Lagrangean Duals and Exact Solution to the Capacitated $p$-Center Problem

Maria Albareda-Sambola
Dpt. Estadística
Universidad Carlos III de Madrid
e-mail:maria.albareda@uc3m.es

Juan A. Díaz
Dpt. Ingeniería Industrial y Mecánica
Universidad de las Américas. Puebla. México
e-mail:juana.diaz@udlap.mx

Elena Fernández
Dpt. d’ Estadística i Investigació Operativa
e-mail:e.fernandez@upc.edu

Abstract

In this work we study the Capacitated $p$-Center Problem ($CpCP$) and we propose an exact algorithm to solve it. We study two auxiliary problems and their relation to $CpCP$, and we propose two different Lagrangean duals based on each of the auxiliary problems. The lower and upper bounds provided by each of the Lagrangean duals reduce notably the set of candidate radii and allow to solve the problem with an exact algorithm based on binary search. The results obtained with experimental testing on various data sets from literature show the efficiency of the proposal that outperforms previous proposals.

1 Introduction

In $p$-center problems we have to partition a set of customers in exactly $p$ clusters. A cluster is defined both by the location of its facility and by its set of customers. There is a given cost for assigning each customer to each facility, and we want to minimize the maximum assignment cost among all the customers. These are discrete location problems since facilities must be located within a given set of potential locations. In the Capacitated $p$-Center Problem $CpCP$, in addition, each customer has a known demand and each potential location a known capacity. Each cluster must be such that the total demand of all its customers cannot exceed the capacity
of its facility. Thus, the \( CpCP \) is the problem of finding the set of \( p \) locations and the assignment pattern that satisfies the capacity constraints were the maximum assignment cost is as small as possible.

The uncapacitated version of the problem has been extensively studied, and both exact and approximated algorithms have been proposed. The paper by Elloumi, Labbé and Pochet [3] is a recent work with a comprehensive up-to-date state of the art. On the contrary, the capacitated \( p \)-center problem has received much less attention in the literature. A local search heuristic is presented in [10]. Approximation algorithms for the particular case in which all the demands are the same have been proposed in [1, 7], and a polynomial exact algorithm for tree networks is developed in [6]. To the best of our knowledge, there is only one exact algorithm for the problem in the literature, recently proposed by Özsoy and Pinar [11].

In this work we propose a new model for this min-max problem with an objective function that is linear, together with an exact algorithm to solve the \( CpCP \). The model uses decision variables associated with different coverage radii and extends the one proposed by Elloumi, Labbé and Pochet in [3] for the uncapacitated \( p \)-center problem. Similarly to the uncapacitated case, the efficiency of a solution method based on the model relies strongly on the ability to derive \( a \) \textit{priori} upper and lower bounds on the optimal value, since this makes it possible to reduce considerably the number of variables in the model. To this end we study two auxiliary problems: the Maximum Demand Coverage with Fixed Radius Problem, denoted \( PD_\delta \), and the Minimum Required Centers with Fixed Radius Problem, denoted \( PC_\delta \). Problem \( PD_\delta \) consists of finding the maximum demand that can be satisfied with at most \( p \) centers within a given coverage radius \( \delta \), whereas problem \( PC_\delta \) consists of finding the minimum number of centers that are needed in order to satisfy the customers demand within a given fixed radius \( \delta \). We derive bounds (both lower and upper) for \( CpCP \) from two different Lagrange duals based on \( PD_\delta \) and \( PC_\delta \), respectively, for fixed radii \( \delta \). The radius that provides the best lower bound for \( CpCP \) is obtained with binary search. Experimental testing on various data sets from literature shows the quality of the obtained bounds that allow a reduction on the number of considered radii within 88.93% and 100.00%. This, in turn, allows to obtain the optimal solution to the problems very efficiently in small computation times.

The paper is structured as follows: Section 2 gives some notation and presents the proposed model. In Sections 3 and 4 we define the two auxiliary problems \( PD_\delta \) and \( PC_\delta \), their relation to the \( CpCP \), and their corresponding Lagrangean dual and heuristic. Sections 5 and Section 6 present the algorithmic framework to obtain the lower and upper bounds, and the exact algorithm, respectively. The computational experiments that we have performed as well as the obtained results are presented in Section 7. Finally, in Section 8 we draw some conclusions and give some final remarks.

2 The Problem

Let \( I = \{1, \ldots, n\} \) and \( J = \{1, \ldots, m\} \) be the sets of indices for customers and centers, respectively. Let also \( h_i \) denote the demand of customer \( i \in I \), and \( b_j \) the capacity of a center located at site \( j \in J \). For each pair \((i, j), i \in I, j \in J, d_{ij} \) is the distance from customer \( i \) to center \( j \).
We define two sets of decision variables

\[ y_j = \begin{cases} 
1 & \text{if a center is located at site } j \in J; \\
0 & \text{otherwise.} 
\end{cases} \]

and

\[ x_{ij} = \begin{cases} 
1 & \text{if customer } i \in I \text{ is assigned to a center located at site } j \in J; \\
0 & \text{otherwise.} 
\end{cases} \]

Then, a classical mathematical programming formulation for the Capacitated \( p \)-Center Problem \( CpCP \) is:

\[
(M0) \quad \min \quad z \\
\text{s.t.} \quad \sum_{j \in J} d_{ij} x_{ij} \leq z \quad \forall \ i \in I \\
\sum_{j \in J} x_{ij} = 1 \quad \forall \ i \in I \\
\sum_{i \in I} h_i x_{ij} \leq b_j \quad \forall \ j \in J \\
x_{ij} \leq y_j \quad \forall \ i \in I, j \in J \\
\sum_{j \in J} y_j = p \\
z \geq 0, x_{ij} \in \{0,1\}, y_j \in \{0,1\} \quad \forall \ i \in I, j \in J
\]

Variable \( z \) together with constraints (2) model the min-max objective function. The assignment constraints (3) ensure that each customer is assigned to exactly one center, and the capacity constraints (4) guarantee that the capacities of the open centers are not violated. Constraints (5) ensure that no customer is assigned to a facility that is not open. Finally, constraints (6) state that \( p \) centers are used. Model \( M0 \) has one continuous variable, \(|I| \times |J| + |J|\) binary variables and \( 2 \times |I| + |J| + |I| \times |J| + 1 \) constraints.

For building the new model for \( CpCP \) we use some additional notation. For a given radius \( \delta \), for each \( j \in J \), let \( I_\delta(j) \) denote the set of customers whose distance to \( j \) does not exceed the radius \( \delta \). That is, \( I_\delta(j) = \{i \in I : d_{ij} \leq \delta \} \). Let \( D^0 < D^1 < \cdots < D^{k_{max}} \) be the sorted different values in the distance matrix \( D = (d_{ij}) \), and \( K = \{1, \cdots, k_{max}\} \). For each \( k \in K \) we define one additional binary variable \( z^k \) that takes the value 0 if it is possible to satisfy the demand of all customers with \( p \) plants within a radius \( D^{k-1} \). That is \( z^k = 1 \) if there is no feasible solution to \( CpCP \) within a radius \( D^{k-1} \). Then, an alternative model for the problem is the following:
\( (M1) \min \quad D^0 + \sum_{k \in K} (D^k - D^{k-1}) z^k \) \hfill (8)

s.t. \[ \sum_{j \in J} x_{ij} = 1 \quad \forall \ i \in I \] \hfill (9)
\[ \sum_{i \in I} h_{ij} x_{ij} \leq b_j \quad \forall \ j \in J \] \hfill (10)
\[ x_{ij} \leq y_{j} \quad \forall \ i \in I, j \in J \] \hfill (11)
\[ x_{ij} \leq z^k \quad \forall \ i \in I, j \in J, k \in K \text{ s.t. } i \notin I_{D^{k-1}}(j) \] \hfill (12)
\[ \sum_{j \in J} y_j = p \] \hfill (13)
\[ z^k \in \{0, 1\} \forall \ k \in K, x_{ij} \in \{0, 1\}, y_j \in \{0, 1\} \quad \forall \ i \in I, j \in J \] \hfill (14)

