European Journal of Operational Research 307 (2023) 33–47

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

Dynamically second-preferred p-center problem

Yolanda Hinojosa[a,b,*], Alfredo Marín[d], Justo Puerto[a,c]

[a] Instituto de Matemáticas, Universidad de Sevilla (IMUS). Edificio Celestino Mutis, Avda. Reina Mercedes s/n, Sevilla 41012, Spain
[b] Dpto. de Economía Aplicada I, Universidad de Sevilla. Fac. de C. Económicas y Empresariales, Avda. Ramón y Cajal 1, Sevilla 41018, Spain
[c] Dpto. de Estadística e Investigación Operativa, Universidad de Sevilla. Fac. de Matemáticas, Avda. Reina Mercedes s/n, Sevilla 41012, Spain
[d] Dpto. de Estadística e Investigación Operativa, Universidad de Murcia. Fac. de Matemáticas, Campus de Espinardo, Murcia 30100, Spain

ARTICLE INFO

Article history:
Received 14 October 2021
Accepted 20 September 2022
Available online 24 September 2022

Keywords:
Location
p-Center
Integer programming
p-Next center

ABSTRACT

This paper deals with the Dynamically Second-preferred p-center Problem (DSpP). In this problem, customers' preferences and subsets of sites that each customer is willing to accept as service centers are taken into account. It is assumed that centers can fail and, thus, the decision maker is risk-averse and makes his decision taking into account not only the most favourite centers of the customers but also the worst case situation whenever they evaluate their preferred second opportunity. Specifically, the new problem aims at choosing at most p centers so that each demand point can visit at least two acceptable centers and the maximum sum of distances from any demand point to any of its preferred centers plus the distance from any of the preferred centers to any of the centers the user prefers once he is there is minimized. The problem is NP-hard as an extension of the p-next center problem. The paper presents three different mixed-integer linear programming formulations that are valid for the problem. Each formulation uses different space of variables giving rise to some strengthening using valid inequalities and variable fixing criteria that can be applied when valid upper bounds are available. Exact methods are limited so that a heuristic algorithm is also developed to provide good quality solution for large size instances. Finally, an extensive computational experience has been performed to assess the usefulness of the formulations to solve DSpP using standard MIP solvers.

1. Introduction

Location science is a very active discipline within Operations Research (Laporte, Nickel, & Saldanha da Gama, 2019). It studies the optimal location of facilities in order to provide service to a set of demand points under different assumptions. When these problems are defined on discrete settings (the set of potential facility locations is discrete) we talk about discrete location.

There are several classical discrete location problems. Among them, the p-center problem (pCP) aims at selecting p out of n locations where to install centers so that the maximum distance between a user and its closest center is minimized (Daskin, 2000; Elloumi, Labbé, & Pochet, 2004; Kariv & Hakimi, 1979; Mladenovic, Labbé, & Hansen, 2003). This type of objective function, focused on the worst case, has been useful to locate emergency facilities and also as a means of identifying equitable solutions when locating essential services. But, even if the services to be located are not essential services, this type of objective function ensures the location of a service center close to any customer, which will improve the customers' perception of the firm.

In the recent years, several extensions of the pCP and other classical models have been studied to incorporate issues that are present in real life. This includes, for instance, considering facility capacities, facility failures or user preferences.

To the best of our knowledge, the extension of the pCP where facilities have limited capacities (Capacitated p-Center Problem - CpCP) was first considered in Jaeger & Goldberg (1994). In this paper, the authors considered a tree underlying network and assumed that capacities were stated in terms of the number of customers each center can serve. For this problem, they proposed a polynomial time algorithm. Other works considering this type of capacities are (Cornejo Acosta, García Díaz, Menchaca-Méndez, & Menchaca-Méndez, 2020) and (Khuller & Sussmann, 2000). For the case with general capacities, exact algorithms are proposed in Albareda-Sambola, Díaz, & Fernández (2010), Kramer, Iori, & Vidal (2020), Özsoy & Pinar (2006), while heuristic methods can be found, for instance, in Quevedo-Orozco & Ríos-Mercado (2015), Scaparra, Pallotino, & Scutellà (2004).

* Corresponding author.
E-mail address: yhinojos@us.es (Y. Hinojosa).

https://doi.org/10.1016/j.ejor.2022.09.019
0377-2217/

Facility reliability (Herrera, Kalcsics, & Nickel, 2007) is another issue that has attracted the attention of many researchers since the seminal paper (Drezner, 1987). There, a simple heuristic was proposed for the pCP with unreliable facilities. More recent extensions consider alternative issues such as unknown failures (Albareda-Sambola, Hinojosa, & Puerto, 2015b), failure correlations (Berman, Krass, & Menezes, 2013), and demand distribution among facilities (Brimberg, Maier, & Schöbel, 2021).

In Albareda-Sambola, Hinojosa, Marín, & Puerto (2015a) the so-called next p-center problem was introduced and several exact formulations were analyzed. In this problem it is assumed that centers can fail and then, the decision maker locates the facilities taking into account not only the closest center of each customer, but also a close second one. The computational results reported in that paper showed the limitations of exact methods, and so (López-Sánchez, Sánchez-Oro, & Hernández-Díaz, 2019) provided some GRASP and VNS heuristics for this problem. The results in Albareda-Sambola et al. (2015a) have been applied to a well-known problem in software defined networks: the controller placement problem (see e.g. Das, Sridharan, & Gurusamy, 2019 for a recent survey on this matter). Given a network, this problem consists in determining the optimal location of controllers and assignment of the switches to the controllers. The proposal by Killi (2018); Killi & Rao (2016, 2017), related to Albareda-Sambola et al. (2015a), consists in developing a location strategy that plans in advance for failures considering second closest controllers, improving disconnections and drastic increase in latency.

In any realistic situation, the appropriate evaluation of the quality of a candidate solution requires an accurate modeling of the actual system behavior. This includes capturing correctly how demands are satisfied. In some contexts, the service provider has the capacity to decide which of the located facilities gives service to each demand point. However, depending on the considered service, each customer makes his own decision on this respect. This has motivated the inclusion of customers' preferences in some location problems. The first work where these preferences were taken into account is Hanjoul & Peeters (1987), where the Simple Plant Location problem with Order (SPLO) is introduced. In the SPLO, the number of facilities is not defined beforehand, and a fixed cost is incurred for setting each facility. Moreover, customers' preferences are known in advance, and each customer is assumed to be served from its most preferred facility among those that have been located. The work presents a simple heuristic for this problem. The SPLO was tackled later by means of mathematical programming with different formulations and the study of valid inequalities, for instance, in Cánovas, García, Labbé, & Marín (2007), Hansen, Kochetov, & Mladenović (2004), Vasilyev, Klimentova, & Boccia (2013), Vasil'ev, Klimentova, & Kochetov (2009). Even more focus on customers' preferences is placed in the minimum-envy location problem (Espejo, Marín, Puerto, & Rodríguez-Chía, 2009), where a set of facilities is sought yielding an equitable customer allocation according to these preferences.

More recent location works involving customer preferences are essentially focused on providing heuristics. For the problem with fixed costs and uncapacitated facilities, Marić, Stanimirović, & Milenković (2012) proposes a heuristic that combines a reduced and a basic Variable Neighborhood Search and shows that it outperforms other heuristics such as Particle Swarm Optimization and Simulated Annealing. The same methods are improved in Marić, Stanimirović, Milenković, & Djenić (2015) to tackle large scale instances. Heuristics for similar problems involving a fixed number of facilities are also considered in the literature. For instance, Casas-Ramírez & Camacho-Vallejo (2017) proposes a heuristic based on Scatter Search for the p-median problem where assignments are made according to customer preferences, and Díaz, Luna, Camacho-Vallejo, & Casas-Ramírez (2017) explores the use of GRASP and Tabu Search for a problem where a fixed number of facilities is to be set in a context with existing competitors. Again, customers patronize facilities according to their preferences, but only customers having a facility at a given distance can be served.

Preferences have also been considered in combination with other issues. For instance, Casas-Ramírez, Camacho-Vallejo, & Martínez-Salazar (2018) addresses a problem with fixed costs and capacitated facilities. In this bilevel problem, customer assignments are made according to their preferences but taking into account facility capacities. The paper proposes a heuristic based on the Cross Entropy method. A closely related field is that of pricing with preferences, where pairs product-price can be seen as locations and users have preferences on products and a given budget, see for instance (Calvete, Domínguez, Galé, Labbé, & Marín, 2019; Domínguez, Labbé, & Marín, 2021; 2022).

In the current paper facility reliability in combination with customers' preferences has been considered, elaborating upon the model in Albareda-Sambola et al. (2015a). Here, we consider a new problem called the *Dynamically Second-preferred p-center Problem* (DSpP) where each customer has a subset of sites that is willing to accept as service centers and customers' preferences are taken into account. One can understand this problem as a hierarchical decision-making process where the planner (leader) sets the service centers considering the worst case customers' behavior and then, the customers (followers) choose according with their preferences, the most preferred open center and in case of failure the most preferred one from that new location. Since centers can fail, the decision maker assumes a risk-averse attitude so that he evaluates an alternative by the worst case situation, with respect to distances, among the most preferred open centers for the customers. Therefore, this problem aims at choosing at most p centers so that each demand point can visit at least two acceptable centers and the maximum sum of distances from any demand point to any of its preferred centers (among the chosen ones) plus the distance from any of the preferred centers to any of the centers the user prefers once he is there is minimized.

One can find applications of this problem in situations where the decision of placing the service centers and trips are supported (paid) by a planner (leader) but once these centers are located the trips to visit the centers are chosen by the customers (followers). Using this model the planner assumes a conservative (risk-averse) attitude and tries to hedge against misbehavior of customers that may choose the longest trips to maximize their preferences. This model has direct applications, for instance in cash machines and in bike-sharing systems. In this last situation, the dealership company wishes to offer service in a given area (usually a city) and thus, it decides on where to place the dock stations in the strategic level. Once the dock stations are in operation, the users pick the bikes in a dock station and return them in different ones, where upon arrival, it might not have empty slots. This situation leads the user to move to his/her most preferred station from his/her current position. All these operations imply to the dealership company extra costs because it has to relocate the bikes in their original stations. This is the reason why the dealership company wants to minimize the maximum sum of distances of any pair of most preferred pairs of sequential dock stations.

The contributions of our paper are the following. First of all, we present three different mixed-integer linear programming formulations for the problem. We strengthen the formulations using valid inequalities and variable fixing criteria (that can be applied when valid upper bounds are available). A heuristic algorithm is developed to obtain valid upper bounds and to provide good quality solutions for large size instances. Finally, a computational experience has been performed to compare their utility to solve DSpP using standard solvers for MIP.

The rest of the paper is organized as follows. Section 2 states the problem, sets the notation and gives an example clarifying the details of feasible solutions of the problems. Section 3 presents a first formulation with 3-indexed variables which is the most intuitive at the price of losing efficiency as will be shown in our computational experiments. Section 4 introduces a straight formulation for (DSpP) which makes use of only one index variables. Section 5 develops another valid formulation based on radius variables similar to those already used in García, Labbé, & Marín (2011), Marín, Nickel, Puerto, & Velten (2009), Marín, Nickel, & Velten (2010). The most efficient of our three formulations can handle medium size instances with up to 150 customers. Due to the difficulty of solving instances of larger sizes, we develop in Section 6 a heuristic procedure providing good quality solutions of (DSpP). These solutions can be also used to feed the exact methods improving their performance by means of the preprocessing described in Section 7. We present in the Section 8 the computational results. The paper ends with a section devoted to conclusions and future research.

2. Problem statement

Let $A=\{1,\dots,n\}$ be a given set of sites where users are situated and which is also the set of candidate sites for locating the centers. For each pair (i,j), $i,j\in A$, let $d_{ij}\geq 0$ be the distance (travel time, cost) from i to j and for each triplet (i,ℓ,j), $i,\ell,j\in A$, let $d_{i\ell j}\geq 0$ be the length of the two-leg path $i\to\ell\to j$, that is, $d_{i\ell j}=d_{i\ell}+d_{\ell j}$.

Each user $i\in A$ has a subset of sites $A_i\subseteq A$ that is willing to accept as service centers. The sites in A_i are ranked by the customer, and we use $j_1<^i_i j_2$ (resp. $j_1>^i_i j_2$, $j_1=^i_i j_2$, $j_1\leq^i_i j_2$, $j_1\geq^i_i j_2$) to denote that i prefers center j_1 more than (resp. less than, the same as, more or the same, less or the same) he prefers center j_2. User i will try to visit one of his preferred installed service centers but, after arriving at it, it could be closed due to a failure and then customer i can change his list of preferences once he has moved to the failed center. Then we use $j_1<^\ell_i j_2$ (resp. $j_1>^\ell_i j_2$, $j_1=^\ell_i j_2$, $j_1\leq^\ell_i j_2$, $j_1\geq^\ell_i j_2$) to denote that i, after noticing that center ℓ failed, prefers center j_1 more than (resp. less than, the same as, more or the same, less or the same) he prefers center j_2, for all j_1, j_2, $\ell\in A_i$.

