The Hampered K-Median Problem with Neighbourhoods

Justo Puerto[a,*], Carlos Valverde[b,*]

[a] *Institute of Mathematics (IMUS) and Department of Statistics and Operations Research, University of Seville, Seville, 41012, Spain*

[b] *Institute of Mathematics (IMUS) and Department of Statistics and Operations Research, University of Seville, Seville, 41012, Spain*

Abstract

This paper deals with facility location problems in a continuous space with neighbours and barriers. Each one of these two elements, neighbours and barriers, make the problems harder than their standard counterparts. Therefore, mixing both together results in a new challenging problem that, as far as we know, has not been addressed before but that has applications for inspection and surveillance activities and the delivery industry assuming uniformly distributed demand in some regions. Specifically, we analyze the K-Median problem with neighbours and polygonal barriers under two different situations. As a first building block, we deal with the problem assuming that neighbourhoods are not visible from one another and therefore there are no rectilinear paths joining any two of them without crossing barriers. Under this hypothesis we derive a valid mixed integer, linear formulation. Removing that hypothesis leads to the more general, realistic problem but at the price of making it more challenging. Adapting the elements of the first formulation, we also develop another valid mixed integer, bilinear formulation. Both formulations handle polygonal barriers and neighbours that are second-order cone (SOC) representable, that we preprocess and strengthen with valid inequalities. These mathematical programming formulations are also instrumental to derive an adapted matheuristic algorithm that provides good quality solutions for both problems in short computing time. The paper also reports an extensive computational experience showing that our exact and heuristic approaches are useful: the exact approach can solve to optimality instances with up to 50 neighbourhoods and different number of barriers within one hour of CPU time, whereas the matheuristic always returns good feasible solutions in less than 100 seconds.

Keywords: Facility location, Continuous Location, Barriers, Mixed integer Conic programming

1. Introduction

Location analysis is a classical branch of operations research that studies the best way to place some facilities to satisfy the demand of customers. In location analysis, problems are usually classified in discrete or continuous facility location problems. The first class is considered when there is a finite number of candidates to allocate facilities (see Ulukan and Demircioğlu (2015) for a survey). Continuous facility location problems arise if facilities can be placed anywhere in some continuous regions. Both versions are widely investigated in the literature (see Drezner and Hamacher (2004) or Nickel and Puerto

*Equally contributing authors

Email addresses: puerto@us.es (Justo Puerto), cvalverde@us.es (Carlos Valverde)

(2007) for more details) by their many applications in transportation, logistics or telecommunication. For these problems, lot of variants have been studied in terms of the objective functions to be optimized, the number of facilities that must be allocated or the maximum capacity that facilities can supply, among many other respects (we refer the reader to Kuehn and Hamburger (1963) and Puerto (2008)).

In Blanco (2019), the Ordered k-Median Problem with Neighbourhoods is presented as a single source uncapacitated continuous facility location problem that extends its respective underlying discrete location problem. In this problem, facilities are allowed to be allocated in certain regions called neighbourhoods. If those are points, the problem reduces to the single source uncapacitated facility discrete location problem, that have been already studied in the literature. Otherwise, the continuous version is considered. In this version, different shapes and sizes for the neighbourhoods allow one to model how imprecise the provided locational information is. This problem also has interest on drone delivery and inspection problems. Neighbourhoods can represent regions that the drone must reach and where the customers are willing to pick up the orders (they can be seen as uniform probability densities) in the delivery industry. Moreover, they can be also used for modelling some areas that must be inspected by the drone (whenever visiting a point of these areas is enough to consider them as inspected). This framework will be called Facility Location with Neighbourhoods, a terminology borrowed from the neighbourhood versions of the Minimum Spanning Tree problem, described in Blanco et al. (2017) and the Traveling Salesman problem, studied in Gentilini et al. (2013), Yuan and Zhang (2017) or Puerto and Valverde (2022).

