s_j/d(l) = 5$  |      |        |        |        |           |
| 100×100                    | 0.00 | 188    | 1138   | 66     | 3.2       |
| 200×100                    | 0.00 | 1226   | 7181   | 1593   | 87.6      |
| 200×200                    | 0.00 | 639    | 3368   | 445    | 51.8      |
| 500×100                    | 0.00 | 34718  | 131362 | 66407  | 11448.1   |
| 500×200                    | 0.04 | 92860  | 303274 | 113136 | 32849.0   |
| max                        | 0.18 | 127377 | 380128 | 173687 | 45003.7   |
| mean                       | 0.01 | 25926  | 89265  | 36329  | 8887.9    |
| $r = \sum_j s_j/d(l) = 10$ |      |        |        |        |           |
| 100×100                    | 0.00 | 113    | 410    | 18     | 1.3       |
| 200×100                    | 0.00 | 1180   | 5161   | 1639   | 74.4      |
| 200×200                    | 0.00 | 331    | 1596   | 115    | 17.6      |
| 500×100                    | 0.00 | 13913  | 47185  | 25698  | 4104.5    |
| 500×200                    | 0.08 | 85571  | 259254 | 109259 | 28911.2   |
| max                        | 0.41 | 124113 | 380786 | 220294 | 45001.6   |
| mean                       | 0.02 | 20221  | 62721  | 27346  | 6621.8    |
| Total                      |      |        |        |        |           |
| max                        | 0.41 | 127377 | 380786 | 220294 | 45003.7   |
| mean                       | 0.01 | 18317  | 62410  | 23668  | 6019.8    |

Table 1 shows the results (averages over the five instances of each problem type) obtained in this way. The computation time was limited to 12.5 hours per instance. In Table 1, Nodes is the number of nodes checked, Iter is the total number of subgradient steps performed, #TPs is the number of transportation problems solved,  $T_{Tot}$  is the total CPU time in seconds, and UB% is the percentage deviation of the computed solution from optimality in the case that the procedure was aborted after 12.5 hours of computation time. Although the computation times for problems with a tightness of  $r = 10$  were slightly better than those for problems with a tightness of  $r = 5$ , the computational effort tends to increase with decreasing capacity tightness. The quality of the lower and upper bound computed by the subgradient procedure usually deteriorates with increasing values of  $r$ , and the preprocessing procedure does not succeed in fixing a large number of  $y$  variables. Two of the largest problems with ratio  $r = 5$  and one of the largest problems with ratio  $r = 10$  could not be solved to optimality by means of CAPLOC, at least not within a computation time of 12.5 hours.

The second exact solution approach first applies an LP-based heuristic originally proposed in Klose (1999) for a two-stage facility location problem and adopted here to the case of the CFLP. Starting with the weak LP-relaxation, the method iteratively refines the relaxation by means of adding different polyhedral cuts. After recalculating the LP solution, some heuristics are

