ELSEVIER

Discrete Mathematics 194 (1999) 229–237

DISCRETE MATHEMATICS

Communication

Stabilized column generation

Olivier du Merle [a], Daniel Villeneuve [b], Jacques Desrosiers [c], Pierre Hansen [c,*]

[a] *France Télécom 92 794 Issy les Mx, Cedex 9, France*

[b] *GERAD and École Polytechnique de Montréal, Québec H3C 3A7, Canada*

[c] *GERAD and École des Hautes Études Commerciales, Montréal, Québec H3T 2A7, Canada*

Received 7 July 1998; accepted 14 July 1998

Communicated by D. de Werra

Abstract

Column generation is often used to solve large-scale optimization problems, and much research has been devoted to improve the convergence of the solution process. We focus on Kelley's algorithm, which frequently exhibits slow convergence, and propose an algorithm that stabilizes and accelerates the solution process while remaining within the linear programming framework. Preliminary numerical results, obtained on air transportation and location problems, show that the stabilized algorithm can be used to improve the solution times for difficult instances and to solve larger ones. © 1999 Published by Elsevier Science B.V. All rights reserved

Résumé

Les problèmes d'optimisation de grande taille sont souvent résolus par génération de colonnes et de nombreux travaux de recherche ont porté sur l'amélioration de la convergence du processus de résolution. Nous concentrons notre attention sur l'algorithme de Kelley, dont la convergence est fréquemment lente, et nous proposons un algorithme qui stabilise et accélère cette procédure tout en restant dans le cadre de la programmation linéaire. Les premiers résultats numériques, obtenus pour des problèmes de transport aérien et de localisation, montrent que l'algorithme de stabilisation permet de diminuer les temps de résolution et aussi de résoudre des problèmes de plus grande taille. © 1999 Published by Elsevier Science B.V. All rights reserved

* Correspondence address. GERAD-HEC, 3000, chemin de la Côte-Ste-Catherine, Montréal, Québec, Canada H3T 2A7. E-mail: pierreh@crt.umontreal.ca.

PII S0012-365X(98)00213-1

1. Introduction

Consider a feasible and bounded linear program P and its dual D:

$$(P)\quad \begin{array}{ll} \min & c^{\mathrm{T}}x \\ \text{s.t.} & Ax=b, \\ & x\geqslant 0, \end{array} \qquad\qquad (D)\quad \begin{array}{ll} \max & b^{\mathrm{T}}\pi, \\ \text{s.t.} & A^{\mathrm{T}}\pi\leqslant c. \end{array}$$

When P has many more variables than D, as when applying Benders [2] or Dantzig–Wolfe [5] decomposition, it is usually solved by column generation. This amounts to applying Kelley's algorithm [16] to D and convergence is often slow. Such undesirable behavior is also observed when one tries to prove optimality of a degenerate solution of P. In other words, many iterations are necessary to obtain a good polyhedral approximation of the domain of D. One way to overcome degeneracy is to perturb P by adding bounded surplus and slack variables:

$$(P_\varepsilon)\quad \min_{(x,y_-,y_+)\geqslant 0}\{c^{\mathrm{T}}x\colon Ax-y_-+y_+=b,\ y_-\leqslant\varepsilon_-,\ y_+\leqslant\varepsilon_+\}.$$

Alternatively, one could also use exact penalties in l_1 norm [10], as in nonlinear programming, to narrow the domain of D. Solving P is then equivalent to solving the following problem, with a suitably chosen scalar $\delta\geqslant 0$:

$$(P_\delta)\quad \min_{x\geqslant 0} c^{\mathrm{T}}x+\delta\|Ax-b\|_1=\min_{(x,y_-,y_+)\geqslant 0}\{c^{\mathrm{T}}x+\delta y_-+\delta y_+\colon Ax-y_-+y_+=b\}.$$

In the above formulation, one can observe that the dual variables associated with the equality constraints in problem P_δ are restricted to the box $[-\delta e,\delta e]$, where e is a vector of ones. Centering the box at an estimated dual vector different from the origin, problem P_δ would correspond to one of the problems solved in the iterative process of the BOXSTEP method [19].