We assume that "$\leq^\ell_i$" defines a weak order on the set A_i for each $i\in A$, $\ell\in A_i$. Furthermore, $=^\ell_i$ is an equivalence relation which defines a partition $\mathscr{A}^\ell_i=\{A^\ell_i(1),\dots,A^\ell_i(n^\ell_i)\}$ of the set A_i such that (i) $j_1,j_2\in A^\ell_i(t)$ if $j_1=^\ell_i j_2$ and (ii) $j_1\in A^\ell_i(t)$, $j_2\in A^\ell_i(t')$ with $t<t'$ when $j_1<^\ell_i j_2$. Then, for a given user i and center ℓ, $<^\ell_i$ defines a total order on the set of the equivalence classes $\mathscr{A}^\ell_i$.

Additionally, let p be the number of centers which can be installed among the n possibilities.

The problem, named the *Dynamically Second-preferred p-center Problem* (DSpP), is to choose at most p elements of A so as each element can visit at least two acceptable centers and the maximum sum of distances from any element in A to any of its preferred centers (among the chosen ones) plus the distance from any of the preferred centers to any of the centers the user prefers once he is there is minimized. Note that we assume that the user can visit any of the most preferred centers and, in case of failure, any of the most preferred centers from his position, and the objective is to minimize the maximum of the distances travelled by the users.

We formalize the problem as

$$\nu(\text{DSpP}) := \min_{\substack{Q\subset A\\ |Q|\leq p}} \max_{i\in A}\{\max_{\ell\in\arg\min_{A_i\cap Q}}\{d_{i\ell}+\max_{j\in\arg\min^{\ell}_{A_i\cap Q}}\{d_{\ell j}\}\}\}$$

where "arg min" is the set of elements of $A_i\cap Q$ most preferred by i, and "arg min$^\ell$" is the set of elements of $A_i\cap Q$ most preferred by i when i is at position ℓ.

Example 2.1. Consider six locations (users and candidate centers) 1,...,6 on the real line sited at 0, 1, 2, 5, 8 and 11, respectively (see Figure 1A), $p=4$ and Euclidean distances between the points. Assume that when any user i is in a given position, he first prefers his own position, secondly the closest site at the right and the closest site at the left (indifferently, and independently of the distance), then the second closest sites and so on. Also assume that user 1 rejects location 3 (sited at 2) and user 4 rejects location 1 (sited at 0). With the introduced notation we have $A=\{1,2,3,4,5,6\}$,

$$(d_{ij})=\begin{pmatrix} 0 & 1 & 2 & 5 & 8 & 11\\ 1 & 0 & 1 & 4 & 7 & 10\\ 2 & 1 & 0 & 3 & 6 & 9\\ 5 & 4 & 3 & 0 & 3 & 6\\ 8 & 7 & 6 & 3 & 0 & 3\\ 11 & 10 & 9 & 6 & 3 & 0\end{pmatrix}$$

and, for instance, $A_4=\{2,3,4,5,6\}$, $n^3_4=4$ and $\mathscr{A}^3_4=\{A^3_4(1)=\{3\},A^3_4(2)=\{2,4\},A^3_4(3)=\{5\},A^4_4(4)=\{6\}\}$.

A feasible solution to problem DSpP is to locate centers 1, 3, 5, 6 sited at 0, 2, 8 and 11 respectively. We see in Figure 1B-1G the routes that users 1 – 6 could choose. Black-filled nodes are sites that the user rejects and dashed lines represent maximum paths traveled by the users. For instance, user 4 sited at 5 (Figure 1E) could choose the point at the left, center 3 located at 2 and, in case of failure, he would go to center 5 located at 8 (he does not want to use center 1 located at 0). But he could also choose the point at the right, center 5 located at 8 and, in case of failure, he would go to center 6 located at 11. The maximum distance that could be traveled by this user is 9 (from point 4 to center 3 and then from center 3 to center 5). In fact, this is the maximum distance that any user could travel and the objective value of this feasible solution.

In order to summarize the information about the preferences of the users, we will use an $n\times n$ matrix for every user. Every row of this matrix refers to a site where the customer could be, either at the beginning or after noticing that the first choice failed. Columns are also associated with sites. A column with a given entry is preferred to columns with greater numbers, and is equally preferred to columns with the same number (see e.g., the matrix in Example 2.2). A sign "-" in an entry means that the user is not willing to use that site.

Example 2.2. Consider the situation described in Example 2.1 and user 4 located at 5. A matrix of preferences that is compatible with the given example is

$$\begin{pmatrix} - & - & - & - & - & -\\ - & 1 & 2 & 3 & 4 & 5\\ - & 2 & 1 & 2 & 3 & 4\\ - & 3 & 2 & 1 & 2 & 3\\ - & 4 & 3 & 2 & 1 & 2\\ - & 5 & 4 & 3 & 2 & 1\end{pmatrix}.$$

Now, we slightly modify the preferences of this example to show that, contrary to what is usual in location problems, opening more centers does not imply smaller objective values. Consider again Example 2.1 but reverse the preferences (assume that greater numbers mean more preferred sites). Table 1 reports the optimal solutions for different values of p (assuming that the decision maker is forced to open exactly p centers). Columns 2 to 6 refer to the number of centers forced to be open. Second row states the opened centers in each optimal solution, whereas rows 3 to 8 state the first and second centers for each user that give the maximum total distance. For instance, row 4 refers to the second user and in the second column (5,4) means that when exactly two centers are opened, the route of maximum length that user 2 will choose is the one from the position of user 2 to center 5 and then

Table 1
Optimal solution of Example 2.2.

p	2	3	4	5	6
Centers	4,5	1,2,3	1,2,3,4	1,2,3,4,5	1,2,3,4,5,6
User 1	(5,4)	(2,1)	(4,1)	(5,1)	(6,1)
User 2	(5,4)	(1,3)	(4,1)	(5,1)	(6,1)
User 3	(5,4)	(1,3)	(1,4)	(5,1)	(6,1)
User 4	(5,4)	(2,3)	(2,4)	(2,5)	(6,2)
User 5	(4,5)	(1,3)	(1,4)	(1,5)	(1,6)
User 6	(4,5)	(1,3)	(1,4)	(1,5)	(1,6)
Optimal value	11	13	16	19	22

to center 4. Notice that the optimal value of the instance increases with the number of opened centers (see last row in Table 1).

3. Three-indexed formulation

Two-stage location problems (see e.g. Landete & Marín, 2009), among others, have been formulated by means of binary variables associated with the possible routes from each origin through the plants/hubs to the final destination. In our case, a route is given by three elements: site i, first center ℓ and second center j. Thus we introduce binary variables $x_{i\ell j}$ to indicate whether the first and second centers of user i giving the path of maximum length are ℓ and j, respectively. Note that these variables are only defined for $\ell \neq j$ and $\ell \leq_i^i j$, since the remaining indices represent paths that cannot be used in any feasible solution. For each $i \in A$ we denote by I_i this set of indices, i.e., $I_i = \{(\ell, j): \ \ell \in A_i, \ j \in A_i, \ j \neq \ell, \ \ell \leq_i^i j\}$.

With these variables, plus the standard variables in location analysis

$$y_i = \begin{cases} 1 & \text{if a center is located at site } i, \\ 0 & \text{otherwise,} \end{cases} \qquad \forall i \in A,$$

plus a continuous variable z for the objective function, the problem is formulated as follows.

$$\text{(T)} \quad \min \ z$$

$$\text{s.t.} \quad \sum_{i \in A} y_i \leq p \tag{1}$$

$$\sum_{(\ell,j)\in I_i} x_{i\ell j} = 1 \qquad \forall i \in A, \tag{2}$$

$$\sum_{j:(\ell,j)\in I_i} x_{i\ell j} + \sum_{j:(j,\ell)\in I_i} x_{ij\ell} \leq y_\ell \qquad \forall i \in A, \ell \in A_i, \tag{3}$$

$$y_\ell + \sum_{\substack{(h,j)\in I_i: \ h\neq\ell, \\ \ell <_i^h j \ \text{or} \ \ell <_i^i h}} x_{ihj} \leq 1 \qquad \forall i \in A, \ell \in A_i, \tag{4}$$

$$y_j + \sum_{\substack{(\ell,j')\in I_i: \ell\neq j, \\ (j <_i^\ell j') \ \text{or} \ ((j =_i^\ell j') \wedge (d_{\ell j'} < d_{\ell j}))}} x_{i\ell j'} \leq 1 \quad \forall i \in A, j \in A_i, \tag{5}$$

$$y_\ell + y_j + \sum_{\substack{(\ell',j')\in I_i: \ \ell' =_i^i \ell, \\ d_{i\ell' j'} < d_{i\ell j}}} x_{i\ell' j'} \leq 2 + \sum_{\substack{h\in A_i: \\ (h\neq\ell)\wedge(h <_i^\ell j)}} y_h \qquad \forall i \in A, \ (\ell, j) \in I_i \tag{6}$$

$$z \geq \sum_{(\ell,j)\in I_i} d_{i\ell j} x_{i\ell j} \qquad \forall i \in A, \tag{7}$$

$$x_{i\ell j} \in \{0, 1\} \qquad \forall i, \ (\ell, j) \in I_i. \tag{8}$$

$$y_i \in \{0, 1\} \qquad \forall i \in A. \tag{9}$$

Constraint (1) states that at most p centers are open. Constraints (2) force each site to be assigned a two-leg trip that is consistent with the preference relationship, and (3) ensures that those paths go through two sites where centers have been set forbidding the use of non-opened centers. To ensure the preferences of users are satisfied, constraints (4) forbid, when center sited at ℓ is open, assigning user i to a center that is less preferred than site ℓ as first or second center. Constraints (5) and (6) enforce the correct computation of the objective function. On the one hand, (5) forbids *second* centers less preferred than j for i from ℓ whenever j is open and state that in case there exist two *second* centers, j and j', with the same preference for i from ℓ then i must choose the furthest one (to ensure the worst case behavior of the objective function in case of ties). On the other hand, (6) forbid paths of the form $i \to \ell' \to j'$, whenever centers ℓ and j are set, ℓ' is equally preferred to ℓ for i, none of the centers most preferred than j for i once in ℓ is open and the length $d_{i\ell' j'}$ of the path $i \to \ell' \to j'$ is less than the length $d_{i\ell j}$ of the path $i \to \ell \to j$.

Finally, constraints (7) determine the length of the longest path and constraints (8) and (9) define the domain of the decision variables.

Note that if constraints (8) are relaxed to $0 \leq x_{i\ell j} \leq 1$ variables will still take integer values in some optimal solution to the problem. Indeed, if there exists an optimal solution of the problem with two different fractional variables $0 < x_{i\ell j} < 1$ and $0 < x_{i\ell' j'} < 1$, constraints (3) force $y_\ell = y_j = y_{\ell'} = y_{j'} = 1$. Then, by constraints (4), (5) and (6), centers ℓ, ℓ', j and j' must satisfy ($\ell =_i^i \ell'$, $j =_i^{\ell'} j'$ and $j =_i^\ell j'$), ($d_{\ell j} = d_{\ell j'}$ and $d_{\ell' j} = d_{\ell' j'}$) and ($d_{i\ell j} = d_{i\ell' j'}$) respectively. Under these conditions, it is clear that taking $x_{i\ell j} = 1$ and $x_{i\ell' j'} = 0$ or vice versa gives an alternative optimal solution of the problem.

A stronger version of Constraints (6) can be used to improve formulation (T):

$$y_\ell + y_j + \sum_{\substack{(\ell',j')\in I_i: \\ (\ell' =_i^i \ell)\wedge(d_{i\ell' j'} < d_{i\ell j}) \ \text{or} \\ \ell <_i^i \ell' \ \text{or} \\ (\ell\neq\ell')\wedge(\ell'\neq j)\wedge(\ell =_i^i \ell')\wedge(j <_i^{\ell'} j') \ \text{or} \\ (\ell =_i^i \ell')\wedge(\ell <_i^{\ell'} j')\wedge(j'\neq\ell')\wedge(\ell\neq\ell')\wedge(j\neq j')}} x_{i\ell' j'} \leq 2 + \sum_{\substack{h\in A_i: \\ (h\neq\ell)\wedge(h <_i^\ell j)}} y_h \qquad \forall i \in A, \ (\ell, j) \in I_i. \tag{10}$$

In addition to the forbidden paths in (6), constraints (10) forbid paths of the form $i \to \ell' \to j'$ with ℓ' less preferred to ℓ for i, or ℓ' equally preferred to ℓ for i but j' less preferred to j for i from ℓ', or ℓ' equally preferred to ℓ for i but j' less preferred to ℓ for i from ℓ'.