**Table 2 Results LP/cutting plane approach**

| Size                       | SLP% | LP%  | UB <sub>0</sub> % | UB%  | Gap% | Nodes | T <sub>SLP</sub> | T <sub>LP</sub> | T <sub>H</sub> | T <sub>Tot</sub> |
|----------------------------|------|------|-------------------|------|------|-------|------------------|-----------------|----------------|------------------|
| $r = \sum_j s_j/d(l) = 3$  |      |      |                   |      |      |       |                  |                 |                |                  |
| 100×100                    | 0.33 | 0.11 | 0.33              | 0.00 | 0.00 | 129   | 0.6              | 2.3             | 0.8            | 15.2             |
| 200×100                    | 0.34 | 0.27 | 0.16              | 0.00 | 0.00 | 431   | 3.0              | 5.9             | 1.2            | 141.1            |
| 200×200                    | 0.12 | 0.08 | 0.69              | 0.00 | 0.00 | 1633  | 5.8              | 13.9            | 2.6            | 968.8            |
| 500×100                    | 0.44 | 0.41 | 0.70              | 0.00 | 0.00 | 3764  | 27.7             | 63.4            | 13.5           | 10277.2          |
| 500×200                    | 0.18 | 0.17 | 0.32              | 0.09 | 0.10 | 7787  | 28.0             | 38.7            | 6.7            | 24164.9          |
| max                        | 0.70 | 0.61 | 2.01              | 0.44 | 0.48 | 13765 | 31.1             | 128.5           | 33.3           | 45031.8          |
| mean                       | 0.28 | 0.21 | 0.44              | 0.02 | 0.02 | 2749  | 13.0             | 24.8            | 5.0            | 7113.4           |
| $r = \sum_j s_j/d(l) = 5$  |      |      |                   |      |      |       |                  |                 |                |                  |
| 100×100                    | 0.65 | 0.43 | 0.34              | 0.00 | 0.00 | 367   | 0.8              | 3.9             | 0.8            | 51.7             |
| 200×100                    | 0.56 | 0.46 | 0.56              | 0.00 | 0.00 | 941   | 7.0              | 23.0            | 4.2            | 706.7            |
| 200×200                    | 0.18 | 0.17 | 0.38              | 0.00 | 0.00 | 2212  | 8.1              | 12.4            | 1.6            | 2337.5           |
| 500×100                    | 0.55 | 0.48 | 0.64              | 0.00 | 0.00 | 1902  | 99.3             | 162.8           | 13.9           | 7494.2           |
| 500×200                    | 0.42 | 0.40 | 0.85              | 0.67 | 0.81 | 6159  | 108.3            | 163.0           | 10.9           | 45139.9          |
| max                        | 1.05 | 0.70 | 1.40              | 1.15 | 1.35 | 10438 | 130.0            | 203.8           | 19.6           | 45163.6          |
| mean                       | 0.47 | 0.39 | 0.55              | 0.13 | 0.16 | 2316  | 44.7             | 73.0            | 6.3            | 11146.0          |
| $r = \sum_j s_j/d(l) = 10$ |      |      |                   |      |      |       |                  |                 |                |                  |
| 100×100                    | 1.02 | 0.62 | 0.14              | 0.00 | 0.00 | 88    | 1.5              | 5.5             | 0.5            | 17.6             |
| 200×100                    | 0.53 | 0.41 | 0.55              | 0.00 | 0.00 | 179   | 30.7             | 65.0            | 4.5            | 320.6            |
| 200×200                    | 0.54 | 0.40 | 0.76              | 0.00 | 0.00 | 1057  | 12.0             | 34.6            | 2.2            | 1011.9           |
| 500×100                    | 0.25 | 0.22 | 0.15              | 0.00 | 0.00 | 122   | 330.0            | 486.6           | 18.1           | 1614.0           |
| 500×200                    | 0.47 | 0.43 | 0.85              | 0.52 | 0.64 | 1384  | 607.4            | 935.1           | 27.7           | 36136.0          |
| max                        | 1.45 | 1.14 | 2.16              | 1.58 | 1.86 | 1998  | 657.8            | 1138.2          | 50.6           | 46135.9          |
| mean                       | 0.56 | 0.42 | 0.49              | 0.10 | 0.13 | 566   | 196.3            | 305.4           | 10.6           | 7820.0           |
| Total                      |      |      |                   |      |      |       |                  |                 |                |                  |
| max                        | 1.45 | 1.14 | 2.16              | 1.58 | 1.86 | 13765 | 657.8            | 1138.2          | 50.6           | 46135.9          |
| mean                       | 0.44 | 0.34 | 0.49              | 0.09 | 0.10 | 1877  | 84.7             | 134.4           | 7.3            | 8693.1           |

applied in order to compute a feasible solution from the current LP solution. If no additional inequalities are found, the problem together with the added cuts and the computed feasible solution is passed to CPLEX's subroutine *CPXmipoptimize()* in order to close the remaining gap between the lower and upper bound. Table 2 summarizes the results obtained with this procedure. In Table 2, 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<sub>0</sub>% and UB% are the percentage deviations of the solution computed by the LP-based heuristic and the best feasible solution found from optimality; Gap% is the remaining gap between the global lower bound and the best feasible solution found in the case that the computation time exceeded the maximum allowed time of 12.5 hours; Nodes is the number of nodes checked by CPLEX's *mipoptimize()*; T<sub>SLP</sub>, T<sub>LP</sub>, and T<sub>H</sub> are the times required to compute the strong LP bound, the LP-bound, and heuristic solutions; T<sub>Tot</sub> is the total computation time in seconds. A comparison between Table 1 and Table 2 shows that CAPLOC outperforms the LP approach in the case of tight capacity constraints or problems of smaller to medium size. On the other hand, the LP approach was much faster for several larger problems (size 500×100) with tightness of 5 and 10. Solving the largest problems by means of the above procedure, however, can be time-consuming.