Model \( M1 \) extends the model of Elloumi, Labbé and Pochet [3] for the uncapacitated \( p \)-center problem to the capacitated case. Constraints (9), (10), and (11) are the same as constraints (3), (4), and (5) of \( M0 \). For a fixed \( k \in K \), its corresponding set of constraints (12) activates the variable \( z^k \) at value 1 whenever some customer is assigned to a facility located at distance at least \( D^k \) from the customer. The value of the objective function is precisely the value of the radius associated with the activated variable \( z^k \) with a larger index. This corresponds to the maximum assignment distance among all customers. Therefore, in an optimal solution \( z^k = 0 \) if and only if all customers can be served at a distance strictly smaller than \( D^k \). Thus, the above model is valid for \( CpCP \).

Model \( M1 \) has \(|I| \times |J| + |J| + |K| \) binary variables, which is considerably more than the number of variables in \( M0 \), given that \(|K| \) typically will be quite large. The number of constraints is \(|I| + |J| + |I| \times |J| + |I| \times |J| \times |K| + 1 \) which will also be considerably larger than the number of constraints in \( M0 \). However, as we will see, when upper and lower bounds on the optimal value of the problem are available, it is possible to reduce considerably the size of this model so it becomes very useful to solve efficiently large instances of the problem in small computation times.

To this end we will obtain lower and upper bounds on the optimal value of \( CpCP \) to reduce the size of \( M1 \) by fixing variables at values 0 or 1. In particular, if \( LB \) is a lower bound, then in any optimal solution to \( M1 \), \( z^k = 1, \forall k \in K \text{ s.t. } D^k < LB \). Similarly, if \( UB \) is a valid upper bound, then \( z^k = 0, \forall k \in K \text{ s.t. } D^k \geq UB \) in any optimal solution to \( M1 \). In order to obtain tight lower and upper bounds for \( CpCP \) in the next two sections we address two different auxiliary problems.

### 3 Maximum demand coverage within fixed radius problem.

The same \( y \) and \( x \) variables defined in the above section can also be used for building a model for the problem of finding the maximum demand that can be satisfied with at most \( p \) plants within a given radius \( \delta \). For \( i \in I \), let \( J_\delta(i) := \{ j \in J : i \in I_\delta(j) \} \). The model is as follows:
\[
(PD_\delta)
\]
\[
H(\delta) = \max \sum_{j \in J} \sum_{i \in I_{\delta}(j)} h_i x_{ij}
\]
\[
s.t. \sum_{j \in I_\delta(i)} x_{ij} \leq 1 \quad \forall i \in I
\]
\[
\sum_{i \in I_{\delta}(j)} h_i x_{ij} \leq b_j y_j \quad \forall j \in J
\]
\[
\sum_{j \in J} y_j \leq p
\]
\[
x_{ij} \in \{0, 1\}, y_j \in \{0, 1\} \quad \forall j \in J, i \in I_{\delta}(j)
\]

In constraints (17) we substitute the coefficients \(b_j\) by \(\tilde{b}_j\), \(\forall j \in J\), where \(\tilde{b}_j = \sum_{i \in I_{\delta}(j)} h_i\), when \(\sum_{i \in I_{\delta}(j)} h_i \leq b_j\); and \(\tilde{b}_j = b_j\), otherwise. Throughout we assume that capacity constraints (17) are expressed in this stronger form.

The following observations help us to appreciate the close relationship between the above problem and \(CpCP\). Throughout \(H_{ag} = \sum_{i \in I} h_i\) will denote the aggregated demand.

Remarks

1. The optimal solution to \(CpCP\) can be obtained by finding the smallest \(k \in K\) such that \(H(D^k) \geq H_{ag}\). Note that the solutions to \(PD_\delta\) correspond to binary assignments of customers to open facilities. Thus, if the total demand that can be satisfied within the radius \(D^k\) is at least the aggregated demand, all customers are assigned and \(D^k\) is a valid upper bound on \(CpCP\).

2. If, for a given \(\delta\), \(H(\delta) < H_{ag}\), then \(\delta\) gives a valid lower bound on the value of \(CpCP\). Moreover, when \(\delta = D^k\) for some \(k \in K\) such that \(H(D^{k-1}) < H_{ag}\), then \(D^k\) is also a valid lower bound on the optimal value of \(CpCP\), even if \(H(D^k) \geq H_{ag}\).

The above remarks can be summarized in the following result:

\textbf{Proposition 1} The optimal solution to \(CpCP\) is given by \(D^{k^*}\), where \(k^* \in K\) is such that \(H(D^{k^*-1}) < H_{ag} \leq H(D^{k^*})\).
of all the customers. On the contrary, Remark 2 also holds for the relaxations of $PD_δ$. For this reason we will obtain a valid lower bound for $CpCP$ by solving a relaxation of $PD_δ$.

**Proposition 2** For $k \in K$, if $\overline{H}(D^k) < H_{ag}$ then $D^k$ is a valid lower bound on the optimal value to $CpCP$. The best such bound is given by $D^{k^*}$, where $k^* \in K$ is such that $\overline{H}(D^{k^*-1}) < H_{ag} \leq \overline{H}(D^{k^*})$.

We next present the relaxation of $PD_δ$ that we have used and we show how to solve it in order to obtain the value $\overline{H}(δ)$, for a given value of $δ$.

### 3.1 Variable splitting relaxation of $PD_δ$

In this section we define a relaxation of $PD_δ$ that is a Lagrangean decomposition based on variable splitting. Lagrangean decomposition based on variable splitting has been successfully used to address various discrete problems (see, for instance, [5]) and, in particular, for solving some discrete location problems [2]. In our case we do variable splitting on the $x$ variables of $PD_δ$, by making mirror copies $w$ of these variables, and by introducing a parameter $α$, $0 < α < 1$. The resulting model is:

\[(PDS_δ) \quad v(δ) = \max \sum_{j \in J} \sum_{i \in I_δ(j)} α h_i x_{ij} + \sum_{j \in J} \sum_{i \in I_δ(j)} (1 - α) h_i w_{ij}\]

s.t.

\[\sum_{j \in J} w_{ij} \leq 1 \quad \forall \ i \in I\]

\[\sum_{i \in I_δ(j)} h_i x_{ij} \leq b_j^δ y_j \quad \forall \ j \in J\]

\[\sum_{j \in J} y_j \leq p\]

\[x_{ij} = w_{ij} \quad \forall \ j \in J, i \in I_δ(j)\]

\[w_{ij} \in \{0, 1\}, x_{ij} \in \{0, 1\}, y_j \in \{0, 1\} \quad \forall \ j \in J, i \in I_δ(j)\]