This paper extends the k-Median Problem with Neighbourhoods by including a set of barriers, represented by line segments, that the trips between demand points and service facilities cannot cross. These barriers simulate buildings in urban areas that vehicles (drones) cannot cross. The resulting problem keeps geometric components from the p-median problem with neighbourhoods that must be exploited to partially overcome the difficulties of the solution approaches and algorithms in the network design among neighbourhoods with barriers. The use of barriers in location problems has been studied (see Klamroth (2002)). However, the combination of both elements has attracted less attention in the Operations Research literature.

Our goal in this paper is to deal with the k-median problem with neighbourhoods and barriers that we call the Hampered k-Median problem with Neighbourhoods (H-KMPN). We present exact mathematical programming formulations assuming linear barriers and second-order cone (SOC) representable neighbourhoods. These formulations are modeled by using a geodesic shortest-path representation, based on problems studied in Mitchell (2017). These assumptions lead to quadratically-constrained mixed-integer formulations. Solving this family of formulations in addition to the classic k-median, that is NP-hard, makes the solution of the problem under study a hard challenge. On-the-shelf solvers can deal only with small-size instances. This fact motivates the design of a matheuristic that provides good quality solutions for medium-size instances.

The paper is organized in six sections. In Section 2 the problem and its variant are introduced and described. Section 3 is devoted to provide quadratically-constrained mixed-integer programming formulations of the problems. In Section 4 the matheuristic approach is described. The results of some computational experiments are reported in Section 5. Finally, some conclusions are presented in Section

6.

2. Description of the Problem

In this section, the framework of the two versions of the problem considered in the manuscript are analyzed: the Hampered k-Median Problem with Hidden Neighbourhoods H-KMPHN and the Hampered k-Median Problem with Neighbourhoods H-KMPN. Since we have in mind their applications to the drone delivery problem with uniformly distributed demand in regions and inspection problems, at times, we will refer to the moving object as the *drone*.

First of all, we state the sets that describe the main elements of the problems. Second, we set the assumptions that barriers must verify. Finally, the goal and the sets of parameters used in the following are defined in order to give valid formulations for these problems.

2.1. Parameters and Assumptions of the Problem

The sets describing both versions of the problem are:

- $\mathcal{S}$: Set of neighbourhoods describing the possible sources where a facility can be allocated. It is assumed, wlog, that one facility can be allocated to each source at most once.
- $\mathcal{T}$: Set of neighbourhoods representing the targets that must be served by a facility. It is assumed, wlog, that each target is served when it has been assigned to a facility.
- $\mathcal{B}$: Set of barriers (line segments) that can not be crossed when a facility is joined with a target. The assumptions made for this set of line segments are the following:

  A1 The line segments of $\mathcal{B}$ are located in general position, i.e., the endpoints of these segments are not aligned. Although it is possible to model the most general case, one can always slightly modify one of the endpoints so that the segments are in general position.

  A2 The line segments of $\mathcal{B}$ are open sets, that is, it is possible that the drone visits endpoints of segments, but entering in its interior is not allowed. Observe that without loss of generality, we can always slightly enlarge these segments to make them open.

  A3 If there are two overlapping barriers, we assume that there is only one barrier given by the union of them.

  A4 There is no rectilinear path joining a pair of source-target neighbourhoods without crossing an obstacle.

The H-KMPN is the relaxed version of the H-KMPHN without imposing assumption **A4**. In this case, it is not required that the barriers separate neighbourhoods completely, i.e., when moving from one neighbourhood to another one it is possible to go following a straight line without crossing any barrier. Figure 1 shows an example of each version of the problem that is being considered. The left picture shows an instance of the H-KMPHN, where green neighbourhoods represent possible sources to allocate the facilities, blue neighbourhoods represent targets to be assigned to the sources and the red

line segments show the barriers that the drone cannot cross. The right picture illustrates an instance of the H-KMPN where some sources and targets can be joined by a rectilinear path.