Especially in the case of “loose” capacity constraints, the linear program is usually quite large. Furthermore, the enumeration tree can be of considerable size and the progress in the lower bound very small. In total, 10 problems of size  $500 \times 200$  (2 problems in the case of  $r = 3$ , 5 problems in the case  $r = 5$ , and 3 problems in the case of  $r = 10$ ) could not be solved this way within a computation time of 12.5 hours.

Since an alternative to an exact computation of the bounds by means of column generation is the use of subgradient optimization, as in the case of CAPLOC, we also included results obtained with a Lagrangean heuristic based on  $Z_D^U$  and subgradient optimization. Table 3 shows the results obtained with this method, which simply consists in the subgradient phase of the described procedure for computing  $Z_D^U$ . In this table, UB% and LB% denote the percentage deviation of the computed upper and lower bound from optimality, Iter is the number of subgradient steps performed,  $T_H$  the time spend on computing feasible solutions, and  $T_{Tot}$  the total computation time in seconds. As Table 3 shows, the Lagrangean heuristic is very fast. However, the lower and upper bounds computed by this method are only of medium quality and can be worse in the case of relatively “loose” capacity constraints. As the results obtained with CAPLOC illustrate, this bound quality may be far insufficient for the purposes of computing optimal solutions for larger problem instances within a branch-and-bound procedure.

Computational results for the described procedure for computing  $Z_D^U$  are summarized in Table 4. In this table,  $It_{LR}$  and  $It_M$  denotes the number of subproblems and master problems solved, respectively;  $Col_A$  is the number of columns in the last master problem, and  $Col_{Tot}$  is the total number of columns generated;  $T_{LR}$ ,  $T_H$ , and  $T_M$  denote the computation times required for solving the subproblems, the transportation problems and the master problems, respectively;  $T_{Tot}$  is the total computation time in seconds. Compared to the subgradient procedure (Table 3), the column generation approach contributed to a significant improve in the lower bound (0.28% on average compared to 0.33% on average) and a considerable improve in the upper bound (0.24% compared to 0.98%). Partly, the improve in the upper bound is simply due to the larger number of transportation problems solved; on the other hand, Lagrangean heuristics usually produce better feasible solutions the better the Lagrangean multipliers are. Compared to the lower bound produced by the linear programming approach (column LP% in Table 2), the bound  $Z_D^U$  is on average better than this LP-bound. Furthermore, the proposed column generation method for computing  $Z_D^U$  consumes less computation time than the computation of a bound based on the linear relaxation and additional cutting planes (compare columns  $T_{Tot}$  in Table 4 and  $T_{LP}$  in Table 2). 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  $T_{Tot}$  in Table 4 and  $T_{SLP}$  in Table 2). This indicates that this bounding procedure can be useful in the framework of a branch-and-bound 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.

Table 5 shows the results obtained with the column generation procedure for computing  $Z_C$ . As can be seen from Table 5, the approach for computing  $Z_C$  produces very strong lower and upper bounds. The lower bound deviates only by 0.18% on average from the optimum value  $Z$ . The percentage deviation of the upper bound from optimality amounts to only 0.01% on average. Thus, in almost all cases an optimal solution was obtained in this way. Even in the case of tight capacity constraints, the quality of the bounds is very good. The computational effort required for computing  $Z_C$ , however, is substantial. The APLP (25) is usually far easier to solve than the CFLP, nevertheless the APLP remains a difficult, strongly NP-hard problem. The effort required for computing  $Z_C$  is, therefore, relatively large, although the analytic center cutting plane method showed a good convergence behaviour in this case and succeeded in keeping the required number of calls to the oracle satisfactorily small. For the smaller problems (less