4. Straight formulation

The following formulation aims at representing directly the objective function using only location variables. To this end, we will use again the set of location variables previously defined:

$$y_i = \begin{cases} 1 & \text{if a center is located at site } i, \\ 0 & \text{otherwise,} \end{cases} \qquad \forall i \in A,$$

plus a continuous variable z for the objective value. Using these variables our problem can be formulated as follows.

$$\text{(S)} \qquad \min \ z \tag{11}$$

$$\text{s.t.} \quad (1), (9)$$

$$\sum_{s\in A_i} y_s \geq 2 \qquad \forall i \in A, \tag{12}$$

$$z \geq d_{i\ell j}\Big(y_\ell + y_j - 1 - \sum_{s\in A_i: \ s <_i^i \ell} y_s - \sum_{s:(\ell,s)\in I_i \wedge (s <_i^\ell j)} y_s\Big) \qquad \forall i \in A, (\ell, j) \in I_i. \tag{13}$$

Constraints (12) guarantee that each customer will be able to attend at least two centers. The objective function to be minimized in (11) is bounded from below in constraints (13). Namely, (13) makes z to take a value greater than or equal to $d_{i\ell j}$ if all y_s variables with s in the set

$$\{s \in A_i : \; s <_i^i \ell\} \cup \{s : (\ell, s) \in I_i, s <_i^\ell j\}$$

take value 0. The reason is the following. In the case that $y_s = 1$ for some s in the previous set, user i will never choose the route from i to ℓ and then j because s is preferred by i either as first center or as second center. However, if the route through center s is longer than or equal to the route through ℓ and j, the bound still applies.

There are two cases in which we can assure that the length of the route through s is large enough: (i) if s goes after ℓ in the route and $d_{\ell s} \geq d_{\ell j}$ and (ii) if s is the first stop in the route and d_{is} plus the minimum length to be added after visiting s is greater than or equal to $d_{i\ell j}$. Therefore, we can modify constraints (13) to obtain

$$z \geq d_{i\ell j}(y_\ell + y_j - 1 - \sum_{\substack{s \in A_i: \; s <_i^i \ell \\ d_{is} + \theta_{is} < d_{i\ell j}}} y_s - \sum_{\substack{s:(\ell,s) \in I_i \wedge \\ (s <_i^\ell j) \wedge (d_{\ell s} < d_{\ell j})}} y_s) \; \forall i \in A, (\ell, j) \in I_i \tag{14}$$

where for any $i \in A$ and $s \in A_i$, θ_{is} represents the minimum length from center s to any of the centers that user i prefers once he is in center s, that is,

$$\theta_{is} := \min_{\substack{t \in A_i, \; t \neq s: \\ t \leq_i^s r \;\; \forall r \in A_i, \; r \neq s}} \{d_{st}\}. \tag{15}$$

Finally, we note in passing that the variable z is free although by (13) it always assumes non-negative values.

5. Radius formulation

We develop now a third formulation for the problem, the so-called radius formulation. This kind of formulations for discrete location problems were first used in Cornuéjols, Nemhauser, & Wolsey (1980), Elloumi et al. (2004), and more recently exploited e.g. in García et al. (2011), Marín et al. (2009, 2010).

To this end, some previous operations with the data are needed. We sort the values $d_{i\ell j}$ $\forall i \in A$, $(\ell, j) \in I_i$ in increasing order and ignoring ties. Let

$$\Delta = (\Delta_1, \Delta_2, \ldots, \Delta_g)$$

denote the strictly increasing vector and let G be the set $\{1, \ldots, g\}$.

Example 5.1. Using the same data as in Example 2.1, we get the matrices $(d_{i\ell j})$ for each user $i = 1, 2, 3, 4, 5, 6$ respectively, where each row of these matrices refers to the first visited center, ℓ, and each column refers to the second visited center, j,

$$\begin{pmatrix} - & 1 & - & 5 & 8 & 11 \\ - & - & - & 5 & 8 & 11 \\ - & - & - & - & - & - \\ - & - & - & - & 8 & 11 \\ - & - & - & - & - & 11 \\ - & - & - & - & - & - \end{pmatrix} \begin{pmatrix} - & - & 3 & 6 & 9 & 12 \\ 1 & - & 1 & 4 & 7 & 10 \\ 3 & - & - & 4 & 7 & 10 \\ - & - & - & - & 7 & 10 \\ - & - & - & - & - & 10 \\ - & - & - & - & - & - \end{pmatrix} \begin{pmatrix} - & - & - & - & 10 & 13 \\ 2 & - & - & 5 & 8 & 11 \\ 2 & 1 & - & 3 & 6 & 9 \\ 8 & 7 & - & - & 6 & 9 \\ 14 & - & - & - & - & 9 \\ - & - & - & - & - & - \end{pmatrix}$$

$$\begin{pmatrix} - & - & - & - & - & - \\ - & - & - & - & - & 14 \\ - & 4 & - & - & 9 & 12 \\ - & 5 & 3 & - & 3 & 6 \\ - & 10 & 9 & - & - & 6 \\ - & 16 & - & - & - & - \end{pmatrix} \begin{pmatrix} - & - & - & - & - & - \\ 8 & - & - & - & - & - \\ 8 & 7 & - & - & - & - \\ 8 & 7 & 6 & - & - & - \\ 8 & 7 & 6 & 3 & - & 9 \\ 14 & 13 & 12 & 9 & - & - \end{pmatrix} \begin{pmatrix} - & - & - & - & - & - \\ 11 & - & - & - & - & - \\ 11 & 10 & - & - & - & - \\ 11 & 10 & 9 & - & - & - \\ 11 & 10 & 9 & 6 & - & - \\ 11 & 10 & 9 & 6 & 3 & - \end{pmatrix}$$

and where '–' means that the entry does not satisfy the conditions given above. Then $\Delta = (1, 2, \ldots, 13, 14, 16)$, $g = 15$ and $G = \{1, \ldots, 15\}$.

For this third formulation we will use the same y-variables as before, plus binary variables

$$z_k = \begin{cases} 1 & \text{if the distance that could be travelled} \\ & \quad \text{by some user is at least } \Delta_k, \\ 0 & \text{otherwise,} \end{cases} \qquad \forall k \in G.$$

Using these variables the radius formulation of our problem is

$$(\mathrm{R}) \quad \min \quad \Delta_1 z_1 + \sum_{k=2}^{g} (\Delta_k - \Delta_{k-1}) z_k \tag{16}$$

$$\text{s.t.} \quad (1), (9), (12)$$

$$z_k \leq z_{k-1} \qquad \forall k = 2, \ldots, g \tag{17}$$

$$z_k + \sum_{\substack{s \in A_i: \\ s <_i^i \ell}} y_s + \sum_{\substack{s:(\ell,s) \in I_i \\ \wedge (s <_i^\ell j)}} y_s \geq y_\ell + y_j - 1 \quad \forall i \in A, (\ell, j) \in I_i,$$

$$\forall k \in G : \; \Delta_k = d_{i\ell j}. \tag{18}$$

Here the objective function (16) measures the maximum distance needed by a user if he goes first to his closest center and then to the next closest center, ties broken arbitrarily. The y-variables in the left hand side of (18) correspond with centers that either are preferred by i before ℓ or will be chosen as the second center in the route after ℓ instead of j. If no site in those conditions exists, then z_k will take value 1 provided that ℓ and j are open. Since by (9) the y-variables are binary we do not have to impose any condition on the z_k-variables. Constraints (17) guarantee that if the distance traveled by some user is at least Δ_k, then this user has traveled a distance greater than Δ_{k-1}. These constraints are necessary because constraints (18) are not sufficient to guarantee this fact as we can see in the following example.

Example 5.2. Consider again the data of Example 2.1. Without constraints (17), one optimal solution to (R) for $p = 4$ is to locate centers 3, 4, 5 and 6 (sited in 2, 5, 8 and 11, respectively), and its optimal value is 3. Nevertheless, the correct optimal value of (R) for $p = 4$ is 6 (an optimal solution is given in Fig. 2) and, moreover, the cost of locating centers 3, 4, 5 and 6 is 8 (see Fig. 3). The reason for this error is the following.

When $k = 7$ with $\Delta_7 = 7$, six different possible combinations of values (i, ℓ, j) satisfy $\Delta_k = d_{i\ell j}$ (see Example 2.1), namely (2,3,5), (2,4,5), (3,4,2), (5,3,2), (5,4,2), (5,5,2). The corresponding constraints (18) with right hand side 1 are

$$z_7 + \sum_{\substack{s:(s <_2^2 3) \vee \\ [(3,s) \in I_2 \wedge (s <_2^3 5)]}} y_s \geq y_3 + y_5 - 1 \equiv z_7 + y_2 + y_4 \geq y_3 + y_5 - 1$$

$$z_7 + \sum_{\substack{s:(s <_2^2 4) \vee \\ [(4,s) \in I_2 \wedge (s <_2^4 5)]}} y_s \geq y_4 + y_5 - 1 \equiv z_7 + y_2 + y_3 + y_1 \geq y_4 + y_5 - 1$$

$$z_7 + \sum_{\substack{s:(s <_3^3 4) \vee \\ [(4,s) \in I_3 \wedge (s <_3^4 2)]}} y_s \geq y_4 + y_2 - 1 \equiv z_7 + y_3 + y_5 \geq y_4 + y_2 - 1$$

$$z_7 + \sum_{\substack{s:(s <_5^5 3) \vee \\ [(3,s) \in I_5 \wedge (s <_5^3 2)]}} y_s \geq y_3 + y_2 - 1 \equiv z_7 + y_4 + y_5 + y_6 \geq y_3 + y_2 - 1$$

$$z_7 + \sum_{\substack{s:(s <_5^5 4) \vee \\ [(4,s) \in I_5 \wedge (s <_5^4 2)}} y_s \geq y_4 + y_2 - 1 \equiv z_7 + y_5 + y_3 \geq y_4 + y_2 - 1$$

$$z_7 + \sum_{\substack{s:(s <_5^5 5) \vee \\ [(5,s) \in I_5 \wedge (s <_5^5 2)]}} y_s \geq y_5 + y_2 - 1 \equiv z_7 + y_3 + y_4 + y_6 \geq y_5 + y_2 - 1$$

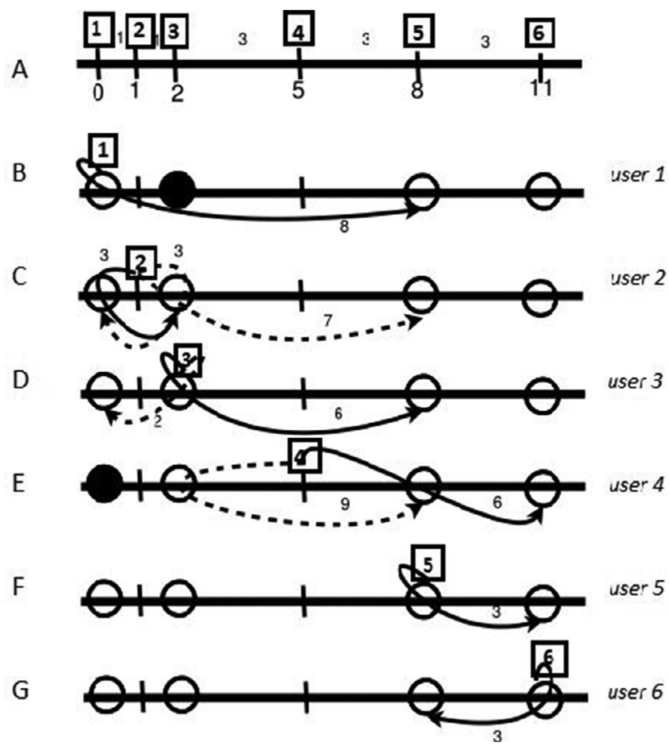

Fig. 1. Example of possible travels of users.

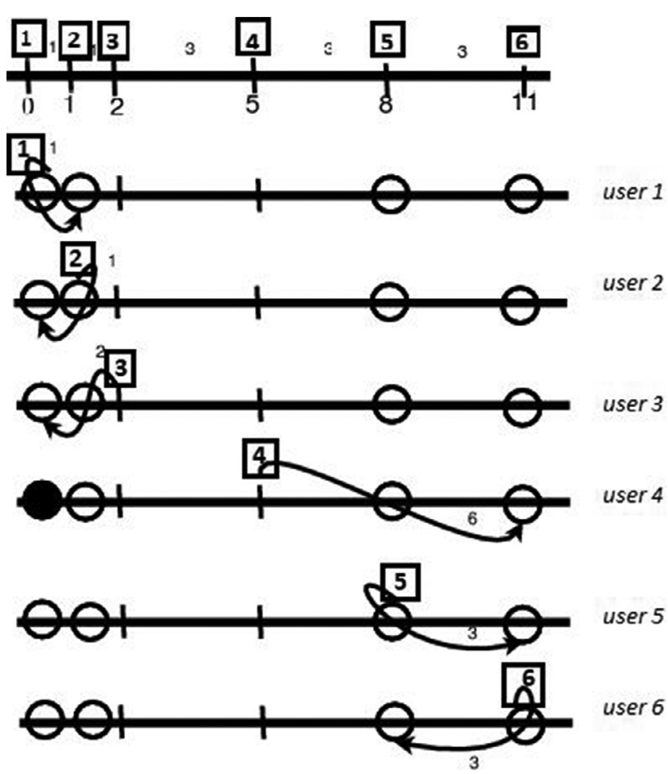

Fig. 2. Optimal solution to Example 2.1 (Fig. 1) for $p = 4$.

which do not force z_7 to take any value since there is always a variable taking value 1 in the left hand side. A similar effect is produced when $k = 1$, 2, 5 and 6, and then the objective function value is $\Delta_8 + \Delta_4 - \Delta_7 - \Delta_2 = 3$, reducing in 3 units the objective value of the solution.