Many nonlinear approaches have been proposed to overcome the bad behavior of Kelley's algorithm: augmented Lagrangian and bundle methods [15], central cutting plane methods [23] (Shor ellipsoids, volumetric and analytic centers),.... The originality of the present paper is to define a stabilization scheme which remains within the linear programming framework and to demonstrate its efficiency. We propose to stabilize and accelerate the column generation procedure by merging the perturbation and exact penalty methods. Section 2 presents a problem $\tilde{P}$ which includes both problems P_ε and P_δ as special cases. Section 3 describes, within the framework of column generation, an algorithm which updates vector parameters $(\varepsilon_-,\varepsilon_+)$ and (δ_-,δ_+), defined below, in order to obtain rapidly an exact solution of P. Section 4 concludes with three case studies where stabilized column generation leads to large speedup factors or to solution of much larger instances than done before.

2. Problem $\tilde{P}$

Define the primal problem $\tilde{P}$ and its dual $\tilde{D}$ as follows:

$$
(\tilde{P}) \quad \begin{array}{ll} \min & c^{\mathrm{T}}\tilde{x} - \delta_-^{\mathrm{T}} y_- + \delta_+^{\mathrm{T}} y_+ \\ \text{s.t.} & A\tilde{x} - y_- + y_+ = b, \\ & y_- \leqslant \varepsilon_-, \\ & y_+ \leqslant \varepsilon_+, \\ & \tilde{x}, y_-, y_+ \geqslant 0, \end{array}
\qquad
(\tilde{D}) \quad \begin{array}{ll} \max & b^{\mathrm{T}}\tilde{\pi} - \varepsilon_-^{\mathrm{T}} w_- - \varepsilon_+^{\mathrm{T}} w_+ \\ \text{s.t.} & A^{\mathrm{T}}\tilde{\pi} \leqslant c, \\ & -\tilde{\pi} - w_- \leqslant -\delta_-, \\ & \tilde{\pi} - w_+ \leqslant \delta_+, \\ & w_-, w_+ \geqslant 0. \end{array}
$$

In the primal problem $\tilde{P}$, y_- and y_+ are vectors of surplus and slack variables, with upper bounds ε_- and ε_+, respectively. These variables are penalized in the objective function by vectors δ_- and δ_+, respectively. In the dual problem $\tilde{D}$, the last two constraints may be rewritten as $\delta_- - w_- \leqslant \tilde{\pi} \leqslant \delta_+ + w_+$, which amounts to penalizing dual variables $\tilde{\pi}$ when they lie outside of the interval $[\delta_-, \delta_+]$. On the contrary, in the BOXSTEP method, $\varepsilon_- = \varepsilon_+ = \infty$ and dual variables cannot lie outside of the chosen intervals. In the above model, even though the chosen intervals may be far from an optimal dual solution of P, taking parameters ε_- and ε_+ small enough allows problem $\tilde{P}$ to have primal and dual solutions close to the solutions of problem P.

Denote by $x^\star$, $\pi^\star$, $(\tilde{x}^\star, y_-^\star, y_+^\star)$ and $(\tilde{\pi}^\star, w_-^\star, w_+^\star)$ optimal solutions of P, D, $\tilde{P}$ and $\tilde{D}$, respectively, and by $v(\cdot)$ the value of an optimal solution of problem $(\cdot)$. Then, $P \equiv \tilde{P}$ (i.e., $y_-^\star = y_+^\star = 0$) if one of the following two conditions is met: (i) $\varepsilon_- = \varepsilon_+ = 0$, (ii) $\delta_- < \tilde{\pi}^\star < \delta_+$. Moreover, (iii) $v(\tilde{P}) \leqslant b^{\mathrm{T}}\tilde{\pi}^\star \leqslant v(P)$. Conditions (i) and (ii) provide stopping criteria for the algorithm described in the next section. Inequality (iii) shows that while $\tilde{P}$ is a relaxation of P, $b^{\mathrm{T}}\tilde{\pi}^\star$ can be a better lower bound than $v(\tilde{P})$ on $v(P)$, which may be used when embedding this algorithm in branch&bound procedures for mixed integer linear programming problems.