**Table 3 Results subgradient procedure**

| Size                       | LB%  | UB%  | Iter | $T_H$ | $T_{Tot}$ |
|----------------------------|------|------|------|-------|-----------|
| $r = \sum_j s_j/d(l) = 3$  |      |      |      |       |           |
| 100×100                    | 0.07 | 0.00 | 105  | 0.3   | 1.0       |
| 200×100                    | 0.24 | 0.02 | 105  | 1.0   | 2.1       |
| 200×200                    | 0.08 | 0.00 | 105  | 1.3   | 3.5       |
| 500×100                    | 0.43 | 0.45 | 103  | 8.4   | 10.9      |
| 500×200                    | 0.17 | 0.02 | 106  | 12.7  | 17.7      |
| max                        | 0.68 | 1.02 | 110  | 17.5  | 23.2      |
| mean                       | 0.20 | 0.10 | 105  | 4.7   | 7.0       |
| $r = \sum_j s_j/d(l) = 5$  |      |      |      |       |           |
| 100×100                    | 0.22 | 0.13 | 105  | 0.2   | 0.9       |
| 200×100                    | 0.48 | 0.37 | 104  | 1.4   | 2.6       |
| 200×200                    | 0.12 | 0.06 | 105  | 1.6   | 4.1       |
| 500×100                    | 0.63 | 2.05 | 101  | 8.2   | 10.4      |
| 500×200                    | 0.42 | 0.44 | 103  | 10.4  | 15.5      |
| max                        | 0.81 | 3.55 | 105  | 13.4  | 19.7      |
| mean                       | 0.37 | 0.61 | 104  | 4.3   | 6.7       |
| $r = \sum_j s_j/d(l) = 10$ |      |      |      |       |           |
| 100×100                    | 0.30 | 0.00 | 105  | 0.1   | 0.7       |
| 200×100                    | 0.57 | 1.76 | 101  | 1.2   | 2.1       |
| 200×200                    | 0.24 | 0.01 | 106  | 0.7   | 3.0       |
| 500×100                    | 0.45 | 6.07 | 101  | 8.4   | 10.5      |
| 500×200                    | 0.58 | 3.34 | 100  | 9.8   | 14.3      |
| max                        | 0.83 | 8.31 | 110  | 10.3  | 14.8      |
| mean                       | 0.43 | 2.23 | 103  | 4.0   | 6.1       |
| Total                      |      |      |      |       |           |
| max                        | 0.83 | 8.31 | 110  | 17.5  | 23.2      |
| mean                       | 0.33 | 0.98 | 104  | 4.4   | 6.6       |

than size  $500 \times 100$ ), the approach is of course senseless. For such problems the computation of an optimal solution using CAPLOC has taken less time than the computation of  $Z_C$ . However, the difficulties encountered when solving the largest problem instances by means of CAPLOC (see Table 1) clearly show, that a (stable) column generation method for computing  $Z_C$  has to be taken into account as an alternative to a (stable) column generation method based on  $Z_D^U$  for solving such types of problems in a branch-and-bound framework.

Table 6 shows the results obtained for the procedure based on partitioning the plant set  $J$ .

In addition to the columns of Table 5,  $|Q|$  denotes the (average) number of generated plant subsets, and  $Col_{Tot}$  is the total number of generated columns, that is  $\sum_{q \in Q} \tilde{T}_q$ . The results shown in Table 6 contradict what could be expected: On average, the best lower and also upper bounds have been obtained for the test problems with smallest capacity tightness index  $r$ , although the bound  $Z_D^T$  does not make use of the aggregate capacity constraint (T) and should, therefore, be better in the case of loose capacity constraints. This unexpected behavior of the procedure was also observed in an experiment with three single very large test problems of size  $1000 \times 500$  (see Table 7). A possible explanation is that the heuristic for decomposing the plant set generates more subsets for problems with tight capacities, since in this case more plants are open in optimal solutions to the Lagrangean subproblem (33). At first sight, a small number of

**Table 4 Results procedure for computing  $Z_D^U$** 

| Size                       | LB%  | UB%  | It <sub>LR</sub> | It <sub>M</sub> | Col <sub>A</sub> | Col <sub>Tot</sub> | T <sub>LR</sub> | T <sub>H</sub> | T <sub>M</sub> | T <sub>Tot</sub> |
|----------------------------|------|------|------------------|-----------------|------------------|--------------------|-----------------|----------------|----------------|------------------|
| $r = \sum_j s_j/d(l) = 3$  |      |      |                  |                 |                  |                    |                 |                |                |                  |
| 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             |
| $r = \sum_j s_j/d(l) = 5$  |      |      |                  |                 |                  |                    |                 |                |                |                  |
| 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             |
| $r = \sum_j s_j/d(l) = 10$ |      |      |                  |                 |                  |                    |                 |                |                |                  |
| 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             |