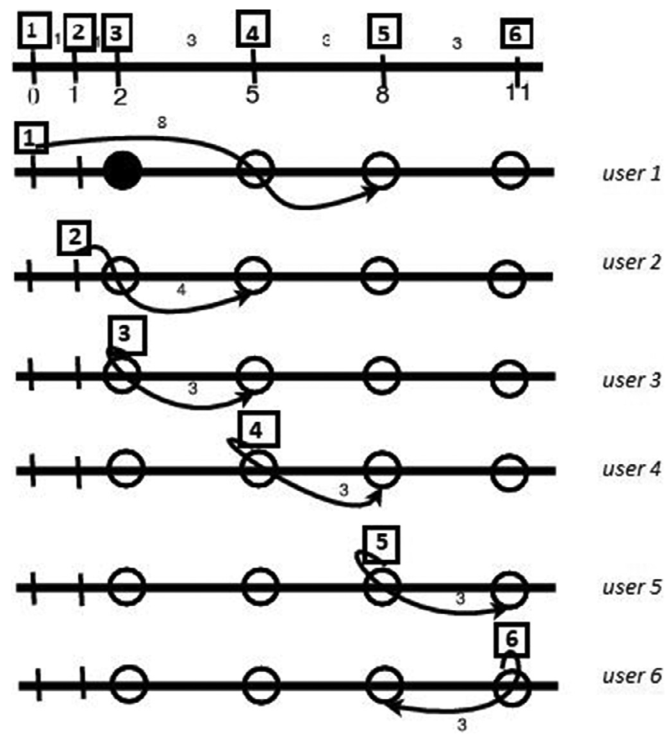

Fig. 3. Solution to Example 2.1 (Fig. 1) corresponding to sites 3-4-5-6.

In what follows, we develop an improvement to be applied to constraints (18) in formulation (R).

Consider a triplet (i, ℓ, j) with $(\ell, j) \in I_i$. Consider also the value of k such that $\Delta_k = d_{i\ell j}$. Constraint (18) for these values makes z_k to take value 1 except when some y_s variable with s in the union of sets

$$\{s \in A_i : \ s <_i^i \ell\} \cup \{s : (\ell, s) \in I_i, s <_i^\ell j\}$$

takes value 1. To improve this constraint, we can reduce the first set of indices in this union to this other set

$$S_1 := \{s \in A_i : \ s <_i^i \ell, \ \{t \in A_i : \ s <_i^i t, \ d_{ist} < \Delta_k\} \neq \emptyset\},$$

in which any center s, preferred by i before ℓ such that $d_{ist} \geq \Delta_k$ for all t preferred by i after s, is excluded. Similarly, the second set can be reduced to this one

$$S_2 := \{s : (\ell, s) \in I_i, s <_i^\ell j, \ d_{i\ell s} < \Delta_k\},$$

in which any center s, chosen as the second center in the route, more preferred by i once i is in ℓ than j verifying $d_{ils} \geq \Delta_k$, is excluded. The resulting constraints (to be used instead of (18)) are

$$z_k + \sum_{s \in S_1 \cup S_2} y_s \geq y_\ell + y_j - 1 \quad \forall i \in A, (\ell, j) \in I_i, \forall k \in G : \ \Delta_k = d_{i\ell j}. \tag{19}$$

Additionally, the following set of valid inequalities can be added to (R).

$$z_k + \sum_{\substack{s \in A_i : \ s <_i^i \ell, \\ d_{is} + \theta_{is} < \Delta_k}} y_s \geq y_\ell \ \forall i \in A, \ \ell \in A_i, \ \forall k : \ d_{i\ell} + \theta_{i\ell} = \Delta_k, \tag{20}$$

where $\theta_{i\ell}$ is calculated as in expression (15).

6. Heuristic procedure

In this section we propose a heuristic approach to obtain an upper bound of the optimal value of DSpP. In order to present it, we assume the following notation.

For any fixed user $i \in A$ we sort the values $d_{i\ell j}$, $\forall(\ell, j) \in I_i$, in increasing order ignoring ties. Let $\Delta^i = (\Delta^i_1, \Delta^i_2, \ldots, \Delta^i_{g^i})$ denote the strictly increasing vector and let P^i_k be the set of pairs giving the value Δ^i_k, that is, $P^i_k = \{(\ell, j) \in I_i : d_{i\ell j} = \Delta^i_k\}$.

A pseudocode of this heuristic is shown in the forthcoming Algorithm 4, that is based on three main steps:

1. Constructing an initial set of open centers, $\mathcal{C}$, verifying that for each user $i \in A$, set $\mathcal{C}$ contains at least two different centers, $\ell, j \in A_i$.
2. Reducing the size of set $\mathcal{C}$.
3. Trying to improve the objective value.

To *construct the initial set* $\mathcal{C}$ we select for each user $i \in A$ a value $\Delta^i_{k_i}$ and we proceed as shown in Algorithm 1. The choice of the

Algorithm 1: Constructing an initial set of open centers $\mathcal{C}$.

input :
- $\mathcal{C} = \emptyset$: Set of open centers
- For each $i \in A$: $\Delta^i_{k_i}$ and $P^i_{k_i}$.

Sort in decreasing order $\Delta^i_{k_i}$. Let $\Delta^{i_r}_{k_{i_r}}$ be the rth sorted value and i_r the index of the corresponding user.

for $r = 1$ *to* $r = n$ **do**
  if *there exists a pair* $(\ell, j) \in P^{i_r}_{k_{i_r}}$ *such that* $\ell \leq^{i_r}_{i_r} \ell' \ \forall \ell' \in \mathcal{C}$
  then
    $\mathcal{C} = \mathcal{C} \bigcup \{\ell\}$
    if $j \leq^{\ell}_{i_r} j' \ \forall j' \in \mathcal{C}$ **then**
      $\mathcal{C} = \mathcal{C} \bigcup \{j\}$
  else
    if $|A_{i_r} \bigcap \mathcal{C}| < 2$ **then**
      $\mathcal{C} = \mathcal{C} \bigcup \{\ell\}$ for some $\ell \in A_{i_r}$ such that $(\ell, j) \in P^{i_r}_{k_{i_r}}$.

output: $\mathcal{C}$.

value $\Delta^i_{k_i}$ for each $i \in A$ can be done in different ways. For instance, once a reference index k is fixed, we can take for each $i \in A$ its corresponding k-index value, Δ^i_k, or we can compute $\Delta^{\max}_k = \max_i \Delta^i_k$ and consider k_i for each $i \in A$ as the index s of value Δ^i_s verifying that $\Delta^i_s \leq \Delta^{\max}_k < \Delta^i_{s+1}$.

Given a set of open centers $\mathcal{C}$ let $z_i(\mathcal{C}) = \max\{d_{i\ell j} : (\ell, j) \in I_i,\ \ell, j \in \mathcal{C},\ \ell \leq^i_i \ell' \ \forall \ell' \in \mathcal{C},\ j \leq^{\ell}_i j' \ \forall j' \in \mathcal{C}\}$ be the distance traveled by the user $i \in A$. If there exists a user $i \in A$ such that the set $\mathcal{C}$ does not contain two centers $\ell, j \in A_i$ then $z_i(\mathcal{C}) = \infty$. And let $z(\mathcal{C}) = \max_i z_i(\mathcal{C})$ be the objective value given the set of open centers $\mathcal{C}$. To *reduce the size of a given set* $\mathcal{C}$ we proceed as shown in Algorithm 2.

Finally, to *improve (whenever this is possible) the objective value* we proceed as shown in Algorithm 3. Given a set of open centers $\mathcal{C}$, let $i_{\max}(\mathcal{C}) \in \arg\max_{i \in A} z_i(\mathcal{C})$ denote a user index that gives the maximum (worst) distance to be traveled and let $i_{2max}(\mathcal{C}) \in \arg\max_{i \in A \setminus \{i_{\max}(\mathcal{C})\}} z_i(\mathcal{C})$ denote a user index that gives the second worst distance.

The heuristic algorithm for the DSpP is described below (Algorithm 4). As mentioned above it consists of a while loop that modifies (augmenting or reducing) the set of open centers trying to improve the objective value.

Algorithm 2: Reducing the size of set $\mathcal{C}$.

input :
- $\mathcal{C}$: Set of open centers.
- $\bar{z} = z(\mathcal{C})$: Current value of the objective function.

for $c \in \mathcal{C}$ **do**
  Compute $z(\mathcal{C} \setminus \{c\})$
  if $z(\mathcal{C} \setminus \{c\}) < \bar{z}$ **then**
    $\bar{z} = z(\mathcal{C} \setminus \{c\})$
    $\bar{c} = c$

output: $\mathcal{C} = \mathcal{C} \setminus \{\bar{c}\}$, $\bar{c}$ and$\bar{z}$

Algorithm 3: Trying to improve the objective value.

input :
- $\mathcal{C}$: Set of open centers.
- $i_{\max}(\mathcal{C})$ and $i_{2max}(\mathcal{C})$.
- $\bar{z} = z_{i_{\max}(\mathcal{C})}(\mathcal{C})$: Current objective value.
- p: maximum number of open centers.

Compute the index $\bar{k}$ such that
$\Delta^{i_{\max}(\mathcal{C})}_{\bar{k}} \leq z_{i_{2max}(\mathcal{C})}(\mathcal{C}) < \Delta^{i_{\max}(\mathcal{C})}_{\bar{k}+1}$.

for $r = \bar{k}$ *to* $r = 1$ **do**
  Select $(\ell, j) \in P^{i_{\max}(\mathcal{C})}_r$ and take $\mathcal{C} = \mathcal{C} \cup \{\ell, j\}$
  while $|\mathcal{C}| > p$ **do**
    $\mathcal{C},\ c_a,\ z_a$ =*Reducing the size of* $\mathcal{C}$
  while $|\mathcal{C}| > 2$ **do**
    $\mathcal{C},\ c_b,\ z_b$=*Reducing the size of* $\mathcal{C}$
    if $z_b < z_a$ **then**
      $z_a = z_b$
    else
      $\mathcal{C} = \mathcal{C} \cup \{c_b\}$
      break (while)
  if $z_a < \bar{z}$ **then**
    $\bar{z} = z_a$
    break (for).

output: $\mathcal{C}$ and $\bar{z}$

7. Preprocessing

Let z_{UB} be the value of the objective function given by Algorithm 4. The following valid inequalities can be added to the three formulations, for all $\ell, j \in A$ such that for some $i \in A$, $(\ell, j) \in I_i$, $\ell \leq^i_i \ell'$ for all $\ell' \in A_i$, $j \leq^{\ell}_i j'$ for all $j' \in A_i$ verifying that $(\ell, j') \in I_i$, and $z_{UB} < d_{i\ell j}$:

$$y_\ell + y_j \leq 1.$$

Additionally, in formulation (T) we can fix to zero the corresponding variables $x_{i\ell j}$ and $x_{ij\ell}$.

On the other hand, if there exists a customer i and a center $\ell \in A_i$, $\ell \leq^i_i s \ \forall s \in A_i$ such that

$$z_{UB} < d_{i\ell} + \min_{\substack{j \neq \ell,\, j \in A_i \\ j \leq^{\ell}_i t \ \ \forall t \neq \ell \in A_i}} d_{\ell j}$$

then one can fix $y_\ell = 0$ in the three formulations.

Regarding the formulation (R), if $\Delta_k > z_{UB}$, then we can fix variable z_k to zero. Note also that $\Delta^{\max}_1 = \max_i \Delta^i_1$ gives a lower bound of the optimal value and then, we can fix to one all the variables z_k such that $\Delta_k \leq \Delta^{\max}_1$.

Similarly, in the formulation (T), if $d_{i\ell j} > z_{UB}$ we can fix to zero variable $x_{i\ell j}$.

Algorithm 4: Heuristic.

input :
- $\mathcal{C} = \emptyset$: Set of open centers
- For each $i \in A$: $\Delta^i_{k_i}$ and $P^i_{k_i}$.
- $\bar{z} = \infty$: Current value of the objective function.
- p: maximum number of open centers.