3. Algorithm

Let the linear program P and its dual D be defined as in Section 1. Problem P may be solved by the classical column generation method presented on the left-hand side of Fig. 1. At iteration k, a restricted linear program is given by P^k: $\min\{c^{k\mathrm{T}}x^k\colon A^k x^k = b, x^k \geqslant 0\}$ and its dual by D^k: $\max\{b^{\mathrm{T}}\pi^k\colon A^{k\mathrm{T}}\pi^k \leqslant c^k\}$. The following procedures need to be defined:

- $(x^k; \pi^k) \leftarrow$ `optimizer(`P^k`)`, where x^k and π^k are optimal solutions of P^k and D^k, respectively;
- $(\hat{a}; \hat{c}) \leftarrow$ `oracle(`π^k`)`, where $\hat{a}$ is a column of A and $\hat{c}$ is its corresponding component in c, such that the reduced cost $\hat{c} - \hat{a}^{\mathrm{T}}\pi^k$ is minimum over all columns of A.

The use of perturbation and penalties stabilizes the solution process in the dual space and usually leads to a reduction in the number of iterations necessary for obtaining an optimal solution to P. At iteration k, define the restricted linear program

Initialization: A^0, c^0	Initialization: $A^0, c^0, \delta^0, \epsilon^0$
Assumptions: P^0 feasible and bounded	Assumptions: $\tilde{P}^0$ feasible and bounded

```
k ← 0, end ← false                     k ← 0, end ← false
DO                                     DO
  (x^k; π^k) ← optimizer(P^k)            (x̃^k, y_-, y_+; π̃^k, w_-, w_+) ← optimizer(P̃^k)
  (â; ĉ) ← oracle(π^k)                   (â; ĉ) ← oracle(π̃^k)
  IF â^T π^k ≤ ĉ                         IF â^T π̃ ≤ ĉ and y_- = y_+ = 0
    x* ← x^k, end ← true                   x* ← x̃^k, end ← true
  ELSE                                   ELSE
    (A^{k+1}; c^{k+1}) ← (A^k, â; c^k, ĉ)  (A^{k+1}; c^{k+1}) ← (A^k, â; c^k, ĉ)
                                           δ^{k+1} ← update-δ(k)
                                           ε^{k+1} ← update-ε(k)
    k ← k + 1                              k ← k + 1
WHILE ¬ end                            WHILE ¬ end
```

Fig. 1. Generic column generation algorithms.

$\tilde{P}^k$: $\min\{c^{k\mathrm{T}}\tilde{x}^k - \delta_-^{k\mathrm{T}} y_- + \delta_+^{k\mathrm{T}} y_+ \colon A^k\tilde{x}^k - y_- + y_+ = b,\ \tilde{x}^k \geqslant 0, 0 \leqslant y_- \leqslant \varepsilon_-^k, 0 \leqslant y_+ \leqslant \varepsilon_+^k\}$ and its dual $\tilde{D}^k$. Besides optimizer and `oracle` defined above, one needs the procedures

- $\delta^{k+1} \leftarrow$ `update-`$\delta(k)$ and
- $\varepsilon^{k+1} \leftarrow$ `update-`$\varepsilon(k)$,

where the parameter k in $\delta(k)$ and $\varepsilon(k)$ indicates all the solution process history until iteration k. For example, the update procedures may involve the use of the primal and dual solutions of the previously solved restricted problems. A generic algorithm is presented on the right-hand side of Fig. 1. This algorithm, which may be interpreted as a bundle method in l_1 norm, differs from the classical one in being able to stabilize the dual variables $\tilde{\pi}^k$ with the linear penalties $(\varepsilon_-^k, \varepsilon_+^k)$ that take effect outside the range $[\delta_-^k, \delta_+^k]$. Obviously, it is of great importance to develop efficient strategies for adjusting these parameters.