plant subsets should result in a strong bound  $Z_D^T$ . However, in the case of a larger number of not too small plant subsets, more constraints of type (U) have a chance to be violated by solutions to the Lagrangean subproblem (33). Regarding the quality of the lower and upper bounds, the procedure computed lower bounds which are on the average slightly worse than  $Z_D^U$  but better than the LP-bound in Table 2, while the upper bound is significantly better than in the case of  $Z_D^U$  but worse than the upper bound obtained from the procedure for computing  $Z_C$ . As the results in Table 6 show, the bound  $Z_D^T$  is usually better in the case of large problems than in the case of small problems. If only the test problems of size  $500 \times 100$  or larger are considered, the bound  $Z_D^T$  even improves the bound  $Z_C$  (0.23 % vs. 0.27 % average deviation from an optimal value  $Z$  for problems of size  $500 \times 100$ , and 0.22 % vs. 0.26 % for problems of size  $500 \times 200$ ). The computational effort required to compute  $Z_D^T$ , however, is very large, even larger than the effort required for computing  $Z_C$ . Also the variation in the times spent on computing  $Z_D^T$  is substantial: For one problem of size  $500 \times 100$  and ratio  $r = 3$ , the total computation time was 27.8 hours, while for the other problems of this class the computation time was only 1.8 hours on average. For one problem of size  $500 \times 200$  and ratio  $r = 10$  even 66.7 hours of computation time were consumed compared to an average computation time of 4.5 hours for other problems of

**Table 5** Results procedure for computing  $Z_C$ 

| Size                       | LB%  | UB%  | lt <sub>LR</sub> | lt <sub>M</sub> | T <sub>LR</sub> | T <sub>H</sub> | T <sub>M</sub> | T <sub>Tot</sub> |
|----------------------------|------|------|------------------|-----------------|-----------------|----------------|----------------|------------------|
| $r = \sum_j s_j/d(l) = 3$  |      |      |                  |                 |                 |                |                |                  |
| 100×100                    | 0.04 | 0.00 | 122              | 122             | 35.5            | 0.5            | 2.0            | 38.1             |
| 200×100                    | 0.20 | 0.00 | 131              | 131             | 73.9            | 1.7            | 3.3            | 78.9             |
| 200×200                    | 0.03 | 0.00 | 229              | 228             | 139.3           | 4.5            | 17.6           | 161.4            |
| 500×100                    | 0.33 | 0.03 | 161              | 161             | 835.6           | 9.4            | 19.9           | 865.1            |
| 500×200                    | 0.15 | 0.00 | 290              | 290             | 1273.2          | 34.0           | 66.3           | 1373.5           |
| max                        | 0.54 | 0.15 | 326              | 326             | 1855.3          | 40.6           | 95.7           | 1967.3           |
| mean                       | 0.15 | 0.01 | 187              | 186             | 471.5           | 10.0           | 21.8           | 503.4            |
| $r = \sum_j s_j/d(l) = 5$  |      |      |                  |                 |                 |                |                |                  |
| 100×100                    | 0.11 | 0.00 | 91               | 90              | 51.5            | 0.3            | 1.2            | 53.2             |
| 200×100                    | 0.28 | 0.00 | 106              | 106             | 330.6           | 1.2            | 5.0            | 336.8            |
| 200×200                    | 0.09 | 0.00 | 164              | 164             | 432.7           | 2.8            | 9.4            | 444.9            |
| 500×100                    | 0.31 | 0.01 | 122              | 122             | 2332.9          | 5.0            | 11.5           | 2349.5           |
| 500×200                    | 0.33 | 0.04 | 267              | 267             | 11487.0         | 22.6           | 51.1           | 11560.9          |
| max                        | 0.48 | 0.14 | 314              | 314             | 25128.4         | 35.2           | 73.8           | 25209.3          |
| mean                       | 0.22 | 0.01 | 150              | 150             | 2927.0          | 6.4            | 15.6           | 2949.0           |
| $r = \sum_j s_j/d(l) = 10$ |      |      |                  |                 |                 |                |                |                  |
| 100×100                    | 0.11 | 0.00 | 37               | 36              | 4.6             | 0.1            | 0.1            | 4.7              |
| 200×100                    | 0.21 | 0.04 | 91               | 51              | 114.8           | 0.5            | 0.9            | 116.2            |
| 200×200                    | 0.14 | 0.00 | 96               | 96              | 103.5           | 0.9            | 0.7            | 105.2            |
| 500×100                    | 0.17 | 0.00 | 125              | 125             | 798.5           | 3.1            | 5.3            | 807.0            |
| 500×200                    | 0.29 | 0.02 | 190              | 190             | 9853.1          | 10.0           | 30.6           | 9893.9           |
| Max                        | 0.56 | 0.18 | 296              | 296             | 16382.0         | 14.0           | 39.5           | 16427.4          |
| mean                       | 0.18 | 0.01 | 108              | 99              | 2174.9          | 2.9            | 7.5            | 2185.4           |
| Total                      |      |      |                  |                 |                 |                |                |                  |
| max                        | 0.56 | 0.18 | 326              | 326             | 25128.4         | 40.6           | 95.7           | 25209.3          |
| mean                       | 0.18 | 0.01 | 148              | 145             | 1857.8          | 6.4            | 15.0           | 1879.3           |