$\mathcal{C}$=*Constructing an initial set of open centers.*
if $|\mathcal{C}| \le p$ **then**
  Compute $\bar{z} = z(\mathcal{C})$
else
  while $|\mathcal{C}| > p$ **do**
    $\mathcal{C}$, c_a, z_a =*Reducing the size of* $\mathcal{C}$
  while $|\mathcal{C}| > 2$ **do**
    $\mathcal{C}$, c_b, z_b=*Reducing the size of* $\mathcal{C}$
    if $z_b < z_a$ **then**
      $z_a = z_b$
    else
      $\mathcal{C} = \mathcal{C} \cup \{c_b\}$
      break (while)
  $\bar{z} = z_a$
$\mathcal{C}$, z =*Trying to improve the objective function value.***while** $z < \bar{z}$ **do**
  $\bar{z} = z$
  $\mathcal{C}$, z =*Trying to improve the objective function value.*
output: $\mathcal{C}$ and $\bar{z}$.

8. Computational tests

In this section we report on the results of a series of computational tests to assess the usefulness of the formulations discussed in previous sections to solve the DSpP with standard software, and to compare their performances. All experiments were carried out with the Gurobi 9.1.1 optimizer, under a Windows 10 environment in an Intel(R) Core(TM) i7-6700K CPU @ 4.00 GHz 4.01 GHz processor and 32 GB of RAM. Default values were used for all parameters of Gurobi. A CPU time limit of 3600 seconds was set.

To carry out the experiment we generate instances in the following way. Sites in the set A are points randomly placed in a square of size 100×100. Distances (d_{ij}) are rounded Euclidean distances between these points. Each site accepts to be assigned to each other site with probability 0.8. In order to determine the preferences with respect to the accepted sites, we consider two cases:

1. *Random instances*. In this case we generate integer numbers $P(i, \ell, j) \sim \lceil \lfloor \frac{n}{3} \rfloor U(0, 1) \rceil$ so that $j_1 <^{\ell}_{i} j_2$ when $P(i, \ell, j_1) < P(i, \ell, j_2)$ and $j_1 =^{\ell}_{i} j_2$ when $P(i, \ell, j_1) = P(i, \ell, j_2)$.
2. *Euclidean instances*. In order to have a positive correlation between distances and preferences, we initially obtain the values of $P(i, \ell, j)$ sorting in increasing order the distances from a site ℓ to all sites j, removing the sites ℓ and j rejected by i, and assigning, for each i and ℓ, numbers $1, 2 \ldots$ in the given order. Afterwards, we slightly perturbate these values (at the same time creating ties in the preference list) by either adding 3 units to a value with a probability of 0.4 or subtracting 5 units with probability of 0.24. Negative numbers are finally replaced with number 1.

Note that we checked also some instances with different probability distributions that gave rise to similar computational results.

The study is organized in two phases, each presented in one subsection. A last subsection is devoted to discussing the number of open centers and the possibility of distributing them in an alternative way.

8.1. Preliminary results

In this phase n ranges in $\{25, 50, 75\}$, whereas p varies in either $\{5, 10\}$ (when $n = 25$) or $\{5, 10, 20\}$ (when $n \in \{50, 75\}$). The performance of each of the three formulations is tested with and without preprocessing (Prepr $\in$ {Yes, No}) and with and without dispensation of the initial solution provided by Algorithm 4 to the solver (InitSol$\in$ {Yes, No}). Moreover Improv $\in$ {Yes, No} with the meaning depending on the formulation. Specifically, Improv = Yes means (i) constraints (10) are added to formulation (T), (ii) constraints (14) are added to formulation (S) and (iii) constraints (19) and (20) are added to formulation (R). We solve five instances for each type (random, Euclidean), each dimension (n, p), each formulation and each combination of Prepr, InitSol and Improv, totalizing 1920 instances.

The results of the preliminary study are summarized in Figs. 4, 5 and 6 for the random instances and in Figs. 7, 8 and 9 for the Euclidean instances. All instances have been solved up to optimality within the time limit except some instances of the three-indexed formulation with $n = 50$ and $n = 75$, where, either the time limit was exceeded or a feasible solution was not found within the time limit or the flag “Out of memory” was the output of the solver. These figures show the average CPU time required to solve the instances. Clearly, the best formulations are (S) and (R) in all configurations of parameters. For Random instances, it seems that the best combination is Prepr = Yes, Improv = Yes and InitSol = Yes. For Eucliean instances, there are two configurations with similar behaviour, namely (i) Prepr = Yes, Improv = Yes, InitSol = Yes and, (ii) Prepr = No, Improv = Yes and InitSol = No. We will asses the performance of these combinations on the second phase of our computational study devoted to larger sized instances.

8.2. Advanced results

The aforementioned best solution methods are compared in a second phase, using larger instances with $n \in \{100, 125, 150, 175, 200\}$ and $p \in \{5, 10, 20\}$. As before, we consider five instances for each dimension (n, p). We solve a total of 450 instances.

Tables 2 and 3 show the results obtained by using formulations (S) and (R) with preprocessing, initial solution and valid inequalities, for random and Euclidean instances, respectively. On the other hand, the results for Euclidean instances using formulations (S) and (R) only with valid inequalities are given in Table 4. In these tables the information contained in each row refers to average values of 5 instances.

These tables are organized in two different blocks of columns: Gap and CPU Time. The block named “Gap” contains three blocks of two columns, each block reporting different types of gaps. “UBGap” is the (average) percentage gap computed using the objective values of the best solution obtained by the corresponding formulation, say $\bar{v}$, and the feasible solution found using the heuristic, say $\bar{v}_H$. Specifically, “UBGap” is computed as $[\bar{v}_H - \bar{v}]/\bar{v} \times 100$. “MIPGap” is the (average) percentage MIP gap returned by Gurobi when solving the problem with each formulation and “RootGap” is the (average) percentage gap between the best solution obtained by the corresponding formulation and the solution obtained in the root node.

The second big block, named “CPU Time”, is organized in four blocks, the first one of one column and the other three of two columns. “HTime” stands for the CPU time spent when solving the heuristic, “ETime” is the CPU time needed when solving the problem using the different formulations, being “TotalTime”=“HTime”+“ETime” the total time required for solving the problem when preprocessing or initial solution are used. Note that when neither preprocessing nor initial solution are used, To-

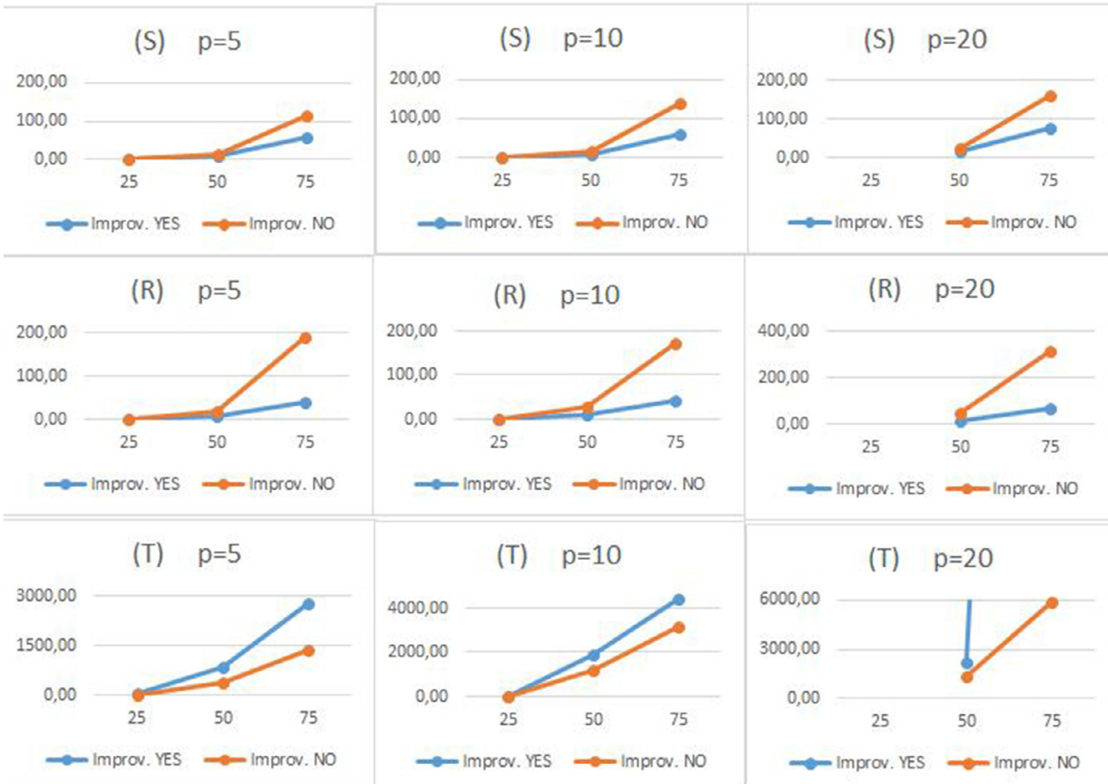

Fig. 4. Comparing the CPU time for random instances varying $Improv \in \{YES, NO\}$.

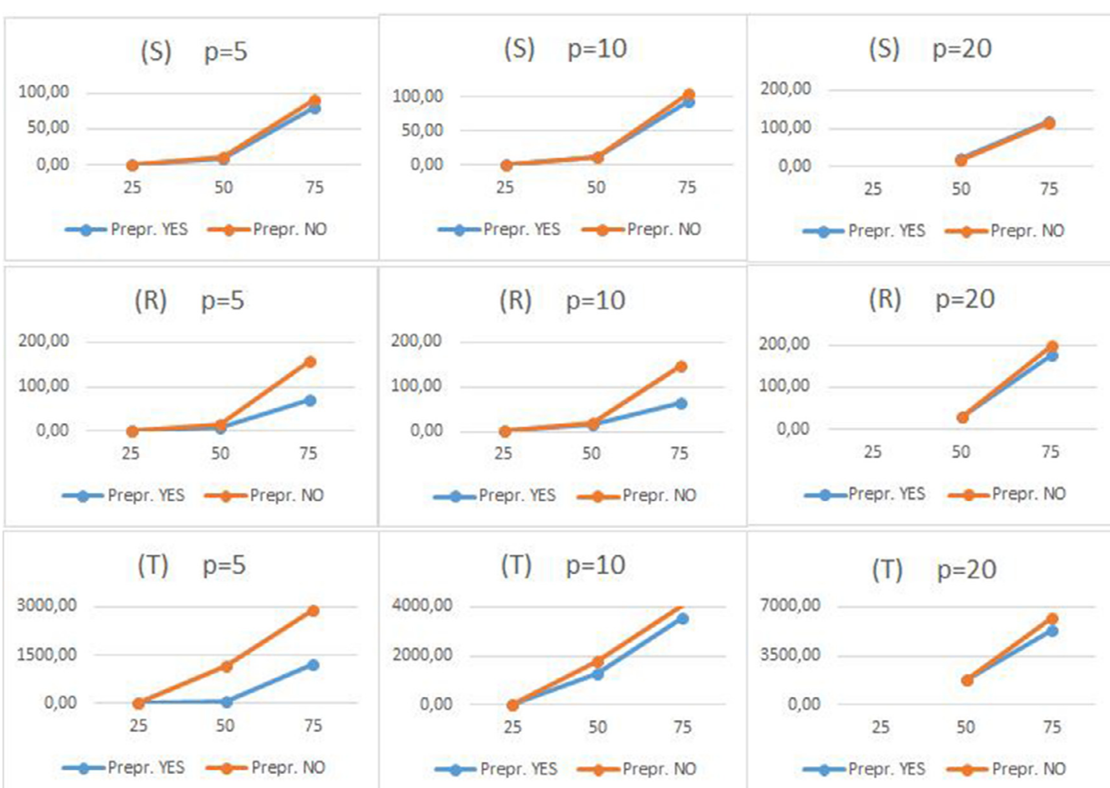

Fig. 5. Comparing the CPU time for random instances varying $Prepr \in \{YES, NO\}$.