One natural strategy for updating δ_- and δ_+ is to set them to the current dual solution: $\delta_-^{k+1} = \delta_+^{k+1} = \tilde{\pi}^k$ (or to use the variant $\delta_-^{k+1} + \xi = \delta_+^{k+1} - \xi = \tilde{\pi}^k$ with $\xi > 0$, taking into account the uncertainty about the estimation of $\pi^\star$ by $\tilde{\pi}^k$). However, it may be possible to estimate the quality of $\tilde{\pi}^k$ as an approximation of $\pi^\star$, in which case the update is performed only if $\tilde{\pi}^k$ is the best known estimate of $\pi^\star$. For example, P has the following structure when derived from Dantzig–Wolfe decomposition:

$$\min\left\{\sum_{i=1}^{p} c_i^{\mathrm{T}} x_i \colon \sum_{i=1}^{p} A_i x_i = b_0,\ e_i^{\mathrm{T}} x_i = b_i,\ x_i \geqslant 0,\ i \in \{1,\ldots,p\}\right\},$$

where e_i is a vector of ones of the same dimension as x_i. An estimate of the quality for $\tilde{\pi}^k$ can be obtained via the computation of a lower bound on P [18],

$$v(P;\tilde{\pi}^k) = b_0^{\mathrm{T}}\tilde{\pi}_0^k + \sum_{i=1}^{p} b_i(\hat{c}_i - \hat{a}_i^{\mathrm{T}}\tilde{\pi}_0^k),$$

where $\tilde{\pi}_0^k$ is the vector of components of $\tilde{\pi}^k$ associated with the first set of constraints $\sum_{i=1}^{p} A_i x_i = b_0$. In the case of many convexity constraints (i.e., $p>1$), the oracle must return one $(column, cost)$-pair for each set i $(1 \leqslant i \leqslant p)$ in order to compute the lower bound $v(P; \tilde{\pi}^k)$.

One natural strategy for updating ε_- and ε_+ is to decrease all values if $\tilde{\pi}^k$ is the best-known estimate of $\pi^\star$ or if the column returned by the oracle has a non-negative reduced cost, and to increase them otherwise.

The updating strategies for both sets of parameters must together ensure finite convergence of the algorithm. The next two strategies have this property:

- After a given number of iterations, decrease ε_- and ε_+ so that they vanish in a finite number of iterations, in which case condition (i) of Section 2 is satisfied.
- After a given number of iterations, update δ_- and δ_+, with $\xi > 0$, only if the column returned by the oracle has a non-negative reduced cost, as in the BOXSTEP method. Then $\tilde{\pi}^k$ is feasible for D and we have $v(\tilde{D}^k) \leqslant v(D)$. If $v(\tilde{D}^k) = v(D)$, condition (ii) of Section 2 is satisfied upon update completion. Otherwise, $\xi > 0 \Rightarrow v(\tilde{D}^k) < v(\tilde{D}^{k+t})$, with $t>0$ denoting the number of iterations necessary for the oracle to return a column with non-negative reduced cost upon update completion. Hence, condition (ii) holds after a finite number of iterations.

Finally, even though the parameters (δ_-^0, δ_+^0) may take any values (subject to $\delta_-^0 \leqslant \delta_+^0$), a good estimate of the optimal dual variables $\pi^\star$ should be preferred over arbitrary values.

4. Applications

We have applied stabilized column generation to airline crew pairing, multisource Weber and p-median problems. In each of these applications, the method made it possible to either reduce solution time or solve larger problem instances. Potential of stabilized column generation for improved solution of extensions of these problems and many similar ones appear to be large. This holds particularly for problems where there is massive degeneracy and for which efficient heuristics for finding good primal and dual solutions are available.