In the computational experiments in Section 7 we have used $α = 0.5$ When relaxing constraints (24) in a Lagrangean fashion, the Lagrangean subproblem for a given multipliers vector $λ \in \mathbb{R}^{I_δ × J}$ is:
\[ L^1(\lambda) = \max \sum_{j \in J} \sum_{i \in I_{\delta}(j)} \alpha h_i x_{ij} + \sum_{j \in J} \sum_{i \in I_{\delta}(j)} (1 - \alpha) h_i w_{ij} + \sum_{j \in J} \sum_{i \in I_{\delta}(j)} \lambda_{ij}(w_{ij} - x_{ij}) \]

s.t.
\[ \sum_{j \in I_{\delta}(i)} w_{ij} \leq 1 \quad \forall \ i \in I \]
\[ \sum_{i \in I_{\delta}(j)} h_i x_{ij} \leq \hat{b}_{\delta} y_j \quad \forall \ j \in J \]
\[ \sum_{j \in J} y_j \leq p \]
\[ w_{ij} \in \{0,1\}, x_{ij} \in \{0,1\}, y_j \in \{0,1\} \quad \forall \ j \in J, i \in I_{\delta}(j) \]

which is separable in:
\[ L^1_w(\lambda) = \max \sum_{j \in J} \sum_{i \in I_{\delta}(j)} [(1 - \alpha) h_i + \lambda_{ij}] w_{ij} \]

s.t.
\[ \sum_{j \in I_{\delta}(i)} w_{ij} \leq 1 \quad \forall \ i \in I \]
\[ w_{ij} \in \{0,1\} \quad \forall \ j \in J, i \in I_{\delta}(j) \]

and
\[ L^1_{x,y}(\lambda) = \max \sum_{j \in J} \sum_{i \in I_{\delta}(j)} [\alpha h_i - \lambda_{ij}] x_{ij} \]

s.t.
\[ \sum_{i \in I_{\delta}(j)} h_i x_{ij} \leq \hat{b}_{\delta} y_j \quad \forall \ j \in J \]
\[ \sum_{j \in J} y_j \leq p \]
\[ x_{ij} \in \{0,1\}, y_j \in \{0,1\} \quad \forall \ j \in J, i \in I_{\delta}(j) \]

The solution to \( L^1_w(\lambda) \) can be obtained by inspection. For solving \( L^1_{x,y}(\lambda) \), we first solve for each \( j \in J \) the following knapsack problem

\[ KP_j(\lambda) = \max \sum_{i \in I_{\delta}(j)} [\alpha h_i - \lambda_{ij}] x_{ij} \]

s.t.
\[ \sum_{i \in I_{\delta}(j)} h_i x_{ij} \leq \hat{b}_{\delta} \]
\[ x_{ij} \in \{0,1\} \quad \forall \ i \in I_{\delta}(j) \]

and then we order the indices in \( J \) by non increasing values of \( KP_j(\lambda) \), so that \( KP_{j_1}(\lambda) \geq KP_{j_2}(\lambda) \geq \cdots \geq KP_{j_{|J|}}(\lambda) \). Let \( p^* = \min \{p, \max \{r : KP_r(\lambda) > 0\}\} \). Then, the solution
to $L_{1,\gamma}^\delta(\lambda)$ consists of opening the plants $j_1, j_2, \ldots, j_p$, and assigning to each of them the customers given by the optimal solution to its corresponding knapsack problem.

Thus, for a given vector of multipliers $\lambda$, we can apply the above procedures to obtain the value $L_1^\delta(\lambda) = L_{1,\gamma}^\delta(\lambda) + L_{1,\nu}^\delta(\lambda)$, that is a valid lower bound on the value of $PD_\delta$. As usual, the best lower bound is obtained by solving the Lagrangean dual. Therefore,

$$(D1) \quad \Pi(\delta) = \max_{\lambda \in \mathbb{R}^{|J| \times |I|}} L_1^\delta(\lambda)$$

For a given vector $\lambda$, a subgradient of $L_1^\delta(\lambda)$ is given by $\gamma = w(\lambda) - x(\lambda)$, where $w(\lambda)$ and $x(\lambda)$ are the solutions to the subproblems, $L_{1,\nu}^\delta(\lambda)$ and $L_{1,\gamma}^\delta(\lambda)$, respectively. Hence, (D1) can be solved by applying subgradient optimization.

### 3.2 Heuristic solutions to $C_{pCP}$ from $L_1^\delta(\lambda)$.

At the inner iterations of subgradient optimization we apply a very simple heuristic for obtaining feasible solutions to $C_{pCP}$. For a given vector of multipliers $\lambda$, let $w(\lambda), x(\lambda), y(\lambda)$ denote the optimal solution to $L_{1,\gamma}^\delta(\lambda)$. Throughout the heuristic, the set of open facilities is given by $J(\lambda) = \{ j \in J : y_j(\lambda) = 1 \}$ and the facility to which customer $i \in I$ is assigned is denoted $j(i)$. The heuristic has 2 steps, that are the following:

1. First, all customers $i \in I$, such that there exists a facility $j^* \in J(\lambda)$ with $x_{ij^*}(\lambda) = 1$, are assigned to that facility: i.e., $j(i) = j^*$ (ties are broken by the minimum assignment distance).

2. All customers that are not yet assigned are ordered by decreasing values of their demands. Then, each unassigned customer is assigned to the open facility with more available capacity (if any). That is, for each unassigned customer let $j(i) \in \arg \max \{ s_j : j \in J(\lambda) \cap J_d(i), s_j \geq h_i \}$ where $s_j = b_j - \sum_{i \in A_j} h_i$ denotes the available capacity of facility $j$ ($A_j$ is the set of customers already assigned to facility $j$).

Obviously, the above heuristic may fail to obtain a feasible solution. At the iterations when the heuristic succeeds, we compare the obtained solution with the best solution obtained so far, and we record it if it improves its objective function value.

### 4 The minimum required centers within fixed radius problem.

The problem that we present next is also closely related to $C_{pCP}$. It consists of finding the minimum number of centers that are needed to satisfy the customers demands within a fixed radius $\delta$. We refer to that problem as to the Minimum Required Centers within Fixed Radius
Problem. Using the same decision variables as before the problem can be modeled as:

\[
(\text{PC}_\delta) \quad Y(\delta) = \min \sum_{j \in J} y_j \quad (26)
\]

s.t.

\[
\sum_{j \in J} x_{ij} = 1 \quad \forall i \in I \quad (27)
\]

\[
\sum_{i \in I_{\delta}(j)} h_i x_{ij} \leq \bar{b}_j \quad \forall j \in J \quad (28)
\]

\[
x_{ij} \leq y_j \quad \forall j \in J, i \in I_{\delta}(j) \quad (29)
\]

\[
\sum_{j \in J} \bar{b}_j y_j \geq H_{ag} \quad \forall j \in J, i \in I_{\delta}(j) \quad (30)
\]

\[
x_{ij} \in \{0, 1\}, y_j \in \{0, 1\} \quad \forall j \in J, i \in I_{\delta}(j) \quad (31)
\]

The aggregated demand constraint (30) is redundant in \(\text{PC}_\delta\), but we include it in its formulation in order to strengthen the relaxation that we will consider later in this section. Problems similar to \(\text{PC}_\delta\), but with the non reinforced expression of the capacity constraints (28) and without constraint (30), have been considered in the work of Özsoy and Pinar [11] for solving \(CpCP\). Note that when for a given \(\delta\), \(Y(\delta) > p\), then \(\delta\) is a lower bound on the optimal value of \(CpCP\). Thus, similarly to Proposition 1 in Section 3, the optimal solution to \(CpCP\) is given by \(D_{k^*}\), where \(k^* \in K\) is such that \(Y(D_{k^*} - 1) > p \geq Y(D_{k^*})\). Again, problems \(\text{PC}_\delta\) are \(NP\)-hard so we will resort to solving relaxations of \(\text{PC}_\delta\) in order to obtain valid lower bounds. Let, \(\overline{Y}(\delta)\) denote the optimal value of the relaxed problem for a fixed value of \(\delta\).