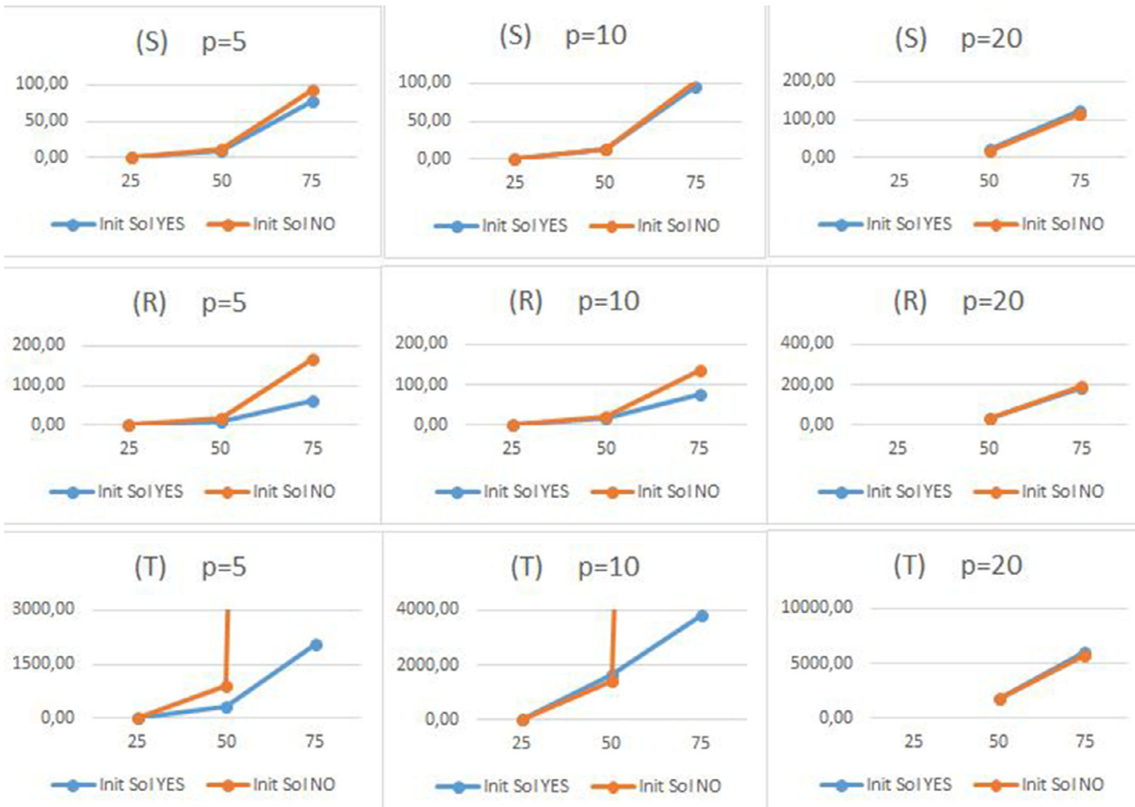

Fig. 6. Comparing the CPU time for random instances varying *Init Sol* $\in \{YES, NO\}$.

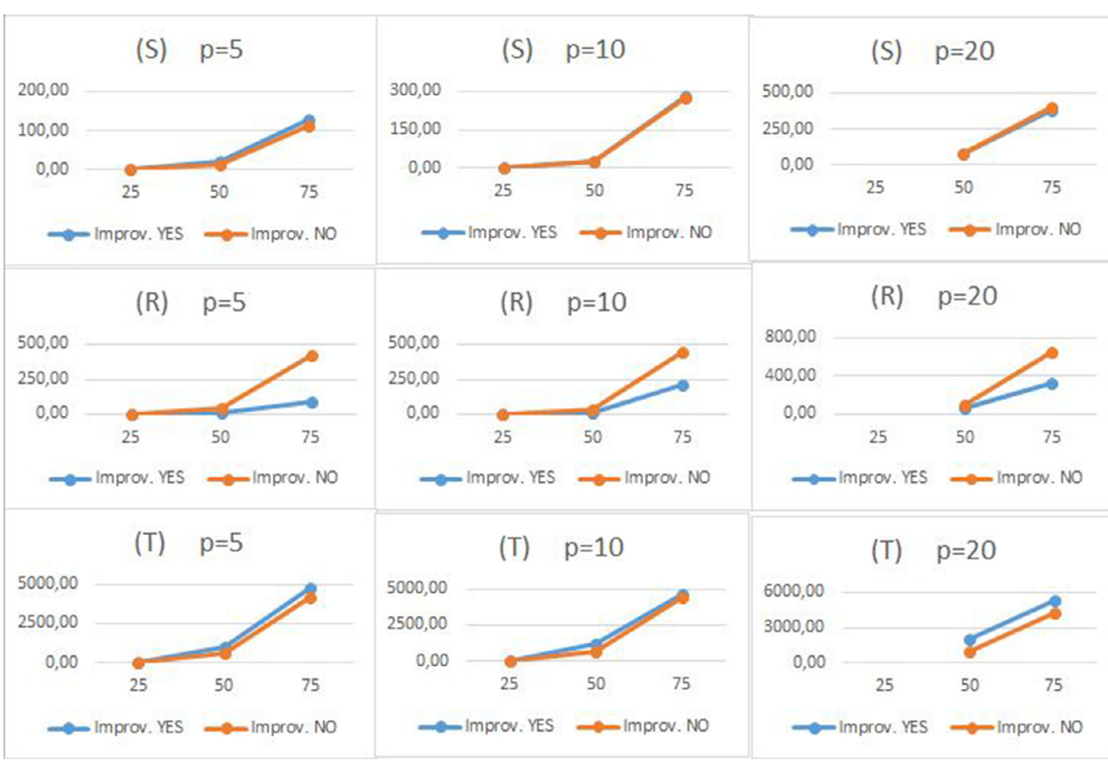

Fig. 7. Comparing the CPU time for Euclidean instances varying *Improv* $\in \{YES, NO\}$.

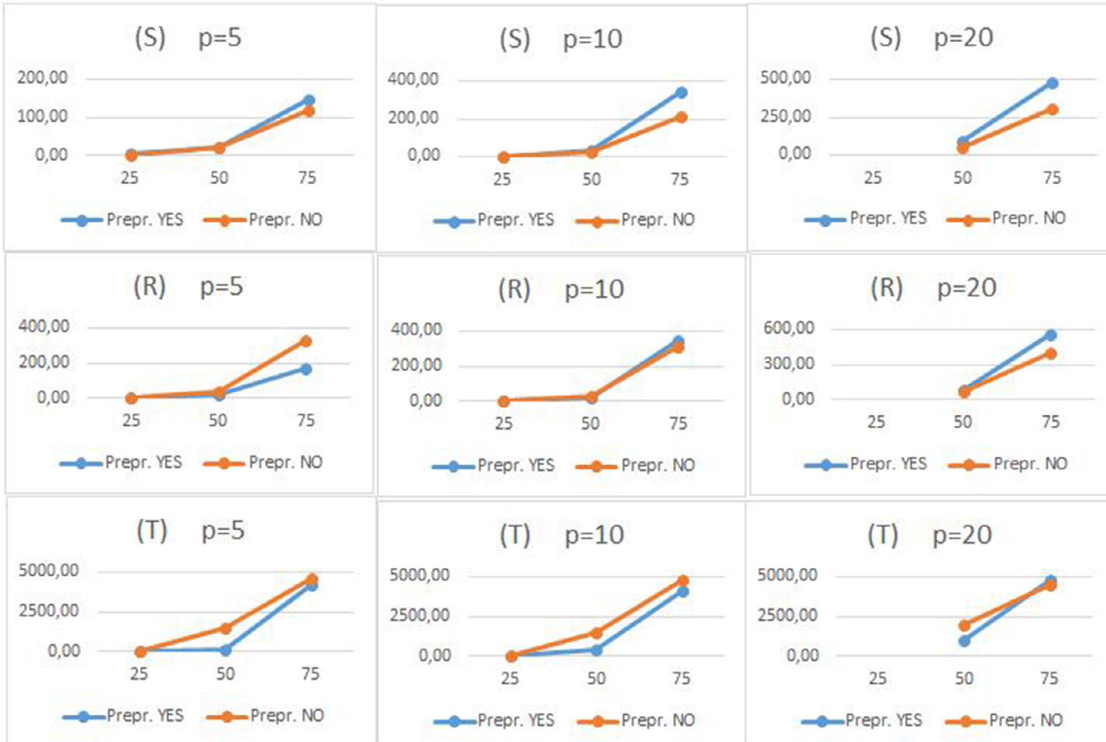

Fig. 8. Comparing the CPU time for Euclidean instances varying *Prepr* $\in \{YES, NO\}$.

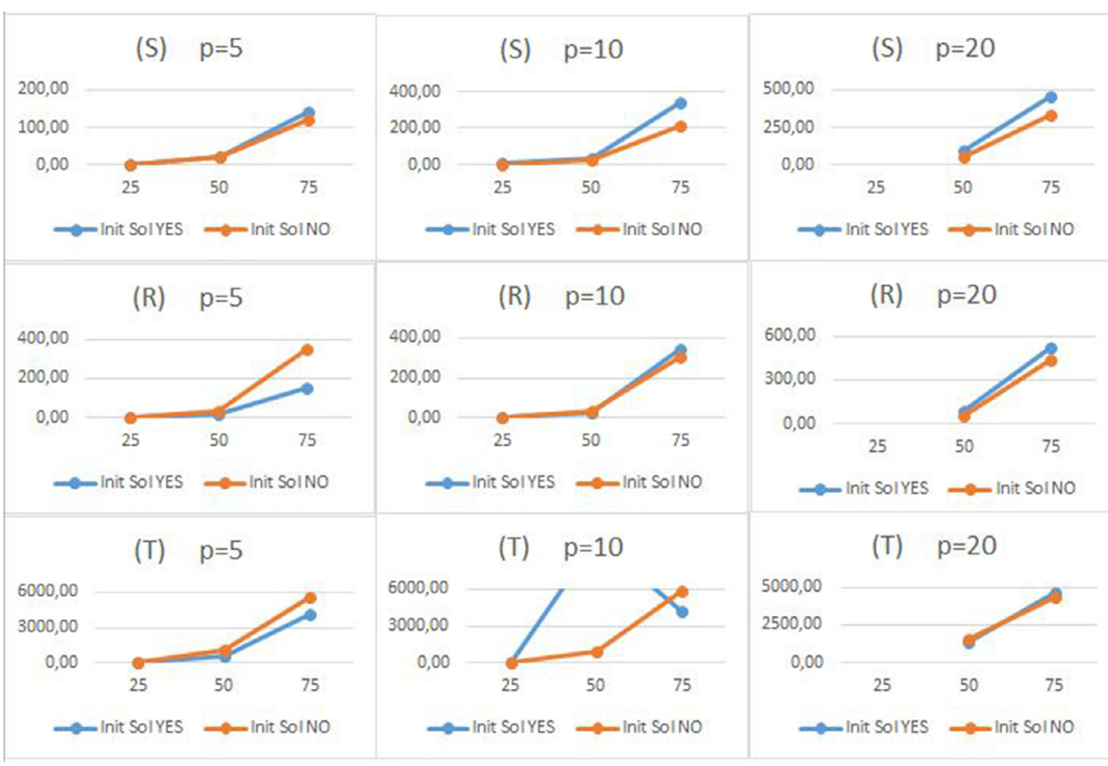

Fig. 9. Comparing the CPU time for Euclidean instances varying *Init Sol* $\in \{YES, NO\}$.

talTime and ETime coincide and we only report on one of them. Finally, “RootTime” stands for the CPU time required to solve the root node of the branching tree.

One can observe in Table 2 (MIPGap) that instances with sizes up to $n = 150$ have been solved up to optimality within the time limit with both formulations. Nevertheless, (R) performs better in terms of CPU time required to solve the instances. For $n = 175$, (R) performs better for $p = 5$ regarding both, Gap and CPU time, but the other way around for $p = 10, 20$. For larger sizes ($n = 200$) only (R) can solve instances, always for $p = 5$.

Table 3 compares the results obtained on Euclidean instances with all the potential improvements. These instances are harder to solve and one can observe that formulation (R) clearly outperforms (S). Formulation (S) already fails to solve up to optimality some

Table 2
Computational results on the random instances with Prepr = Yes, Improv = Yes and InitSol = Yes.

		Gap						CPU Time						
		UBGap		MIPGap		RootGap		Htime	ETime		TotalTime		RootTime	
n	p	(S)	(R)	(S)	(R)	(S)	(R)		(S)	(R)	(S)	(R)	(S)	(R)
100	5	4.10	4.10	0.00	0.00	41.82	0.00	22.95	130.12	15.60	153.07	38.55	76.11	12.25
	10	50.58	50.58	0.00	0.00	26.20	1.52	11.57	202.13	104.14	213.70	115.71	166.47	103.18
	20	82.82	82.82	0.00	0.00	27.63	1.55	23.73	225.05	171.69	248.78	195.42	214.33	170.36
125	5	6.26	6.26	0.00	0.00	76.34	2.94	59.83	460.36	26.86	520.19	86.69	428.95	26.47
	10	39.83	39.83	0.00	0.00	30.31	3.93	24.30	573.29	306.23	597.58	330.53	545.08	301.10
	20	94.49	94.49	0.00	0.00	45.58	2.51	69.31	665.41	511.36	734.72	580.66	595.99	503.85
150	5	11.33	11.33	0.00	0.00	63.42	3.88	120.43	1168.44	202.35	1288.87	322.78	1084.07	193.60
	10	52.00	52.00	0.00	0.00	45.17	5.62	46.23	1291.57	809.96	1337.80	856.19	1269.09	793.69
	20	74.38	74.38	0.00	0.00	61.46	3.16	116.57	1333.08	1263.82	1449.65	1380.39	1277.48	1246.64
175	5	4.61	7.23	16.14	0.00	54.32	2.02	177.53	2299.54	104.13	2477.07	281.66	1723.23	74.56
	10	44.33	13.91	0.00	51.41	31.64	0.96	86.15	2601.35	2893.36	2687.49	2979.51	2451.79	731.75
	20	95.65	0.00	0.00	89.54	15.64	0.00	184.58	2586.86	3602.70	2771.44	3787.28	2463.41	0.60
200	5	**	7.85	**	0.00	**	1.21	267.20	**	298.06	**	565.26	**	239.84
	10	**	**	**	**	**	**	91.58	**	**	**	**	**	**
	20	**	**	**	**	**	**	276.34	**	**	**	**	**	**

Table 3
Computational results on the Euclidean instances with Prepr = Yes, Improv = Yes and InitSol = Yes.