this type. Nevertheless, the bound  $Z_D^T$  improved the usually very strong bound  $Z_C$  for a variety of the larger problem instances. In order to test the heuristic for decomposing the plant set, the bound  $Z_D^T$  was also computed by randomly partitioning the plant set into subsets of a random cardinality between 10 and 20 plants. The percentage deviation from optimality of the lower bound obtained this way amounted to 0.27%, 0.47% and 0.55% on average for test problems with capacity index  $r = 3$ ,  $r = 5$  and  $r = 10$ , respectively; the upper bound deviated from an optimal solution by 0.02% ( $r = 3$ ), 0.19% ( $r = 5$ ) and 0.19% ( $r = 10$ ) on average. Compared to the proposed heuristic for decomposing the plant set (see Table 6), a random partitioning led, therefore, to a significant deterioration in the lower and upper bound. Furthermore, the random partitioning approach no more showed the tendency to provide stronger bounds for larger than for smaller problem instances.

## 6 Conclusions

In this paper, important Lagrangean bounds for the CFLP were computed exactly by means of different stabilized column generation schemes. Furthermore, a new lower bound based on

**Table 6** Results procedure for computing  $Z_D^T$ 

| Size                       | LB%  | UB%  | Q  | lt <sub>LR</sub> | lt <sub>M</sub> | Col <sub>Tot</sub> | T <sub>LR</sub> | T <sub>H</sub> | T <sub>M</sub> | T <sub>Tot</sub> |
|----------------------------|------|------|----|------------------|-----------------|--------------------|-----------------|----------------|----------------|------------------|
| $r = \sum_j s_j/d(l) = 3$  |      |      |    |                  |                 |                    |                 |                |                |                  |
| 100×100                    | 0.32 | 0.00 | 5  | 62               | 62              | 305                | 32.0            | 0.3            | 5.4            | 37.7             |
| 200×100                    | 0.23 | 0.00 | 4  | 67               | 67              | 290                | 267.7           | 0.7            | 16.8           | 285.5            |
| 200×200                    | 0.12 | 0.00 | 9  | 116              | 115             | 987                | 353.7           | 2.4            | 44.1           | 400.7            |
| 500×100                    | 0.27 | 0.18 | 6  | 111              | 111             | 609                | 24977.0         | 3.8            | 297.6          | 25279.9          |
| 500×200                    | 0.09 | 0.02 | 9  | 100              | 100             | 949                | 2514.2          | 9.6            | 154.8          | 2680.3           |
| max                        | 0.70 | 0.43 | 12 | 184              | 184             | 1472               | 99461.4         | 15.2           | 496.9          | 99963.8          |
| mean                       | 0.20 | 0.04 | 7  | 91               | 91              | 628                | 5628.9          | 3.3            | 103.7          | 5736.8           |
| $r = \sum_j s_j/d(l) = 5$  |      |      |    |                  |                 |                    |                 |                |                |                  |
| 100×100                    | 0.63 | 0.02 | 4  | 71               | 71              | 249                | 52.5            | 0.2            | 6.9            | 59.7             |
| 200×100                    | 0.41 | 0.24 | 4  | 65               | 64              | 230                | 859.5           | 0.5            | 26.0           | 886.2            |
| 200×200                    | 0.15 | 0.04 | 6  | 128              | 128             | 774                | 385.9           | 1.7            | 54.1           | 442.0            |
| 500×100                    | 0.31 | 0.14 | 5  | 139              | 138             | 628                | 6772.9          | 2.8            | 355.2          | 7131.7           |
| 500×200                    | 0.26 | 0.34 | 7  | 116              | 115             | 813                | 5657.0          | 7.1            | 217.9          | 5884.1           |
| max                        | 1.05 | 1.01 | 9  | 180              | 180             | 992                | 19724.5         | 9.0            | 666.7          | 20393.4          |
| mean                       | 0.35 | 0.16 | 5  | 104              | 103             | 539                | 2745.6          | 2.5            | 132.0          | 2880.7           |
| $r = \sum_j s_j/d(l) = 10$ |      |      |    |                  |                 |                    |                 |                |                |                  |
| 100×100                    | 0.98 | 0.00 | 2  | 83               | 82              | 183                | 26.2            | 0.1            | 9.1            | 35.4             |
| 200×100                    | 0.43 | 0.06 | 2  | 127              | 86              | 173                | 897.2           | 0.5            | 38.7           | 936.5            |
| 200×200                    | 0.50 | 0.00 | 3  | 112              | 112             | 382                | 235.2           | 0.4            | 70.9           | 306.6            |
| 500×100                    | 0.12 | 0.00 | 3  | 60               | 60              | 236                | 6465.3          | 0.9            | 404.0          | 6870.4           |
| 500×200                    | 0.30 | 0.29 | 4  | 157              | 157             | 537                | 59076.4         | 3.5            | 1647.1         | 60727.9          |
| max                        | 1.45 | 0.57 | 5  | 200              | 198             | 735                | 236956.2        | 4.1            | 2425.6         | 239386.4         |
| mean                       | 0.47 | 0.07 | 3  | 108              | 99              | 302                | 13340.1         | 1.1            | 433.9          | 13775.4          |
| Total                      |      |      |    |                  |                 |                    |                 |                |                |                  |
| max                        | 1.45 | 1.01 | 12 | 200              | 198             | 1472               | 236956.2        | 15.2           | 2425.6         | 239386.4         |
| mean                       | 0.34 | 0.09 | 5  | 101              | 98              | 490                | 7238.2          | 2.3            | 223.2          | 7464.3           |