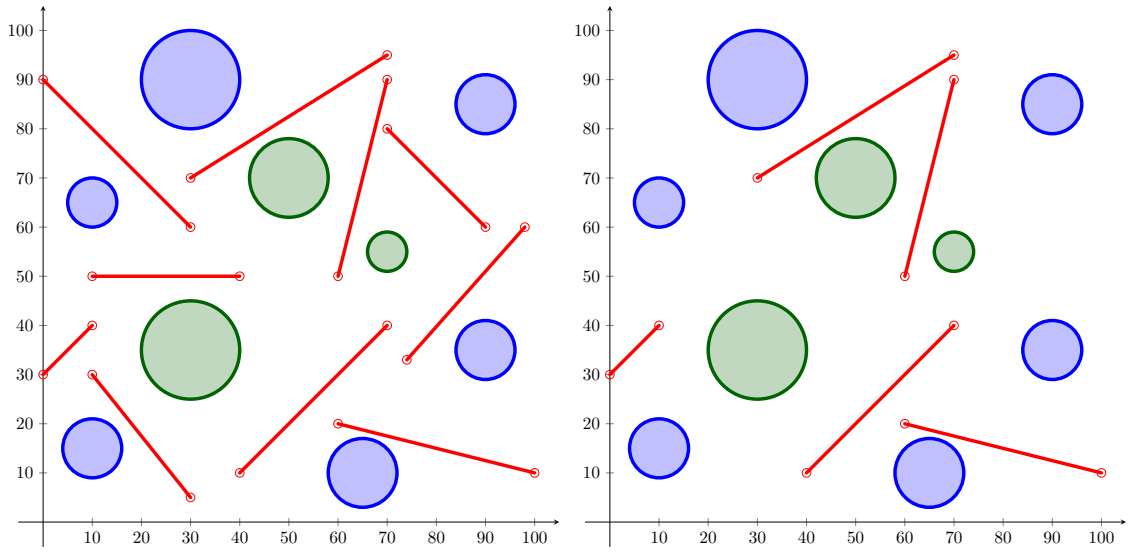

Figure 1: Problem data of the H-KMPHN and H-KMPN

2.2. Description of the Hampered k-Median Problem with Neighbourhoods

The goal of the H-KMPHN is to find a subset of k points in the source set $\mathcal{S}$, at most one in each neighbourhood, and one point in each target set $\mathcal{T}$ that minimize the weighted length of the path joining each target point with its associated source point and the weighted link distance without crossing any barrier of $\mathcal{B}$ assuming **A1-A4**. Recall that the link distance accounts for the numbers of edges of the path joining two points in the underlying graph. The interested reader is referred to de Berg et al. (1990); Daescu et al. (2008) for further details. To state the model, we define the following sets:

- $V_{\mathcal{S}} = \{P_S : S \in \mathcal{S}\}$. Set of the points selected in the sources of $\mathcal{S}$.
- $V_{\mathcal{B}} = \{P_B^1, P_B^2 : B = \overline{P_B^1 P_B^2} \in \mathcal{B}\}$. Set of vertices that come from the endpoints of barriers in the problem.
- $V_{\mathcal{T}} = \{P_T : T \in \mathcal{T}\}$. Set of the points selected in the targets of $\mathcal{T}$.
- $E_{\mathcal{S}} = \{(P_S, P_B^i) : P_S \in V_{\mathcal{S}}, P_B^i \in V_{\mathcal{B}} \text{ and } \overline{P_S P_B^i} \cap B'' = \emptyset, \forall B'' \in \mathcal{B},\ i = 1, 2\}$. Set of edges formed by the line segments that join the point selected in any source neighbourhood $S \in \mathcal{S}$ and every endpoint in the barriers that do not cross any other barrier in $\mathcal{B}$.
- $E_{\mathcal{B}} = \{(P_B^i, P_{B'}^j) : P_B^i, P_{B'}^j \in V_{\mathcal{B}} \text{ and } \overline{P_B^i P_{B'}^j} \cap B'' = \emptyset,\ \forall B'' \in \mathcal{B},\ i, j = 1, 2\}$. Set of edges formed by the line segments that join two vertices of $V_{\mathcal{B}}$ and do not cross any other barrier in $\mathcal{B}$.
- $E_{\mathcal{T}} = \{(P_B^i, P_T) : P_B^i \in V_{\mathcal{B}}, P_T \in V_{\mathcal{T}} \text{ and } \overline{P_B^i P_T} \cap B'' = \emptyset, \forall B'' \in \mathcal{B},\ i = 1, 2\}$. Set of edges formed by the line segments that join the point selected in any target neighbourhood $T \in \mathcal{T}$ and every endpoint in the barriers that do not cross any other barrier in $\mathcal{B}$.