		Gap						CPU Time						
		UBGap		MIPGap		RootGap		Htime	ETime		TotalTime		RootTime	
n	p	(S)	(R)	(S)	(R)	(S)	(R)		(S)	(R)	(S)	(R)	(S)	(R)
100	5	19.62	19.62	0.00	0.00	45.27	2.69	101.92	342.69	55.41	444.61	157.32	342.06	55.06
	10	42.98	42.98	0.00	0.00	45.84	3.24	285.58	409.98	149.75	695.57	435.34	341.12	146.88
	20	50.14	50.14	0.00	0.00	45.52	2.77	552.15	436.28	177.51	988.43	729.66	306.24	174.98
125	5	25.29	25.29	0.00	0.00	22.51	3.39	190.95	1042.27	162.08	1233.21	353.03	1027.96	154.88
	10	52.80	52.80	0.00	0.00	59.14	2.22	395.28	1244.62	391.47	1639.90	786.76	1046.51	386.70
	20	57.06	57.06	0.00	0.00	44.90	2.34	1465.82	1317.31	462.77	2783.13	1928.59	1096.71	453.67
150	5	45.20	45.20	16.00	0.00	36.12	2.31	522.43	2540.69	861.88	3063.12	1384.31	2073.03	844.39
	10	56.10	57.42	16.15	0.00	61.27	2.08	1369.36	2878.73	1194.58	4248.09	2563.95	2435.76	1180.91
	20	64.37	64.37	0.00	0.00	60.07	0.54	5744.96	2353.36	1383.44	8098.31	7128.40	2013.68	1344.12
175	5	**	14.14	**	34.94	**	1.75	633.90	**	2100.75	**	2734.65	**	656.36
	10	**	19.21	**	52.10	**	0.53	1886.49	**	2910.53	**	4797.02	**	739.10
	20	**	26.12	**	34.84	**	0.50	5035.77	**	2620.58	**	7656.34	**	1156.86
200	5	**	**	**	**	**	**	1096.10	**	**	**	**	**	**
	10	**	**	**	**	**	**	2720.41	**	**	**	**	**	**
	20	**	**	**	**	**	**	9687.48	**	**	**	**	**	**

Table 4
Computational results on the Euclidean instances with Prepr = No, Improv = Yes and InitSol = No.

		Gap						CPU Time			
		UBGap		MIPGap		RootGap		ETime		RootTime	
n	p	(S)	(R)	(S)	(R)	(S)	(R)	(S)	(R)	(S)	(R)
100	5	19.62	19.62	0.00	0.00	100.00	3.22	473.18	234.00	411.70	222.03
	10	42.98	42.98	0.00	0.00	100.00	0.50	431.41	229.68	355.44	225.12
	20	50.14	50.14	0.00	0.00	80.00	1.51	425.64	240.08	329.38	235.61
125	5	25.29	25.29	0.00	0.00	82.67	5.18	1778.02	583.56	1669.41	569.42
	10	52.80	52.80	0.00	0.00	56.41	2.46	1583.90	518.24	1219.16	507.42
	20	57.06	57.06	0.00	0.00	80.00	2.46	1705.12	513.37	1144.81	502.52
150	5	45.20	-0.65	20.00	60.00	47.95	0.52	3011.20	3048.64	2569.10	880.98
	10	57.42	30.28	40.00	40.00	60.00	0.54	2793.46	2908.53	2106.14	1451.82
	20	64.37	16.27	60.00	60.00	80.00	0.27	2966.03	3033.80	1790.85	865.90
175	5	**	**	**	**	**	**	**	**	**	**
	10	**	**	**	**	**	**	**	**	**	**
	20	**	**	**	**	**	**	**	**	**	**
200	5	**	**	**	**	**	**	**	**	**	**
	10	**	**	**	**	**	**	**	**	**	**
	20	**	**	**	**	**	**	**	**	**	**

instances for $n = 150$. For larger sizes, (S) does not solve the instances and (R) is able to provide solutions, at least for $n = 175$, although not certifying optimality.

Table 4 reports the same comparison with a different configuration of parameters. One can observe that its performance is worse than the one obtained with the previous configuration. Even the instances with $n = 150$ could not be solved up to optimality now.

From our experiments one can conclude that the radius formulation (R) outperforms the other ones and that using it with the preprocessing and the initial solution provided by the heuristic procedure proposed in Section 6 together with the improve-

Table 5
Number of centers actually opened.

	random instances						Euclidean instances					
	n						n					
p	25	50	75	100	125	150	25	50	75	100	125	150
5	3.6	4.6	4.8	5	4.8	4.8	4.8	5	5	5	5	5
10	3.6	4.8	5.2	5	5.2	5.2	5.4	6	5.4	7	6.6	5.5
20	-	4.8	5.2	5	5.2	5.2	-	6	5.4	7	6.6	5.5

Table 6
Number of centers actually opened assuming that up to two of service facilities can be placed at the same location.

		random instances						Euclidean instances					
		n						n					
	p	25	50	75	100	125	150	25	50	75	100	125	150
	5	4.4	4	4.2	5	4.8	5	4.2	4.4	4.2	4.6	5	4.6
T	10	5.6	6.4	8.4	7.6	7.2	7	6.8	8.2	7.8	8.8	8.2	6.4
	20	-	6.4	8.4	7.6	7.2	7	-	8.2	7.8	8.8	8.2	6.4
	5	2.4	2.2	2.2	4	3.6	4	2.4	2.6	2.2	3.2	4	3.4
D	10	3	3.4	4.2	3.8	3.6	4	3.4	4.4	4	4.4	4.2	3.2
	20	-	3.4	4.2	3.8	3.6	4	-	4.4	4	4.4	4.2	3.2

ments obtained incorporating constraints (19) and (20), instances with up to $n = 175$ customers can be solved. The maximum total time spent on solving the heuristic plus the radius formulation is one hour of CPU time for the random instances and around two hours for larger Euclidean instances.

8.3. An alternative way of distributing centers

In Table 5 we show the average number of centers that are actually open for each combination of parameters (n, p) where optimality is proven considering the random and the Euclidean instances. As mentioned in Example 2.2 we can see that in this problem, contrary to what is usual in location problems, opening more centers does not imply smaller objective values and that the number of opened centers does not reach the maximum number p in most of the instances for $p = 10$ and $p = 20$. Indeed, for the instances considered in this work, it seems that it suffices considering values up to $p = 10$.

This observation led us to consider an alternative distribution of centers that may locate up to two centers per point. In this way one can model the possibility of installing more available centers (up to p), giving service with the second center located at the same place that the first, unavailable, center. Examples of this policy can be found, for instance, when installing cash machines in the bank offices, bikes in bike-sharing dock stations and many other cases. To cope with this situation we double the size of candidate sites for locating centers and we assume that the favorite site of a customer when he is in the first, preferred open center, is a center located at the same point. Then, we slightly modify the formulations as follows.

Let $A = \{1, \ldots, n\}$ be the set of sites for users and let $A' = \{1, \ldots, n, n+1, \ldots, 2n\}$ be the set of candidate sites for locating centers, being site $n + j$ the replica of site j. In the different formulations we distinguish between these two index sets and we update variables and parameters accordingly,

- we consider $A'_i = A_i \bigcup \{n + j : j \in A_i\} \subseteq A'$ as the subset of sites that user $i \in A$ is willing to accept as service centers;
- we suppose variables $x_{i\ell' j'}$ defined for $i \in A$, $(\ell', j') \in I'_i$ where, $I'_i = \{(\ell', j') : \ell' \in A'_i, j' \in A'_i, j' \neq \ell', \ell' \leq^i_i j'\}$ for all $i \in A$;
- variables y_i are defined for $i \in A'$;
- distances are duplicated in the natural way, that is, $d_{i\ell' j'} = d_{i\ell j}$ for $i \in A$, $\ell \in A$, $j \in A$ and $\ell' \in A'$, $j' \in A'$, such that $\ell' \in \{\ell, n + \ell\}$, $j' \in \{j, n + j\}$;
- preferences verify that (i) the level of preference of i for j' when i is at ℓ' is the same as the level of preference of i for j when i is at ℓ for $j' \in \{j, n + j\}$ and $\ell' \in \{\ell, n + \ell\}$, (ii) $j'_1 \in \{j, n + j\}$ is the favorite site of i when i is at $j'_2 \in \{n + j, j\}$.

Using these modifications, we obtain the results given in Tables 6 and 7. In Table 6 block named "T" stands for the total number of actual open centers and the block named "D" represents the number of different locations for the open centers, that is to say, two centers sited at the same point count 2 in block "T" but 1 in block "D".

Note that, for $p = 10, 20$, each average of block "D" is approximately equal to half of the corresponding average of block "T", that is to say, in almost all the cases either a site is not used or it is used to install two centers. This is natural, since once the user has visited a center, his favorite one and the one with the minimum distance (so the best one for the objective function) is the center placed at the same site. As a result, the number of total open centers may increase, with respect to the number of open centers of Table 5, to reach an even number. Although a similar behavior can be observed in the case $p = 5$, since here there is an odd number of available centers, not always the twin centers can be located. The aforementioned phenomenon of equal optimal solutions for the cases $p = 10$ and $p = 20$ is also present here. More open centers bring more options to the users that carry larger distances in the objective function. The trade-off between freedom for the users and total distance for the decision maker fixes the number of open centers in a value that will not change when p increases.

Another fact that merits attention is that in the case $p = 5$ there is a total number of open centers (Table 6, block "T") less than the number of open centers when only one can be installed (Table 5). Again, this is probably due to the fact that 5 is an odd number. Sometimes it is better for the objective value to install $2 + 2$ centers in two sites than locating $2 + 2 + 1$ in three sites.

These phenomena can be observed in Fig. 10 where the solution obtained when solving one of the generated instances for $n = 25$ and $p = 5$ is depicted. In this figure, blue points stand for the user and potential center sites, red points represent open center sites, green point is the user that gives us the value of the objective function and lines symbolize the path followed by this user to the first most preferred open center (black line) and from this one to

Table 7
Percentage of improvement on the objective function value when two different centers can be placed at the same location.

	Random instances						Euclidean instances					
	n						n					
p	25	50	75	100	125	150	25	50	75	100	125	150
5	25.10	15.51	11.78	8.18	4.60	6.50	22.80	15.60	7.01	7.13	2.03	5.82
10	25.10	19.61	23.33	15.45	15.68	15.97	29.62	18.86	15.06	17.66	12.63	14.22
20	-	19.61	23.33	15.45	15.68	16.35	-	18.86	15.06	17.66	12.63	14.22

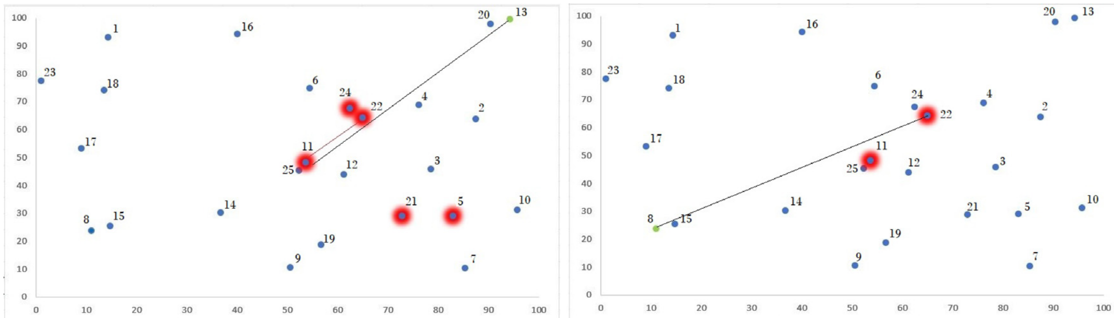

Fig. 10. Solution for an instance with $n = 25$ and $p = 5$ (or $p = 10$) when locating two different centers in the same site is not allowed (left) and when it is allowed (right) .

the most preferred open center for the user when he is in the first center (red line). In the left figure locating two different centers in the same site is not allowed. In this case, the optimal solution is obtained when five centers are open (centers sited at location 5, 11, 21, 22 and 24). The maximum sum of the distance from any user to any of its preferred open centers plus the distance from this center to any of the open centers the user prefers once he is there is obtained when user sited at 13 goes to center sited at 11 (the most preferred center of the five open centers) and, in case of failure, he goes from this position to center sited at 22 (the most preferred center of the remaining four when he is at position 11) and its value is 85 distance units (d.u.).