Airline crew pairing problem. In this integer-programming application, rows represent flight legs to be assigned to crews and columns represent possible schedules for these crews. This problem can be formulated as a set partitioning problem. A column generation algorithm is used to find lower bounds in a branch&bound framework, the oracle consisting in a resource constrained shortest path problem [6]. Precise estimates for the optimal values of dual variables are difficult to obtain but a good approximation of the objective function value can be used to compute an average value for them. The classical column generation algorithm performs poorly, as degeneracy occurs at two levels: when solving the current linear program and also during several successive major iterations for which the added columns do not suffice to modify the objective

Table 1
Instance of an airline crew pairing problem — variations on δ^0

Parameters (δ^0_-, δ^0_+)	Main iterations	CPU time ratio `optimizer/oracle`	CPU time (s)	Speedup factor
$(-\infty, \infty)$	433	1.037	1491.0	1.00
$[-50, \infty)$	440	1.444	1985.3	0.75
$[0, 100]$	157	0.521	267.6	5.57
$[50, 100]$	130	0.301	201.2	7.41

function value. Typically, using stabilized column generation reduces solution time by a factor ranging from 2 to 10.

As an example, we used the stabilized algorithm to solve the linear relaxation of a 986-leg instance for a regional carrier [7]: the number of column generation iterations is reduced by a factor of 3.33 and the CPU time by a factor of 7.41. On the one hand, the strategy for the `update-`δ procedure is to update the vector parameters (δ_-, δ_+) when the column generation algorithm stalls, using the following predefined sequence of embedded intervals: $[50, 100]$, $[0, 100]$, $[-50, \infty)$ and $(-\infty, \infty)$. The last interval imposes no restrictions on the dual variables and corresponds to the set partitioning formulation; the first one gives very rough estimates of the dual variables, the objective function value being around 100 000. On the other hand, the vector parameters ε_- and ε_+ are selected at random in the ranges $[9, 11]$ and $[0, 10^{-4}]$, respectively, and are kept fixed throughout the solution process. The first interval, for selecting ε_-, allows for overcovering the flight legs and simulates a set covering formulation, while the second interval, for choosing ε_+, corresponds to a small perturbation. For different starting values of (δ^0_-, δ^0_+) in the above sequence of embedded intervals, Table 1 shows the results obtained on a SUN Ultra SPARC 1300 workstation: the number of iterations, the ratio between the CPU time of the `optimizer` procedure over the CPU time of the `oracle` one, the total CPU time and the speedup factor as compared to the solution time of the set partitioning formulation. The fastest solution time is obtained from the use of the entire sequence, which confirms the effectiveness of the stabilization method. The anomaly observed when starting with the interval $[-50, \infty)$ can be explained by redundancy of this interval with the interval $(-\infty, \infty)$ and the disadvantage of having a lower bound different from $-\infty$. However, interval $[-50, \infty)$ has proven to be efficient when used in the sequence from interval $[0, 100]$ to interval $(-\infty, \infty)$.

Multisource Weber problem. This problem is a basic one in continuous location theory. It can be expressed as follows: given a set of n users, with fixed locations in the Euclidean plane, determine simultaneously the locations of p facilities in this plane in order to minimize the sum of (weighted) distances from each user to his closest facility. The case $p = 2$ can easily be solved by using optimization algorithms for difference of convex functions programs (d.-c. programming) [4] but only small instances are solved exactly for $p \geqslant 3$ (i.e., $n \leqslant 50$ for $p = 3$ [4], $n \leqslant 30$ for $p = 4, 5$ and $n = 25$ for $p = 6$ [22]). The problem can be expressed as a set partitioning problem, with columns