**Proposition 3** For \(k \in K\), if \(\overline{Y}(D^k) > p\) then \(D^k\) is a valid lower bound on the optimal value to \(CpCP\). The best such bound is given by \(D^{k^*}\), where \(k^* \in K\) is such that \(\overline{Y}(D^{k^*} - 1) > p \geq \overline{Y}(D^{k^*})\).

We next present the relaxation of \(\text{PC}_\delta\) that we have used and we show how to solve it in order to obtain the value \(\overline{Y}(\delta)\), for a given value of \(\delta\).

### 4.1 Lagrangean relaxation of \(\text{PC}_\delta\)

When relaxing constraints (27) in a Lagrangean fashion the subproblem that we obtain for a given multipliers vector \(u \in \mathbb{R}^{\|I\|}\) is:

\[
L^\delta(u) = \min \sum_{j \in J} y_j + \sum_{i \in I} u_i \left(1 - \sum_{j \in I_{\delta}(i)} x_{ij}\right) = \sum_{i \in I} u_i + \min \left\{ \sum_{j \in J} \left(y_j - \sum_{i \in I_{\delta}(j)} u_i x_{ij}\right) \right\}
\]

s.t.

\[
\sum_{i \in I_{\delta}(j)} h_i x_{ij} \leq \bar{b}_j \quad \forall j \in J \quad (28)
\]

\[
x_{ij} \leq y_j \quad \forall j \in J, i \in I_{\delta}(j) \quad (29)
\]

\[
\sum_{j \in J} \bar{b}_j y_j \geq H_{ag} \quad \forall j \in J, i \in I_{\delta}(j) \quad (30)
\]

\[
x_{ij} \in \{0, 1\}, y_j \in \{0, 1\} \quad \forall j \in J, i \in I_{\delta}(j) \quad (31)
\]
For each center $j \in J$, consider the subproblem 

$$KP_j(u) = \max_{i \in I_{\delta}(j)} u_i x_{ij}$$

s.t. 

$$\sum_{i \in I_{\delta}(j)} h_i x_{ij} \leq \hat{b}_j$$

$$x_{ij} \in \{0, 1\} \quad \forall i \in I_{\delta}(j)$$

The index set of centers to be open in the optimal solution to $L^2_\delta(u)$, denoted $J(u) \subseteq J$, can be found by solving 

$$AG(u) = \min \sum_{j \in J} (1 - KP_j(u)) y_j$$

s.t. 

$$\sum_{j \in J} \hat{b}_j y_j \geq D_{ag}$$

$$y_j \in \{0, 1\} \quad \forall j \in J$$

In particular, if $y(u)$ denotes the optimal solution to $AG(u)$ then $J(u) = \{j \in J : y_j(u) = 1\}$. Thus, the optimal solution to $L^2_\delta(u)$ consists of opening the centers indexed in $J(u)$ and for these centers performing the assignment of customers given by the optimal solution to subproblem $KP_j(u)$. Let $x(u)$ denote the resulting assignment vector.

Again, we use subgradient optimization to solve the Lagrangean dual 

$$(D2) \quad \bar{\mathcal{Y}}(\delta) = \max_{u \in \mathbb{R}^{|I|}} L^2_\delta(u)$$

For a given vector $u$, the components of a subgradient vector $\gamma \in \mathbb{R}^{|I|}$ of $L^2_\delta(u)$ are given by 

$$\gamma_i = 1 - \sum_{j \in I_{\delta}(i)} x_{ij}(u).$$

### 4.2 Heuristic solutions to CpCP from $L^2_\delta(u)$.

At the inner iterations of subgradient optimization we apply a heuristic for obtaining feasible solutions to CpCP. The idea of the heuristic is opening enough plants so as to assign all the customers within the coverage radius $\delta$. Note that for a given set of plants $J(u)$ there might be customers that do not fall within the coverage radius $\delta$ of any of the plants in $J(u)$. We will denote $I_{uc} = \{i \in I : d_{ij} > \delta, \forall j \in J(u)\} = \bigcup_{j \in J(u)} (I \setminus I_{\delta}(j))$, the set of such “uncovered” customers. Let $JO$ denote the set of plants opened by the heuristic (initially $JO = \emptyset$). The heuristic consists of three steps which are the following:

1. Let $J_1 = \{j \in J \setminus J(u) : I_{uc} \cap I_{\delta}(j) \neq \emptyset\}$ be the set of plants that cover at least one customer in $I_{uc}$ within the coverage radius $\delta$. When $I_{uc} \neq \emptyset$, in order to obtain a feasible solution to CpCP, some plants in $J_1$ must open. For this reason, we successively open plants in $J_1$, until all the customers in $I_{uc}$ are assigned to some open plant within the
coverage radius $\delta$. The criterion that is used to select plants to open is to maximize the demand that is satisfied. Recall that if plant $j$ is opened the demand that it will satisfy is given by

$$s_j = \min \{ \hat{b}_j, \sum_{i \in I(j) \cap I_{nc}} h_i \}$$

and corresponds to the total demand of the customers that will be assigned to it if opened. Let $A_j$ denote the set of such customers. A summary of Step 1 is depicted in Algorithm 1.

**Algorithm 1** Step 1.

1. Initialize $I_a = \emptyset$.
2. While $I_{nc} \neq \emptyset$ do:
   - Select the non open plant in $J_1$ with more available capacity $j^* \in \arg \max \{ s_j : j \in J_1 \}$
   - $JO := JO \cup \{ j^* \} $
   - $J_1 := J_1 \setminus \{ j^* \}$
   - $j(i) = j^* \forall i \in A_{j^*}$
   - $I_{nc} := I_{nc} \setminus A_{j^*}$
   - $I_a := I_a \cup A_{j^*}$
3. End while

2. In the second step, we order unassigned customers ($I_{na}$) by decreasing values of their demands and try to assign them to some open plant within its coverage radius $\delta$, with enough available residual capacity. If no such plant exists, since unassigned customers fall within the coverage radius $\delta$ of at least one plant in $J(u)$ we open one plant in $J(u)$ and assign the customer to it. In both cases the selected plant is the one (in the corresponding set of candidate plants) with more residual capacity $s_j = \hat{b}_j - \sum_{i : j(i) = j} h_i$. A summary of Step 2 is depicted in Algorithm 2.

3. If there are still customers that have not been assigned, new plants are opened until all unassigned customers are assigned to one open plant. Note that at this point the only non opened plants that could cover some unassigned customer within the radius $\delta$, with enough capacity, are the ones in $J \setminus (JO \cup J(u))$. Again we consider unassigned customers by decreasing values of their demand and the criterion that is used to select the plants is the available capacity.