In the right figure we assume that two centers can be located at the same position. In this last case the optimal number of open centers is a total of four (two centers sited at location 11 and two centers at location 22). The maximum sum of the distance from any user to any of its preferred open centers plus the distance from this center to any of the open centers the user prefers once he is there is obtained when user sited at 8 goes to center sited at 22 (one of the most preferred center of the four open centers) and, in case of failure, he uses the second center located at the same position and its value is 67 d.u. Here, we can observe that if we open a fifth center in any other position the value of the objective function increases. For instance, suppose that a center sited in position 24 is open. In this case, the value of the objective function would be 71, since user sited at 8 has the same preference for center at position 24 than for centers sited at position 22 and then, he can go first to the only center sited at 24 (at a distance of 67 d.u.) and, in case of failure, he goes from position 24 to center sited a 22 (at a distance of 4 d.u. from position 24).

In both cases (only one or two centers can be opened at the same location), the optimal solution obtained for $p = 10$ is the same as for $p = 5$. When only one center can be opened at the same location opening more than five centers is not optimal because the preferences of the users. For instance, suppose that we also open center sited at 3 then, the value of the objective function would be 89. This value is reached because user 8 first goes to center sited at 5 (at a distance of 72 d.u.) which is the most preferred open center for user 8 and, in case of failure, he goes to center sited at 3 (at a distance of 17 d.u. from position 5) which is the most preferred open center for user 8 when he is at location 5. A similar situation happens when any other center or group of centers are added to the list of open centers. In the case that two centers are allowed to open in the same location, the optimal solution obtained for $p = 5$ is also optimal for $p = 10$ but, in this case, there is another alternative optimal solution opening a total of 10 centers (two centers sited at locations 11, two at location 22, two at site 24, two at site 12 and two centers located at site 6).

Finally, Table 7 shows the average improvements (in percentage) on the objective function value of the alternate way of distributing the centers with respect to the objective function value of the original one. We can observe that allowing two different centers to be located in the same site gives rise to a benefit that, in the solved instances, ranges from 2% for $n = 125$ and $p = 5$ to close to 30% for $n = 25$ and $p = 10$.

9. Conclusions

This paper introduces the Dynamically Second-preferred p-center Problem. This is a new problem that extends the p-next center problem in Albareda-Sambola et al. (2015a) considering a worse case situation, customers' preferences and subsets of accepted service centers for each customer. The DSpP aims at choosing at most p centers so that each demand point can visit at least two acceptable centers and the maximum sum of distances from any demand point to any of its preferred centers plus the distance from any of the preferred centers to any of the centers the user prefers once he is there is minimized. This new model responds to those situations where the decision of placing the service centers and trips are supported by a planner but once these centers are located the trips to visit the centers are chosen by the customers. Using this model the planner assumes a risk-averse attitude and tries to hedge against misbehavior of customers that may choose the longest trips to maximize their revenue.

We have presented three different formulations based on different spaces of variables: (i) Three-indexed formulation, (ii) Straight formulation, and (iii) Radius formulation. Each one exhibits interesting features that make it worth to be considered and evaluated. From our experiments we conclude that the Radius formulation (R) outperforms the other two formulations allowing to solve to optimality instances with up to $n = 175$ customers. Going beyond this size requires further analysis looking for strengthening of the current formulations or new ones in other spaces of variables, and perhaps, the use of better heuristic algorithms that provide good solutions in short CPU times. Both subjects, although very interesting, are beyond the scope of this paper and may be the topic of a follow up paper.

Acknowledgments

This research has been partially supported by the Agencia Estatal de Investigación (AEI) and the European Regional Development's funds (ERDF): PID2020-114594GB-C21; Regional Government of Andalusia: projects CEI-3-FQM331, FEDER-US-1256951, and P18-FR-1422; Fundación BBVA: project NetmeetData (Ayudas Fundación BBVA a equipos de investigación científica 2019). Alfredo Marín has been supported by Spanish Ministry of Science and Innovation under project PID2019-110886RB-I00. Part of this research was conducted while he was on sabbatical at Universidad de Sevilla, Spain.

References

Albareda-Sambola, A., Díaz, J. A., & Fernández, E. (2010). Lagrangean duals and exact solution to the capacitated p-center problem. *European Journal of Operational Research, 201*, 71–81.

Albareda-Sambola, M., Hinojosa, Y., Marín, A., & Puerto, J. (2015a). When centers can fail: A close second opportunity. *Computers & Operations Research, 62*, 145–156.

Albareda-Sambola, M., Hinojosa, Y., & Puerto, J. (2015b). The reliable p-median problem with at-facility service. *European Journal of Operational Research, 245*, 656–666.

Berman, O., Krass, D., & Menezes, M. (2013). Location and reliability problems on a line: Impact of objectives and correlated failures on optimal location patterns. *Omega, 41*, 766–779.

Brimberg, J., Maier, A., & Schöbel, A. (2021). When closest is not always the best: The distributed p-median problem. *Journal of the Operational Research Society, 72*(1), 200–216.

Calvete, H., Domínguez, C., Galé, C., Labbé, M., & Marín, A. (2019). The rank pricing problem: Models and branch-and-cut algorithms. *Computers & Operations Research, 105*, 12–31.

Cánovas, L., García, S., Labbé, M., & Marín, A. (2007). A strengthened formulation for the simple plant location problem with order. *Operations Research Letters, 35*, 141–150.

Casas-Ramírez, M., & Camacho-Vallejo, J. F. (2017). Solving the p-median bilevel problem with order through a hybrid heuristic. *Applied Soft Computing, 60*, 73–86.

Casas-Ramírez, M. S., Camacho-Vallejo, J. F., & Martínez-Salazar, I. A. (2018). Approximating solutions to a bilevel capacitated facility location problem with customer's patronization toward a list of preferences. *Applied Mathematics and Computation, 319*, 369–386.

Cornejo Acosta, J. A., García Díaz, J., Menchaca-Méndez, R., & Menchaca-Méndez, R. (2020). Solving the capacitated vertex k-center problem through the minimum capacitated dominating set problem. *Mathematics, 8*(9), 1551.

Cornuéjols, G., Nemhauser, G. L., & Wolsey, L. A. (1980). A canonical representation of simple plant location problems and its applications. *SIAM Journal on Algebraic Discrete Methods, 1*, 261–272.

Das, T., Sridharan, V., & Gurusamy, M. (2019). A survey on controller placement in SDN. *IEEE Communications Surveys & Tutorials, 22*, 472–503.

Daskin, M. S. (2000). A new approach to solving the vertex p-center problem to optimality: Algorithm and computational results. *Communications of the Operations Research Society of Japan, 45*, 428–436.

Díaz, J. A., Luna, D. E., Camacho-Vallejo, J. F., & Casas-Ramírez, M. S. (2017). GRASP and hybrid GRASP-tabu heuristics to solve a maximal covering location problem with customer preference ordering. *Expert Systems with Applications, 82*, 67–76.

Domínguez, C., Labbé, M., & Marín, A. (2021). The rank pricing problem with ties. *European Journal of Operational Research, 294*, 492–506.

Domínguez, C., Labbé, M., & Marín, A. (2022). Mixed-integer formulations for the capacitated rank pricing problem with envy. *Computers & Operations Research, 140*, 105664.

Drezner, Z. (1987). Heuristic solution methods for two location problems with unreliable facilities. *Journal of the Operational Research Society, 38*, 509–514.

Elloumi, S., Labbé, M., & Pochet, Y. (2004). A new formulation and resolution method for the p-center problem. *INFORMS Journal on Computing, 16*, 84–94.

Espejo, I., Marín, A., Puerto, P., & Rodríguez-Chía, A. M. (2009). A comparison of formulations and solution methods for the minimum-envy location problem. *Computers & Operations Research, 36*(6), 1966–1981.

García, S., Labbé, M., & Marín, A. (2011). Solving large p-median problems with a radius formulation. *INFORMS Journal on Computing, 22*, 546–556.

Hanjoul, P., & Peeters, D. (1987). A facility location problem with clients' preference orderings. *Regional Science and Urban Economics, 17*, 451–473.

Hansen, P., Kochetov, Y., & Mladenović, N. (2004). Lower bounds for the uncapacitated facility location problem with user preferences. In *Les Cahiers du GERAD g-2004-24*. HEC Montreal.

Herrera, R., Kalcsics, J., & Nickel, S. (2007). Reliability models for the uncapacitated facility location problem with user preferences. In *Operations research proceedings* (pp. 135–140). Springer, Berlin, Heidelberg.

Jaeger, M., & Goldberg, J. (1994). A polynomial algorithm for the equal capacity p-center problem on trees. *Transportation Science, 28*, 167–175.

Kariv, O., & Hakimi, S. L. (1979). An algorithmic approach to network location problems Part I: The p-centers. *SIAM Journal of Applied Mathematics, 37*, 513–538.

Khuller, S., & Sussmann, Y. J. (2000). The capacitated k-center problem. *SIAM Journal on Discrete Mathematics, 13*, 403–418.

Killi, B. P. R. (2018). *On placement of controllers and hypervisors in software defined networks*. Indian Institute of Technology Guwahati Doctoral dissertation.

Killi, B. P. R., & Rao, S. V. (2016). Controller placement with planning for failures in software defined networks. In *2016 IEEE international conference on advanced networks and telecommunications systems (ANTS)* (pp. 1–6). IEEE.

Killi, B. P. R., & Rao, S. V. (2017). Capacitated next controller placement in software defined networks. *IEEE Transactions on Network and Service Management, 14*, 514–527.

Kramer, R., Iori, M., & Vidal, T. (2020). Mathematical models and search algorithms for the capacitated p-center problem. *INFORMS Journal on Computing, 32*, 444–460.

Landete, M., & Marín, A. (2009). New facets for the two-stage uncapacitated facility location polytope. *Computational Optimization and Applications, 44*, 487–519.

(2019). In G. Laporte, S. Nickel, & F. Saldanha da Gama (Eds.), *Location science*. Springer. ISBN 978-3-030-32176-5

López-Sánchez, A. D., Sánchez-Oro, J., & Hernández-Díaz, A. G. (2019). GRASP and VNS for solving the p-next center problem. *Computers & Operations Research, 104*, 295–303.

Marić, M., Stanimirović, Z., & Milenković, N. (2012). Metaheuristic methods for solving the bilevel uncapacitated facility location problem with clients' preferences. *Electronic Notes in Discrete Mathematics, 39*, 43–50.

Marić, M., Stanimirović, Z., Milenković, N., & Djenić, A. (2015). Metaheuristic approaches to solving large-scale bilevel uncapacitated facility location problem with clients' preferences. *Yugoslav Journal of Operations Research, 25*, 361–378.

Marín, A., Nickel, S., Puerto, J., & Velten, S. (2009). A flexible model and efficient solution strategies for discrete location problems. *Discrete Applied Mathematics, 157*(5), 1128–1145.

Marín, A., Nickel, S., & Velten, S. (2010). An extended covering model for flexible discrete and equity location problems. *Mathematical Methods of Operations Research, 71*(1), 125–163.

Mladenovic, N., Labbé, M., & Hansen, P. (2003). Solving the p-center problem with tabu search and variable neighborhood search. *Networks, 42*, 48–64.

Özsoy, F. A., & Pinar, M. c. (2006). An exact algorithm for the capacitated vertex p-center problem. *Computers & Operations Research, 33*, 1420–1436.

Quevedo-Orozco, D. R., & Ríos-Mercado, R. Z. (2015). Improving the quality of heuristic solutions for the capacitated vertex p-center problem through iterated greedy local search with variable neighborhood descent. *Computers & Operations Research, 62*, 133–144.

Scaparra, M. P., Pallotino, S., & Scutellà, M. G. (2004). Large-scale local search heuristics for the capacitated vertex p-center problem. *Networks, 32*, 241–255.

Vasil'ev, I. L., Klimentova, K. B., & Kochetov, Y. A. (2009). New lower bounds for the facility location problem with clients' preferences. *Computational Mathematics and Mathematical Physics, 49*, 1010–1020.

Vasilyev, I., Klimentova, X., & Boccia, M. (2013). Polyhedral study of simple plant location problem with order. *Operations Research Letters, 41*, 153–158.