Table 2
Results of multisource Weber problems with $n = 1060$

Number of facilities (p)	Optimal value	CPU time (s)	Main interations	Branching nodes
10	1 249 564	18 283	824	0
40	529 660	44 470	265	10
50	453 109	11 740	304	0
100	282 536	8 314	161	0

associated with all possible subsets of users and costs corresponding to those of single facility Weber problems with these sets of users. A column generation algorithm can be used to find lower bounds in a branch&bound framework, the oracle consisting in a single facility Weber problem with limited distances [8], solved by a variant of the BSSS algorithm [14]. Problems with $n \leqslant 287$ and $p \leqslant 100$ can then be solved [17]. Stabilizing this algorithm [11] allows solution of problems with $n \leqslant 1060$ and $p \leqslant 100$ using the following strategies. A heuristic solution, often optimal or very close to the optimum, is first obtained with a variable neighborhood search heuristic [20,12,3]. Let $c_1, c_2, \dots, c_n$, denote the index sets of the corresponding user partition and $f(c_1), f(c_2), \dots, f(c_n)$ their value. Then, given $l \in c_i$, initial ranges for the dual variables are chosen by taking

$$\delta^0_{l-} = f(c_i) - f(c_i \setminus \{l\}) \quad \text{and} \quad \delta^0_{l+} = \min_{j \neq i}(f(c_j \cup \{l\}) - f(c_j)).$$

On the one hand, the strategy for the update-δ procedure is twofold:
1. set the vector parameters δ^k_- and δ^k_+ around $\tilde{\pi}^k$ if $\tilde{\pi}^k$ is the best dual solution found over all the solution process, while preserving the individual ranges between the corresponding components of δ^{k-1}_- and δ^{k-1}_+;
2. otherwise, set the vector parameters δ^k_- and δ^k_+ around the same dual solution as for δ^{k-1}_- and δ^{k-1}_+, and double the range between the corresponding components of δ^{k-1}_- and δ^{k-1}_+ if one of the ranges is found to be too small (i.e., if a range constraint becomes binding).

The range update is very rare since the heuristic solution is very close to the optimal one. On the other hand, the vector parameters ε^0_- and ε^0_+ are initialized to the right-hand-side value, i.e., vectors of ones. The strategy for the update-ε procedure is to modify ε_- and ε_+ only when the column returned by the oracle has a non-negative reduced cost, by dividing each component by 2^n where n is the number of previous such updates in the solution process. Results obtained on a SUN SPARC 10 workstation are presented in Table 2.

***p*-median problem.** This problem is a basic one in discrete location. In fact it is the discrete counterpart of the multisource Weber problem, location of facilities being restricted to a given discrete set of points. The largest problems solved in the literature have $n \leqslant 900$ users and $p \leqslant 200$ facilities [1]. A column generation algorithm for the p-median, with a very simple oracle, was proposed some time ago [9] but could

Table 3
Results of p-median problems with n=3038

Number of facilities (p)	Value of relaxation	Best known integer solution	CPU time (h)	Integrality gap
10	1 213 082.03	1 213 082.02	15.05	0
40	571 878.43	572 032.42	4.46	0.000 269
50	507 418.80	507 743.64	7.62	0.000 639
100	352 494.07	354 433.99	7.61	0.005 473

only solve small instances. Using stabilized column generation coupled with a variable neighborhood search heuristic [13] led to solve the continuous relaxation of instances with $n = 3038$ and $p \leqslant 100$. Embedding of this approach in a branch&bound procedure is under way. The values of δ^0_-, δ^0_+, ε^0_- and ε^0_+, as well as the update-δ and update-ε procedures, are defined as for the multisource Weber problem described previously. Results obtained on a SUN SPARC 20 workstation are given in Table 3. Note that the problem with $p = 10$ is solved exactly and that proven near-optimal solutions are obtained for the other values of p.

Acknowledgements

This work has been supported by a FNRS (Switzerland) grant for the first author, by a FCAR (Québec) grant for the second author, by a CRSNG (Canada) grant and a SYNERGIE (Québec) grant for the third and fourth authors, and by an ONR (USA) grant for the last author.