partitioning the set of potential plant sites was proposed.

For the conventional relaxation of the demand constraints a mixture of a “weighted” Dantzig-Wolfe decomposition method and subgradient optimization, preceded by a pure subgradient optimization phase, gave fairly good results. The computation of optimal solutions to the master problem by means of this method required less computation time than the determination of a bound based on the linear relaxation and additional cutting planes. For large test problems the column generation procedure even required less computation time than a simplex algorithm for computing the strong LP-bound. A branch-and-price algorithm based on this column generation method should, therefore, give good results for problem instances with relatively tight capacity constraints.

Very strong lower and upper bounds for the CFLP are obtainable from a Lagrangean relaxation of the capacity constraints. In order to solve the master problem, an interior point method is suitable. Due to the relatively small size of the restricted master problems, more effort can be spent on computing good Lagrangean multipliers from the localization set in order to avoid too many calls to the difficult oracle. Although the effort required to compute the bound is relatively large, a branch-and-price method based on this approach has to be taken into account for large problem instances with relatively loose capacity constraints.

**Table 7** Bound  $Z_D^T$  for 3 single problem instances of size  $1000 \times 500$ 

|          | Gap% <sup>a</sup> | It <sub>LR</sub> | It <sub>M</sub> | Col <sub>Tot</sub> | T <sub>LR</sub> | T <sub>H</sub> | T <sub>M</sub> | T <sub>Tot</sub> |
|----------|-------------------|------------------|-----------------|--------------------|-----------------|----------------|----------------|------------------|
| $r = 3$  | 0.06              | 129              | 129             | 3483               | 21769.8         | 168.7          | 768.7          | 22735.1          |
| $r = 5$  | 0.10              | 167              | 166             | 2004               | 43286.1         | 66.4           | 1392.1         | 44754.3          |
| $r = 10$ | 0.40              | 234              | 233             | 1872               | 59007.7         | 49.4           | 2927.3         | 61997.1          |

<sup>a</sup>Gap% =  $100 \cdot (UB - LB)/LB$

For a number of large problem instances the lower and upper bounds obtained by means of the partitioning approach even improved the strong bounds resulting from a Lagrangean relaxation of the capacity constraints. Despite the large computation times, the partitioning procedure can, therefore, be useful in the case of very large instances which are not tractable by algorithms based on one of the other bounding schemes considered. However, additional partitioning heuristics have to be devised in order to improve the results of the proposed heuristic for decomposing the plant set in the case of problem instances with relatively large plant capacities. Finally, the basic idea of the partitioning approach is also applicable to other problems of the assignment type, as e.g. uncapacitated facility location problems,  $p$ -median problems, bin packing problems, fixed-charge transportation problems and generalized assignment problems.

## Acknowledgements

We thank M. Guignard-Spielberg for providing us with a FORTRAN code of CAPLOC. Thanks are also due to J.-P. Vial and J. Gondzio for the provision of the library ACCPM.