5 **Lower and upper bounds to $CpCP$**

We next obtain two lower bounds, $LB1 = D^{k1}$ and $LB2 = D^{k2}$, that satisfy the conditions of Propositions 2 and 3, respectively. Both radii, $D^{k1}$ and $D^{k2}$, are obtained by applying binary search on the value of the index, $k$ such that the radius $D^k$ gives a valid lower bound. The procedure to obtain $D^{k1}$ solves a series of Lagrangean duals of the type $\overline{H}(D^k)$, whereas the procedure to obtain $D^{k2}$ resorts to the solution of Lagrangean duals of the type $\overline{Y}(D^k)$. When solving the corresponding Lagrangean duals, both procedures may also generate valid upper bounds, $ub1(D^k)$ and $ub2(D^k)$, by applying the heuristics as explained in Subsections 3.2 and 4.2, respectively. More specifically, the values of the bounds are the objective function value in
Algorithm 2 Step 2.

\[ I_{na} = I \setminus I_{na} \]

for \( i \in I_{na} \) do

Define set of candidate plants

if \( J_O \cap \{ j \in J_{\delta i} : h_i \leq s_j \} \neq \emptyset \) then

\( J_2 = J_O \cap \{ j \in J_{\delta i} : h_i \leq s_j \} \)

else

\( J_2 = \{ j \in J(u) : h_i \leq s_j \} \)

end if

if \( J_2 \neq \emptyset \) then

Select the candidate plant with more available capacity

\( j^* \in \arg \max \{ s_j : j \in J_2 \} \)

if \( j^* \in J(u) \) then

\( J(u) := J(u) \setminus \{ j^* \} \)

\( J_O := J_O \cup \{ j^* \} \)

end if

\( j(i) := j^* \)

\( I_{na} := I_{na} \setminus \{ i \} \)

end if

end for

CpCP of the corresponding feasible solutions. Let \( UB_1 \) and \( UB_2 \) denote the best such bounds. The two procedures follow a very similar structure which is summarized in Algorithm 3:

Algorithm 3 Identification of lower and upper bounds.

\[
a = 1, \; b = k_{\max}, \; UB_1 = D^{k_{\max}}
\]

while \( a \neq b \) do

\[ k := \left\lfloor \frac{a+b}{2} \right\rfloor \]

Solve \( \bar{Y}(D^k) \)

if \( ub1(D^k) < UB_1 \) then

\( UB_1 := ub1(D^k) \)

\( b := k \)

end if

if \( \bar{Y}(D^k) < H_{ag} \) then

\[ a := k + 1 \]

else

\( b := k \)

end if

end while

\[ k_1^* = a \]

\( LB_1 := D^{k_1^*} \)

\[
a = 1, \; b = k_{\max}, \; UB_2 = D^{k_{\max}}
\]

while \( a \neq b \) do

\[ k := \left\lfloor \frac{a+b}{2} \right\rfloor \]

Solve \( \bar{Y}(D^k) \)

if \( ub2(D^k) < UB_2 \) then

\( UB_2 := ub2(D^k) \)

\( b := k \)

end if

if \( \bar{Y}(D^k) > p \) then

\[ a := k + 1 \]

else

\( b := k \)

end if

end while

\[ k_2^* = a \]

\( LB_2 := D^{k_2^*} \)
6 The exact algorithm

In this section we describe the exact algorithm that we apply to obtain the optimal solution of $C_2CP$. The algorithm performs binary search on the value of the largest radius $D_k$ such $z^k = 1$ in the optimal solution to $M_1$. Let $D_k$ denote the trial value at a given iteration of the binary search. At the end of the iteration we want to fix the value of $z^k$. That is, we want to know whether $z^k = 0$ or $z^k = 1$ in the optimal solution to $M_1$. The answer to that question can be obtained by solving the auxiliary problem $PC_\delta$ with $\delta = D^{k-1}$.

If $Y(\delta) \leq p$ for $\delta = D^{k-1}$, then there exists a feasible solution to $M_1$ with a radius no larger than $\delta = D^{k-1}$, so we can fix $z^k = 0$ in the optimal solution to $M_1$. In this case we can update the value of the upper bound to $\delta = D^{k-1}$. On the contrary, if $Y(\delta) > p$ there exists no feasible solution to $M_1$ with a maximum radius of $\delta$. Thus, $z^k = 1$ in the optimal solution to $M_1$. In this case we can update the value of the lower bound to $D^k$. A summary of the procedure is given in Algorithm 4.

In practice, instead of knowing the exact optimal value of $Y(\delta)$, we just need to know whether or not $Y(\delta) \leq p$. To this end lower and upper bounds can be very useful. In particular, if $Y(\delta) > p$ ($Y(\delta)$ denotes the value of the LP relaxation of $Y(\delta)$), then $Y(\delta) > p$. Also if there exists a feasible solution solution to $Y(\delta)$ with value smaller than or equal to $p$, then $Y(\delta) \leq p$.

Algorithm 4 Exact solution to $M_1$ with binary search

1. Elimination Test.
   Let $k_l, k_u$ be such that $LB = D^{k_l}$, and $UB = D^{k_u}$.
   Fix $z^k = 1$, $\forall k \leq k_l$.
   Fix $z^k = 0$, $\forall k \geq k_u$.

2. Optimal Solution to (M1).
   $a = k_l, b = k_u$
   while $(a \neq b)$ do
     $k := \lfloor \frac{a + b}{2} \rfloor$
     if $Y(\delta) > p$ then
       $a := k + 1$
     else
       $b := k$
     end if
   end while
   $k^* = a$

7 Computational experiments

In order to evaluate the effectiveness of our algorithm, we have run a series of computational experiments. In this section we report on the obtained results. All the algorithms have been
coded in C and run on a Dell PC Intel Pentium 4, 2GHz and 512 MB of RAM. All the benchmark instances that we have used, have also been used in [11, 10] for the $CpCP$. They are the following:

1. The set of 20 instances with Euclidean distances, generated by Osman and Christofides [9] for the capacitated $p$ median problem. These instances are available in the OR-Library (http://mscmga.ms.ic.ac.uk/info.html). This set consists of 2 groups of 10 instances each, with $|V| = 50$ and $p = 5$, and $|V| = 100$ and $p = 10$, respectively. These instances are referred to as Set $S1$.

2. The set of eight instances derived in [10] for $CpCP$ from the two instances with non-Euclidean distances generated for the maximum covering location problem by Galvão and ReVelle [4]. There are two instances of each of the following dimensions: (100, 5), (100, 10), (150, 10) and (150, 15). These instances are referred to as Set $S2$.

3. The set of six instances with Euclidean distances of [8] that correspond to real data from the Brazilian city of São José dos Campos. These instances are available from http://www.lac.inpe.br/~lorena/instancias.html. Their dimensions $(n, p)$ are (100,10), (200,15), (300,25), (300,30), (402,30) and (402,40), respectively. These instances are referred to as Set $S3$.