References

[1] J.E. Beasley, A note on solving large p-median problems, European J. Oper. Res. 21 (1985) 270–273.
[2] J.F. Benders, Partitioning procedures for solving mixed-variables programming problems, Numer. Math. 4 (1962) 238–252.
[3] J. Brimberg, P. Hansen, N. Mladenović, E.D. Taillard, Improvements and comparison of heuristics for solving the multisource Weber problem, Les Cahiers du GERAD G-97-37, Montréal, Canada, 1997.
[4] P.C. Chen, P. Hansen, B. Jaumard, H. Tuy, Solution of the multisource Weber and conditional Weber problems by d.-c. programming, Oper. Res. 46 (4) (1998) 548–562.
[5] G.B. Dantzig, P. Wolfe, The decomposition algorithm for linear programming, Econometrica 29 (1961) 767–778.
[6] G. Desaulniers, J. Desrosiers, Y. Dumas, S. Marc, B. Rioux, F. Soumis, Crew pairing at Air France, European J. Oper. Res. 97 (1997) 245–259.
[7] G. Desaulniers, J. Desrosiers, A. Lasry, M.M. Solomon, Crew pairing for a regional carrier, Proc. 7th Internat. Workshop on Computer-Aided Scheduling of Public Transport, 1997.
[8] Z. Drezner, A. Mehrez, G.O. Wesolowsky, The facility location problem with limited distances, Transportation Sci. 25 (1991) 183–187.
[9] R.S. Garfinkel, A.W. Neebe, M.R. Rao, An algorithm for the M-median plant location problem, Transportation Sci. 8 (1974) 217–236.
[10] E.P. Gill, W. Murray, M.A. Saunders, M.H. Wright, Constrained nonlinear programming, Handbooks in Operations Research and Management Science ch. 3, vol. 21, 1989, pp. 171–210.

[11] P. Hansen, B. Jaumard, S. Krau, O. du Merle, A stabilized column generation algorithm for the multisource Weber problem, Les Cahiers du GERAD, Montréal, Canada, 1998, in preparation.
[12] P. Hansen, N. Mladenović, An introduction to variable neighborhood search, Les Cahiers du GERAD G-97-51, Montréal, Canada, 1997, in: S. VOSS et al. (Eds.), Proc. 2nd Internat. Conf. on Metaheuristics — MIC97, Kluwer, Dordrecht, 1998, to appear.
[13] P. Hansen, N. Mladenović, Variable neighborhood search for the p-median, Les Cahiers du GERAD G-97-39, Montréal, Canada, 1998, Location Sci., to appear.
[14] P. Hansen, D. Peeters, D. Richard, J.-F. Thisse, The minisum and minimax location problems revisited, Oper. Res. 33 (1985) 1251–1265.
[15] J. Hiriart-Urruty, C. Lemaréchal, Convex analysis and minimization algorithms II: advanced theory and bundle methods, A Series of Comprehensive Studies in Mathematics, Springer, New York, 1993.
[16] J.E. Kelley, The cutting-plane method for solving convex programs, J. SIAM 8 (1960) 703–712.
[17] S. Krau, Extensions du problème de Weber, Ph.D. Dissertation, École Polytechnique de Montréal, Montréal, Canada, 1997 (in French).
[18] L.S. Lasdon, Optimization Theory for Large Systems, MacMillan, New York, 1970.
[19] R.E. Marsten, W.W. Hogan, J.W. Blankenship, The BOXSTEP method for large-scale optimization, Oper. Res. 23(3) (1975) 389–405.
[20] N. Mladenović, P. Hansen, Variable neighborhood search, Comput. Oper. Res. 24 (1997) 1097–1100.
[21] G.L. Nemhauser, A.H.G. Rinnooy Kan, M.J. Todd, Handbooks in Operations Research and Management Science, vol. 1, Optimization, Elsevier, North-Holland, 1989.
[22] K.E. Rosing, An optimal method for solving the (generalized) multi-Weber problem, European J. Oper. Res. 58 (1992) 414–426.
[23] Y. Ye, Interior Point Algorithms: Theory and Analysis, Wiley-Interscience Series in Discrete Mathematics and Optimization, Wiley, New York, 1997.