The above sets allow us to define the graph $G_{\text{KMPHN}} = (V_{\text{KMPHN}}, E_{\text{KMPHN}})$ induced by the barriers and neighbourhoods, where $V_{\text{KMPHN}} = V_{\mathcal{S}} \cup V_{\mathcal{B}} \cup V_{\mathcal{T}}$ and $E_{\text{KMPHN}} = E_{\mathcal{S}} \cup E_{\mathcal{B}} \cup E_{\mathcal{T}}$.

By taking the same approach, the graph induced for the relaxed version H-KMPN can be described as $G_{\text{KMPN}} = (V_{\text{KMPN}}, E_{\text{KMPN}})$, $V_{\text{KMPN}} = V_{\text{KMPHN}}$ and $E_{\text{KMPN}} = E_{\text{KMPHN}} \cup E_{\mathcal{ST}}$, where:

- $E_{\mathcal{ST}} = \{(P_S, P_T) : P_S \in V_{\mathcal{S}}, P_T \in V_{\mathcal{T}} \text{ and } \overline{P_S P_T} \cap B'' = \emptyset, \forall B'' \in \mathcal{B}, i = 1, 2\}$. Set of edges formed by the line segments that join the point selected in any source neighbourhood $S \in \mathcal{S}$ and the point selected in any target neighbourhood $T \in \mathcal{T}$.

The difference between the set of edges in the H-KMPHN with respect to the graph in H-KMPN is that, in the former case, the edges that join each pair of neighbourhoods must be considered.

3. MINLP Formulations

This section proposes a mixed-integer non-linear programming formulation (MINLP) for the problem described in Section 2. First of all, constraints that describe the k-median are set. Then, two approaches to ensure connectivity between each pair source-target are explained. Later, the conic programming representation of the neighbourhoods and distance are presented. Finally, the constraints that check if a segment is included in the set of edges E_X with $X \in \{\text{KMPHN}, \text{KMPN}\}$ are described.

First of all, we introduce the decision variables that represent the problem. They are summarized in Table 1.

3.1. k-median constraints

In the classical k-median, there are two decisions to make, that is, which sources are selected to allocate a facility and which targets are assigned to be served by each facility. To deal with these decisions, it is necessary to define the binary variables:

- $y(S)$, that assumes value one if the source neighbourhood $S \in \mathcal{S}$ is selected.
- $x(ST)$, that is one if the target neighbourhood $T \in \mathcal{T}$ is assigned to the selected source $S \in \mathcal{S}$.

The block of constraints inherited from the k-median are the following:

$$\sum_{S \in \mathcal{S}} y(S) = k, \tag{k-median-C1}$$

$$x(ST) \leq y(S), \quad \forall S \in \mathcal{S}, \quad \forall T \in \mathcal{T}, \tag{k-median-C2}$$

$$\sum_{S \in \mathcal{S}} x(ST) = 1, \quad \forall T \in \mathcal{T}. \tag{k-median-C3}$$

The first constraint imposes that a subset of k sources is selected in $\mathcal{S}$. The second constraints ensure that one target T is assigned to a source S only if it is selected. The third inequalities ensure that every target is assigned to exactly one source.