The numerical results with sets $S1$, $S2$ and $S3$ are depicted in Tables 1, 2 and 3, respectively. In these tables we evaluate our results in terms of quality, by comparing our lower and upper bounds with the value of the optimal solutions, and also in terms of effectiveness via the required cpu times. Our results are also compared with those of Öszoy and Pinar [11] which, to the best of our knowledge, are the only known results of an exact algorithm for this problem. The results that we report here have been obtained with our implementation of their method, which allows a better comparison of the required cpu times. These results do not fully coincide with the ones reported in their paper. In all the tables the first four columns give information on the instances. In particular, the columns under the headings Prob, $n$ and $p$ give for each instance its name, number of nodes and value of the parameter $p$, respectively. The value of an optimal solution to each instance is given in column Opt (in Table 3 the optimal solution is not known for instance SJC4; for this instance the depicted value is marked with an asterisk and corresponds to the best known lower bound). Columns 5-8, under the heading Lag.Dual, give information on the results obtained with the Lagrangean Duals that we have presented. In particular, columns LB and UB depict the obtained lower and upper bounds respectively. Columns under $D1$ refer to the results obtained with the Lagrangean dual $D1$, based on the solution of $PD_{\delta}$, whereas columns under $D2$ report the results obtained with the Lagrangean dual $D2$, based on the solution of $PC_{\delta}$. Column $OP$ gives the lower bound obtained in the Phase I of the algorithm of Öszoy and Pinar [11]. The next three columns under $\%gap$ give the percent deviation between the lower bound and the optimal solution (defined as $100 \times \frac{Opt - LB}{Opt}$) obtained with our Lagrangean Duals (columns $D1$ and $D2$) and with the approach of Öszoy and Pinar [11] (column $OP$). The last six columns give the required cpu times to obtain the lower bounds (columns under CPU time LB) and to solve exactly the problem, including the time required to obtain the lower bound (columns under CPU time exact). In both cases the columns under $ADF$ refer to our approach and the columns under $OP$ refer to the results of the algorithm of Öszoy and Pinar.
In our opinion, the obtained results are very satisfactory. In general, the lower bounds obtained with the Lagrangian Duals are very good and the percent deviation with respect to the optimal solution is very small. This can be appreciated, in particular, for the instances in the set S1, where this bound was already optimal for 15 out of the 20 instances, both for $D_1$ and $D_2$. For the remaining instances in S1, and the instances in S2 and S3, the gap does not exceed 5.00%, excepting for two instances (instance p20 of S1 with gap 9.52% and instance G3 of S2 with gap 40.68%). The quality of the lower bounds obtained with $D_1$ and $D_2$ is very similar and, excepting two instances (G6 and 3b) with slight differences the two Lagrangian duals give similar lower bounds. On the contrary, there is a clear difference in the quality of the upper bounds obtained with $D_1$ and $D_2$, since $D_2$ gives much better upper bounds. Despite its simplicity, the quality of the upper bounds obtained with $D_2$ is, in general, quite good and the percent deviations from the optimal solutions are, on the average 6.91%, 3.92% and 13.00% for instances in sets S1, S2 and S3, respectively. In general terms, our lower bounds are better than those of OP: For the set S1, our lower bound is better for 17 instances and coincides for three instances. For the set S2 our bound is better for four instances, and coincides with the bound of OP for the other four instances whereas for the set S3, our lower bound is always better. The improvement on the lower bounds that we obtain with the Lagrangian duals with respect to the algorithm of Özsoy and Pinar can also be appreciated in the percent gaps with respect to the optimal solutions. In particular, for the instances in set S1, our average percent gaps are 1.36% both for $D_1$ and $D_2$, whereas it is 7.29% for the algorithm of Özsoy and Pinar. With set S2, our average percent gap is 7.07% for $D_1$ and 6.93% for $D_2$, whereas it is 12.80% for OP. Similarly, for the instances in S3, our percent gap is 1.94% for $D_1$ and 2.09% for $D_2$, and 4.25% for OP.

In our opinion, the increasing difficulty of instances in S1, S2 and S3 is due not only to the increasing sizes of the instances but also to other characteristics on the instances like, for instance, the range between the largest and the smallest radii. Thus, given that, for the considered sets of instances, the distribution of the number of radii in the interval of their range is not uniform, we believe that a better indicator of the quality of the lower bounds than the percent gap is the number of radii in the interval $[LB, Opt]$. As can be seen, the cpu times required to obtain these bounds are small, taking into account the size and the difficulty of the problems. In particular, for the instances in S1 this time never exceeded 26 seconds, neither for $D_1$ nor $D_2$, and was on the average 8.62 seconds for $D_1$ and 3.29 seconds for $D_2$; for S2 the bounds were obtained in less than 132 seconds (68.00 secs. and 46.40 secs. on the average for $D_1$ and $D_2$, respectively); and for S3 the lower bounds were obtained in less than 3,000 secs. (1,276.33 secs. and 161.72 secs. on the average for $D_1$ and $D_2$, respectively). These results indicate that $D_2$ is much more efficient than $D_1$, since it gives similar lower bounds and much better upper bounds in much smaller cpu times. The average cpu times required by OP to obtain their lower bounds are 4.02, 7.48 and 1330.07 seconds, respectively for sets S1, S2 and S3. These numbers show that, on the average, $D_1$ consumes considerably more time than $OP$ for sets S1 and S2, $D_2$ is faster than $OP$ for sets S1 and S3, whereas $OP$ is faster than $D_2$ in the set S2.

The quality of our lower and upper bounds indeed does have an effect on the performance of the exact algorithm. As can be seen in Tables 1, 2 and 3, there is a high number of $z^k$ variables that were fixed after solving our Lagrangian Duals: 96.28% and 98.40% for $D_1$ and $D_2$, respectively, in S1; 83.72% and 95.81 for $D_1$ and $D_2$, respectively, in S2; and 92.06%
and 97.52 % for $D1$ and $D2$, respectively, in S3. The effect of this reduction on the number of variables on the overall performance of the algorithm is that the algorithm is able to solve exactly the instances in cpu times that we consider small for problems of this difficulty. First we should note that the overall times are quite different among problems of the same sizes with roughly the same initial gap. This is common both to our approach and to OP.

Finally, we can observe that the ratio $\frac{\text{CPU time exact ADF}}{\text{CPU time exact OP}}$ of the total cpu time of ADF over total cpu time of OP is on the average 1.58 and 0.77 for $D1$ and $D2$, respectively, in S1; 0.74 and 0.94 for $D1$ and $D2$, respectively, in S2; and 0.42 and 0.2 for $D1$ and $D2$, respectively, in S3. Thus, the results of Tables 2 and 3 show that the cpu times required by our approach are, on the average, considerably smaller than the ones of OP both with $D1$ and $D2$, and this reduction is much higher in the largest more difficult instances in S3. However, for some instances our total times are somewhat bigger than the ones of OP. This can be explained as follows. The exact algorithm of OP starts with a trial value for the radius $\delta$ equal to the lower bound, and increases this value to the next radius until the corresponding problem $PC_\delta$ is feasible. When there are few radii in the interval $[LB, Opt]$ this strategy can be more efficient than binary search, specially for the instances (like p20) where the upper bound is not tight. However, the obtained results indicate that, when the number of radii in the interval $[LB, Opt]$ is big, the strategy of the binary search is much better. For better appreciating the above comment, Table 4 depicts in the columns under the heading “# radii in $[LB, Opt]$”, the values of the number of radii in the intervals $[LB, Opt]$ for our two approaches and for OP. In addition in Table 4 columns under the headings #rad and range give, for each instance, the number of different radii (different values of the $d_{ij}$ distances) the difference between the largest and the smallest radii, respectively. Finally, columns under #fix give, both for $D1$ and $D2$, the number of $z^k$ variables that have been fixed when applying the elimination test with the obtained bounds.

8 Conclusions

In this paper we have presented an exact algorithm for the CpCP, which is an NP-hard problem. We have proposed a new model based on the one of Elloumi, Labbé and Pochet [3] for the uncapacitated $p$-center problem, and we have studied two different auxiliary problems: the Maximum Demand Coverage with Fixed Radius Problem, denoted $PD_\delta$, and the Minimum Required Centers with Fixed Radius Problem, denoted $PC_\delta$. The difficulty of CpCP is remarkable since, even if the optimal value of one instance is known, finding an optimal solution amounts to solve a Maximum Demand Coverage within Fixed Radius Problem, which is also an NP-hard problem that has the Generalized Assignment Problem as particular case.

Lower and upper bounds for CpCP have been obtained from two Lagrangean duals, derived from each of the auxiliary problems. In turn, the lower and upper bounds allow to eliminate most of the variables of the problem and make it possible to solve efficiently large instances in small cpu times.