## References

- P. Carraresi, A. Frangioni, M. Nonato. 1995. Applying bundle methods to the optimization of polyhedral functions: An applications-oriented development. *Ricerca Operativa* **25** 5–49.
- B. Chen, M. Guignard. 1998. Polyhedral analysis and decompositions for capacitated plant location-type problems. *Discrete Appl. Math.* **82** 79–91.
- G. Cornuejols, R. Sridharan, J.-M. Thizy. 1991. A comparison of heuristics and relaxations for the capacitated plant location problem. *European J. Oper. Res.* **50** 280–297.
- CPLEX Division, ILOG Inc. 1997. Using the CPLEX callable library.
- G. B. Dantzig, P. Wolfe. 1960. Decomposition principle for linear programs. *Oper. Res.* **8** 101–111.
- O. du Merle, D. Villeneuve, J. Desrosiers, P. Hansen. 1999. Stabilized column generation. *Discrete Math.* **194** 229–237.
- A. Frangioni, G. Gallo. 1999. A bundle type dual-ascent approach to linear multi-commodity min cost flow problems. *Inform. J. Comput.* **11** 370–393.
- J.-L. Goffin, A. Haurie, J.-P. Vial. 1992. Decomposition and nondifferentiable optimization with the projective algorithm. *Management Sci.* **38** 284–302.
- J.-L. Goffin, A. Haurie, J.-P. Vial, D. L. Zhu. 1993. Using central prices in the decomposition of linear programs. *European J. Oper. Res.* **64** 593–409.
- J.-L. Goffin, J.-P. Vial. 1999. Shallow, deep and very deep cuts in the analytic center cutting plane method. *Math. Programming* **84** 89–103.
- J. Gondzio, O. du Merle, R. Sarkissian, J.-P. Vial. 1996. ACCPM - a library for convex optimization based on an analytic center cutting plane method. *European J. Oper. Res.* **94** 206–211.
- J. Gondzio, R. Sarkissian. 1996. Column generation with a primal-dual method. Technical Report 1996:6. LogiLab, HEC, Section of Management Studies. University of Geneva.

<http://ecolu-info.unige.ch/~logilab/reports>.

- M. Guignard, S. Zhu. 1994. A two-phase dual algorithm for solving Lagrangean duals in mixed integer programming. Report 94-10-03. Operations and Information Management Department. University of Pennsylvania, The Wharton School.
- J. E. Kelley. 1960. The cutting-plane method for solving convex programs. *J. SIAM* **8** 703–712.
- A. Klose. 1998. A branch and bound algorithm for an uncapacitated facility location problem with a side constraint. *Internat. Trans. Oper. Res.* **5** 155–168.
- . 1999. An LP-based heuristic for two-stage capacitated facility location problems. *J. Oper. Res. Soc.* **50** 157–166.
- G. A. Kochmann, C. J. McCallum. 1981. Facility location models for planning a transatlantic communications network. *European J. Oper. Res.* **6** 205–211.
- C. Lemaréchal. 1989. Nondifferentiable optimization. In G. L. Nemhauser, A. H. G. Rinnooy Kan, M. J. Todd, eds., *Optimization*. vol. 1 of *Handbooks in Operations Research and Management Science*. pp. 529–572. North-Holland, Amsterdam.
- R. E. Marsten, W. W. Hogan, J. W. Blankenship. 1975. The Boxstep method for large-scale optimization. *Oper. Res.* **23** 389–405.
- S. Martello, D. Pisinger, P. Toth. 1999. Dynamic programming and strong bounds for the 0-1 Knapsack problem. *Management Sci.* **45** 414–424.
- R. K. Martinson, J. Tind. 1999. An interior point method in Dantzig-Wolfe decomposition. *Comput. Oper. Res.* **26** 1195–1216.
- A. Mirzaian. 1985. Lagrangian relaxation for the star-star concentrator location problem: Approximation algorithm and bounds. *Networks* **15** 1–20.
- P. Neame, N. Boland, D. Ralph. 1998. A unifying framework for column generation stabilization methods. Working paper. Department of Mathematics. University of Melbourne. <ftp://ftp.hpc.uh.edu/pub/ipco98/neame.ps>.
- Y. Pochet, L. A. Wolsey. 1988. Lot-size models with backlogging: Strong reformulations and cutting planes. *Math. Programming* **40** 317–335.
- C. Ryu, M. Guignard. 1992. An efficient algorithm for the capacitated plant location problem. Working Paper 92-11-02. Decision Sciences Department. University of Pennsylvania, The Wharton School.
- R. Sridharan. 1995. The capacitated plant location problem. *European J. Oper. Res.* **87** 203–213.
- T. J. Van Roy. 1986. A cross decomposition algorithm for capacitated facility location. *Oper. Res.* **34** 145–163.
- P. Wentges. 1997. Weighted Dantzig-Wolfe decomposition for linear mixed-integer programming. *Internat. Trans. Oper. Res.* **4** 151–162.