Table 1: Summary of decision variables used in the mathematical programming model

Binary and Integer Decision Variables	
Name	**Description**
$y(S)$	1, if a facility is allocated in the source S in the solution of the model, 0, otherwise.
$x(ST)$	1, if source S and target T are joined by a path in the solution of the model, 0, otherwise.
$f(PQ)$	number of times that edge (P, Q) is traversed in the solution.
$f(PQ\|ST)$	1, if edge (P, Q) is traversed in the path joining S and T, 0, otherwise.
$\alpha(P\|QQ')$	1, if the determinant $\det(P\|QQ')$ is positive, 0, otherwise.
$\beta(PP'\|QQ')$	1, if the determinants $\det(P\|QQ')$ and $\det(P'\|QQ')$ have the same sign, 0, otherwise.
$\gamma(PP'\|QQ')$	1, if the determinants $\det(P\|QQ')$ and $\det(P'\|QQ')$ are both positive, 0, otherwise.
$\delta(PP'\|QQ')$	1, if the line segments $\overline{PP'}$ and $\overline{QQ'}$ do not intersect, 0, otherwise.
$\varepsilon(PP')$	1, if the line segment $\overline{PP'}$ does not cross any barrier, 0, otherwise.
Continuous Decision Variables	
Name	**Description**
P_N	Coordinates representing the point selected in the neighbourhood $N \in \mathcal{S} \cup \mathcal{T}$.
$d(PQ)$	Euclidean distance between the points P and Q.

3.2. Flow constraints

The distances between each pair of source-target neighbourhoods in the formulations are represented by the shortest path joining them without traversing any barrier. Note that, although computing the shortest paths between every pair of neighbourhoods is possible, converting an instance of the H-KMPHN into an instance of the standard k-median is not, since the points in neighbourhoods are not fixed. However, this simplification can be applied to produce an approximation to generate lower bounds for the problem. To model source-target allocation paths, two approaches can be used, involving different variables and constraints.

The first approach is based on a single-commodity formulation. It models a tree routed at each source that connects all targets assigned to that source. The idea is that the model must deliver one unit of commodity from the selected source neighbourhood to each of the required target neighbourhoods.

Then, for each edge $(P,Q) \in E_X$, an integer variable $f(PQ)$ that counts the number of times that edge (P,Q) is traversed in the solution is defined. Then, the flow conservation constraint that represents the path is the following:

$$\sum_{\{Q \in V_X:(P,Q)\in E_X\}} f(PQ) - \sum_{\{Q \in V_X:(Q,P)\in E_X\}} f(PQ) = \begin{cases} \sum_{T\in\mathcal{T}} x(ST), & \text{if } P \in V_{\mathcal{S}}, \\ 0, & \text{if } P \in V_{\mathcal{B}}, \\ -1, & \text{if } P \in V_{\mathcal{T}}, \end{cases} \quad \forall P \in V_X. \tag{single-flow-C}$$

These constraints state that the units of commodity that are delivered from the source must be exactly the number of targets assigned to this source. These units must be transported along the path until an assigned target is reached.

The second approach, related with a multi-commodity scheme, the binary variables $f(PQ|ST)$ must be defined for each $(P,Q) \in E_X$, $S \in \mathcal{S}$ and $T \in \mathcal{T}$. These variables are set to one for those edges that are used to go from the source S to the target T. In this case, the flow conservation constraint reads as follows:

$$\sum_{\{Q \in V_X:(P,Q)\in E_X\}} f(PQ|ST) - \sum_{\{Q \in V_X:(Q,P)\in E_X\}} f(PQ|ST) = \begin{cases} x(ST), & \text{if } P \in S, \\ 0, & \text{if } P \in V_{\mathcal{B}}, \\ -x(ST), & \text{if } P \in T, \end{cases} \quad \forall P \in V_X, \forall S \in \mathcal{S}, \forall T \in \mathcal{T}. \tag{multi-flow-C}$$

These constraints model the path joined the source S and the target T whenever these are assigned by means of variable $x(ST)$.

These two approaches raise different ways to formulate the problem that are later compared in the computational section.