The numerical results obtained with the computational experiments on three well-known sets of benchmark instances show the efficiency of our proposal that outperforms the algorithm of Öszoy and Pinar which is, to the best of our knowledge, the only exact algorithm that has
<table>
<thead>
<tr>
<th>Prob</th>
<th>n</th>
<th>p</th>
<th>Opt.</th>
<th>Lag. Dual</th>
<th>Opt.</th>
<th>% gap</th>
<th>CPU time LB</th>
<th>CPU time exact</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>LB</td>
<td></td>
<td></td>
<td>LB</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>D1   D2</td>
<td>D1   D2</td>
<td></td>
<td>D1   D2</td>
<td></td>
</tr>
<tr>
<td>p1</td>
<td>50</td>
<td>5</td>
<td>29</td>
<td>29</td>
<td>29</td>
<td>29</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>p2</td>
<td>50</td>
<td>5</td>
<td>33</td>
<td>33</td>
<td>33</td>
<td>33</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>p3</td>
<td>50</td>
<td>5</td>
<td>26</td>
<td>26</td>
<td>26</td>
<td>26</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>p4</td>
<td>50</td>
<td>5</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>p5</td>
<td>50</td>
<td>5</td>
<td>29</td>
<td>29</td>
<td>29</td>
<td>29</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>p6</td>
<td>50</td>
<td>5</td>
<td>31</td>
<td>30</td>
<td>30</td>
<td>35</td>
<td>34</td>
<td>3.23</td>
</tr>
<tr>
<td>p7</td>
<td>50</td>
<td>5</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>31</td>
<td>33</td>
<td>0.00</td>
</tr>
<tr>
<td>p8</td>
<td>50</td>
<td>5</td>
<td>31</td>
<td>31</td>
<td>31</td>
<td>31</td>
<td>31</td>
<td>0.00</td>
</tr>
<tr>
<td>p9</td>
<td>50</td>
<td>5</td>
<td>28</td>
<td>28</td>
<td>28</td>
<td>28</td>
<td>28</td>
<td>0.00</td>
</tr>
<tr>
<td>p10</td>
<td>50</td>
<td>5</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>42</td>
<td>0.00</td>
</tr>
<tr>
<td>p11</td>
<td>100</td>
<td>10</td>
<td>19</td>
<td>19</td>
<td>19</td>
<td>30</td>
<td>22</td>
<td>18</td>
</tr>
<tr>
<td>p12</td>
<td>100</td>
<td>10</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>19</td>
</tr>
<tr>
<td>p13</td>
<td>100</td>
<td>10</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>18</td>
</tr>
<tr>
<td>p14</td>
<td>100</td>
<td>10</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>19</td>
</tr>
<tr>
<td>p15</td>
<td>100</td>
<td>10</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>31</td>
<td>23</td>
<td>20</td>
</tr>
<tr>
<td>p16</td>
<td>100</td>
<td>10</td>
<td>20</td>
<td>19</td>
<td>19</td>
<td>21</td>
<td>21</td>
<td>18</td>
</tr>
<tr>
<td>p17</td>
<td>100</td>
<td>10</td>
<td>22</td>
<td>22</td>
<td>22</td>
<td>31</td>
<td>23</td>
<td>20</td>
</tr>
<tr>
<td>p18</td>
<td>100</td>
<td>10</td>
<td>21</td>
<td>20</td>
<td>20</td>
<td>30</td>
<td>22</td>
<td>18</td>
</tr>
<tr>
<td>p19</td>
<td>100</td>
<td>10</td>
<td>21</td>
<td>21</td>
<td>21</td>
<td>30</td>
<td>22</td>
<td>20</td>
</tr>
<tr>
<td>p20</td>
<td>100</td>
<td>10</td>
<td>21</td>
<td>19</td>
<td>19</td>
<td>30</td>
<td>30</td>
<td>17</td>
</tr>
</tbody>
</table>

Table 1: Results with instances S1
<table>
<thead>
<tr>
<th>Prob</th>
<th>n</th>
<th>p</th>
<th>Opt.</th>
<th>LB</th>
<th>Log. Dual</th>
<th>OP</th>
<th>% gap</th>
<th>CPU time LB</th>
<th>CPU time exact</th>
</tr>
</thead>
<tbody>
<tr>
<td>G1</td>
<td>100</td>
<td>5</td>
<td>94</td>
<td>93</td>
<td>93</td>
<td>95</td>
<td>101</td>
<td>1.08</td>
<td>131.24</td>
</tr>
<tr>
<td>G2</td>
<td>100</td>
<td>5</td>
<td>94</td>
<td>93</td>
<td>93</td>
<td>95</td>
<td>94</td>
<td>1.08</td>
<td>83.77</td>
</tr>
<tr>
<td>G3</td>
<td>100</td>
<td>10</td>
<td>83</td>
<td>59</td>
<td>59</td>
<td>116</td>
<td>87</td>
<td>4.68</td>
<td>43.10</td>
</tr>
<tr>
<td>G4</td>
<td>100</td>
<td>10</td>
<td>81</td>
<td>81</td>
<td>116</td>
<td>87</td>
<td>58</td>
<td>3.70</td>
<td>43.10</td>
</tr>
<tr>
<td>G5</td>
<td>150</td>
<td>10</td>
<td>95</td>
<td>93</td>
<td>136</td>
<td>97</td>
<td>93</td>
<td>2.15</td>
<td>43.10</td>
</tr>
<tr>
<td>G6</td>
<td>150</td>
<td>10</td>
<td>96</td>
<td>94</td>
<td>136</td>
<td>97</td>
<td>93</td>
<td>2.15</td>
<td>43.10</td>
</tr>
<tr>
<td>G7</td>
<td>150</td>
<td>15</td>
<td>89</td>
<td>86</td>
<td>97</td>
<td>97</td>
<td>97</td>
<td>4.99</td>
<td>3.49</td>
</tr>
<tr>
<td>G8</td>
<td>150</td>
<td>15</td>
<td>89</td>
<td>88</td>
<td>88</td>
<td>92</td>
<td>92</td>
<td>3.99</td>
<td>3.49</td>
</tr>
</tbody>
</table>

Table 2: Results with instances S2
<table>
<thead>
<tr>
<th>Prob</th>
<th>n</th>
<th>p</th>
<th>Opt.</th>
<th>Lag.Dual</th>
<th>OP</th>
<th>% gap</th>
<th>CPU time LB</th>
<th>CPU time exact</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>D1</td>
<td>D2</td>
<td>LB</td>
<td>D1</td>
<td>D2</td>
</tr>
<tr>
<td>SJC1</td>
<td>100</td>
<td>10</td>
<td>364.73</td>
<td>350.04</td>
<td>350.04</td>
<td>495.14</td>
<td>316.12</td>
<td>4.19</td>
</tr>
<tr>
<td>SJC2</td>
<td>200</td>
<td>15</td>
<td>304.14</td>
<td>302.83</td>
<td>302.83</td>
<td>314.43</td>
<td>301.01</td>
<td>0.43</td>
</tr>
<tr>
<td>SJC3a</td>
<td>300</td>
<td>25</td>
<td>278.96</td>
<td>275.18</td>
<td>275.18</td>
<td>508.04</td>
<td>290.01</td>
<td>275.07</td>
</tr>
<tr>
<td>SJC3b</td>
<td>300</td>
<td>30</td>
<td>253.71</td>
<td>248.71</td>
<td>246.52</td>
<td>508.04</td>
<td>290.01</td>
<td>245.12</td>
</tr>
<tr>
<td>SJC4a</td>
<td>402</td>
<td>30</td>
<td>284.68</td>
<td>277.03</td>
<td>277.03</td>
<td>560.15</td>
<td>320.26</td>
<td>276.00</td>
</tr>
<tr>
<td>SJC4b</td>
<td>402</td>
<td>40</td>
<td>239.38</td>
<td>237.32</td>
<td>237.32</td>
<td>560.15</td>
<td>258.81</td>
<td>237.00</td>
</tr>
</tbody>
</table>

Table 3: Results with instances S3
<table>
<thead>
<tr>
<th>Prob</th>
<th>n</th>
<th>p</th>
<th>#rad</th>
<th>range</th>
<th>#fix</th>
<th>ADF</th>
<th>OP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>D1</td>
<td>D2</td>
</tr>
<tr>
<td>p1</td>
<td>50</td>
<td>5</td>
<td>116</td>
<td>119</td>
<td>116</td>
<td>116</td>
<td>1</td>
</tr>
<tr>
<td>p2</td>
<td>50</td>
<td>5</td>
<td>113</td>
<td>131</td>
<td>113</td>
<td>113</td>
<td>1</td>
</tr>
<tr>
<td>p3</td>
<td>50</td>
<td>5</td>
<td>110</td>
<td>118</td>
<td>110</td>
<td>110</td>
<td>1</td>
</tr>
<tr>
<td>p4</td>
<td>50</td>
<td>5</td>
<td>114</td>
<td>122</td>
<td>114</td>
<td>114</td>
<td>1</td>
</tr>
<tr>
<td>p5</td>
<td>50</td>
<td>5</td>
<td>113</td>
<td>116</td>
<td>113</td>
<td>113</td>
<td>1</td>
</tr>
<tr>
<td>p6</td>
<td>50</td>
<td>5</td>
<td>110</td>
<td>115</td>
<td>110</td>
<td>110</td>
<td>1</td>
</tr>
<tr>
<td>p7</td>
<td>50</td>
<td>5</td>
<td>110</td>
<td>120</td>
<td>110</td>
<td>110</td>
<td>1</td>
</tr>
<tr>
<td>p8</td>
<td>50</td>
<td>5</td>
<td>117</td>
<td>130</td>
<td>117</td>
<td>117</td>
<td>1</td>
</tr>
<tr>
<td>p9</td>
<td>50</td>
<td>5</td>
<td>108</td>
<td>121</td>
<td>108</td>
<td>108</td>
<td>1</td>
</tr>
<tr>
<td>p10</td>
<td>50</td>
<td>5</td>
<td>115</td>
<td>123</td>
<td>115</td>
<td>115</td>
<td>1</td>
</tr>
<tr>
<td>p11</td>
<td>100</td>
<td>10</td>
<td>121</td>
<td>123</td>
<td>110</td>
<td>118</td>
<td>1</td>
</tr>
<tr>
<td>p12</td>
<td>100</td>
<td>10</td>
<td>124</td>
<td>125</td>
<td>124</td>
<td>124</td>
<td>1</td>
</tr>
<tr>
<td>p13</td>
<td>100</td>
<td>10</td>
<td>126</td>
<td>131</td>
<td>125</td>
<td>126</td>
<td>1</td>
</tr>
<tr>
<td>p14</td>
<td>100</td>
<td>10</td>
<td>123</td>
<td>126</td>
<td>113</td>
<td>123</td>
<td>1</td>
</tr>
<tr>
<td>p15</td>
<td>100</td>
<td>10</td>
<td>125</td>
<td>129</td>
<td>114</td>
<td>122</td>
<td>2</td>
</tr>
<tr>
<td>p16</td>
<td>100</td>
<td>10</td>
<td>114</td>
<td>120</td>
<td>112</td>
<td>111</td>
<td>2</td>
</tr>
<tr>
<td>p17</td>
<td>100</td>
<td>10</td>
<td>125</td>
<td>128</td>
<td>116</td>
<td>124</td>
<td>1</td>
</tr>
<tr>
<td>p18</td>
<td>100</td>
<td>10</td>
<td>123</td>
<td>123</td>
<td>113</td>
<td>121</td>
<td>3</td>
</tr>
<tr>
<td>p19</td>
<td>100</td>
<td>10</td>
<td>122</td>
<td>130</td>
<td>113</td>
<td>121</td>
<td>1</td>
</tr>
<tr>
<td>p20</td>
<td>100</td>
<td>10</td>
<td>123</td>
<td>126</td>
<td>112</td>
<td>112</td>
<td>3</td>
</tr>
<tr>
<td>G1</td>
<td>100</td>
<td>5</td>
<td>116</td>
<td>186</td>
<td>114</td>
<td>108</td>
<td>2</td>
</tr>
<tr>
<td>G2</td>
<td>100</td>
<td>5</td>
<td>116</td>
<td>186</td>
<td>114</td>
<td>115</td>
<td>2</td>
</tr>
<tr>
<td>G3</td>
<td>100</td>
<td>10</td>
<td>116</td>
<td>186</td>
<td>79</td>
<td>108</td>
<td>5</td>
</tr>
<tr>
<td>G4</td>
<td>100</td>
<td>10</td>
<td>116</td>
<td>186</td>
<td>81</td>
<td>110</td>
<td>4</td>
</tr>
<tr>
<td>G5</td>
<td>150</td>
<td>10</td>
<td>153</td>
<td>230</td>
<td>111</td>
<td>150</td>
<td>3</td>
</tr>
<tr>
<td>G6</td>
<td>150</td>
<td>10</td>
<td>153</td>
<td>230</td>
<td>111</td>
<td>150</td>
<td>3</td>
</tr>
<tr>
<td>G7</td>
<td>150</td>
<td>15</td>
<td>153</td>
<td>230</td>
<td>142</td>
<td>142</td>
<td>4</td>
</tr>
<tr>
<td>G8</td>
<td>150</td>
<td>15</td>
<td>153</td>
<td>230</td>
<td>144</td>
<td>149</td>
<td>2</td>
</tr>
<tr>
<td>SCJ1</td>
<td>100</td>
<td>10</td>
<td>4,907</td>
<td>2,725.39</td>
<td>4,351</td>
<td>4,351</td>
<td>40</td>
</tr>
<tr>
<td>SCJ2</td>
<td>200</td>
<td>15</td>
<td>19,303</td>
<td>2,758.71</td>
<td>19,194</td>
<td>19,194</td>
<td>14</td>
</tr>
<tr>
<td>SCJ3a</td>
<td>300</td>
<td>25</td>
<td>43,047</td>
<td>3,375.08</td>
<td>39,499</td>
<td>42,862</td>
<td>46</td>
</tr>
<tr>
<td>SCJ3b</td>
<td>300</td>
<td>30</td>
<td>43,047</td>
<td>3,375.08</td>
<td>41,181</td>
<td>42,524</td>
<td>59</td>
</tr>
<tr>
<td>SCJ4a</td>
<td>402</td>
<td>30</td>
<td>75,930</td>
<td>3,509.67</td>
<td>69,143</td>
<td>75,076</td>
<td>134</td>
</tr>
<tr>
<td>SCJ4b</td>
<td>402</td>
<td>40</td>
<td>75,930</td>
<td>3,509.67</td>
<td>68,454</td>
<td>75,573</td>
<td>41</td>
</tr>
</tbody>
</table>

Table 4: Number of fixed radii and number of radii in $[LB, Opt]$
been previously proposed for $CpCP$.

9 Acknowledgements

This work has been partially supported through grant TIC 2000 1750 006/3 of the Inter-Ministerial Spanish Commission of Science and Technology. The research of the third author has been partially supported by the Departamento de Ingeniería Industrial y Textil, Universidad de las Américas, Puebla, México. These supports are gratefully acknowledged.

References


21