3.3. Conic programming constraints

For the two problems considered in this paper, namely H-KMPHN and H-KMPN, there exist two typologies of second-order cone constraints. One of them models the distance between each pair of points P and Q in V_X, $X \in \{\text{KMPHN}, \text{KMPN}\}$, and the other, the representation of source and target neighbourhoods, where the points are chosen.

Firstly, we define the non-negative continuous variable $d(PQ)$ that represents the distance between P and Q:

$$\|P - Q\| \leq d(PQ), \quad \forall (P,Q) \in E_X, \tag{d-C}$$

where E_X is the set of edges E_{KMPHN} or E_{KMPN}, depending on the considered problem.

Secondly, since we are assuming that the neighbourhoods are second-order cone (SOC) representable, they can be expressed by means of the constraints:

$$P_N \in N \iff \|A_N^i P_N + b_N^i\| \leq (c_N^i)^T P_N + d_N^i, \quad i = 1, \ldots, n(N), \tag{N-C}$$

where A_N^i, b_N^i, c_N^i and d_N^i are parameters of the constraint i and $n(N)$ denotes the number of constraints that appear in the block associated with the neighbourhood $N \in \mathcal{S} \cup \mathcal{T}$.

These inequalities can model the special case of linear constraints (for $A_N^i, b_N^i \equiv 0$), ellipsoids and hyperbolic constraints (see Lobo et al. (1998) and Boyd and Vandenberghe (2004) for more information).

3.4. Checking whether a segment is an edge of the induced graph

The goal of this subsection is to represent by linear constraints a test to check whether given two arbitrary vertices $P, Q \in V_X$, the edge $(P, Q) \in E_X$, with $X \in \{\text{KMPHN}, \text{KMPN}\}$, i.e., whether the line segment $\overline{PQ}$ does not intersect with any barrier of $\mathcal{B}$. The following well-known computational geometry result can be used to check if two line segments intersect.

Remark 1. *Let $\overline{PQ}$ and $B = \overline{P_B^1 P_B^2} \in \mathcal{B}$ be two different line segments. If*

$$\text{sign}\left(\det(P|P_B^1 P_B^2)\right) = \text{sign}\left(\det(Q|P_B^1 P_B^2)\right) \quad \text{or} \quad \text{sign}\left(\det(P_B^1|PQ)\right) = \text{sign}\left(\det(P_B^2|PQ)\right),$$

then $\overline{PQ}$ and B do not intersect.

Let $P, Q \in V_X$, where V_X can be the set of vertices V_{KMPHN} or V_{KMPN}. Let $P_B^1, P_B^2 \in V_{\mathcal{B}}$ also be the two extreme points determining the barrier $B \in \mathcal{B}$. To model the conditions of the Remark 1, the use of binary variables that verify the sign of determinants, the equality of signs, and the disjunctive condition are required, since these determinants depend on the location of P and Q.

Firstly, the sign of each determinant in Remark 1 is modelled. The binary variable α is introduced and assumes the value one if the determinant is non-negative and zero, otherwise. Note that determinants can not be null, because the barriers are located in general position.

The following constraints represent the sign condition:

$$\begin{aligned}
&\left[1 - \alpha(P|P_B^1 P_B^2)\right] L(P|P_B^1 P_B^2) \leq \det(P|P_B^1 P_B^2) \leq U(P|P_B^1 P_B^2)\, \alpha(P|P_B^1 P_B^2), && \forall P \in V_X, \forall P_B^1, P_B^2 \in V_{\mathcal{B}}, \quad (\alpha\text{-C})\\
&\left[1 - \alpha(Q|P_B^1 P_B^2)\right] L(Q|P_B^1 P_B^2) \leq \det(Q|P_B^1 P_B^2) \leq U(Q|P_B^1 P_B^2)\, \alpha(Q|P_B^1 P_B^2), && \forall Q \in V_X, \forall P_B^1, P_B^2 \in V_{\mathcal{B}},\\
&\left[1 - \alpha(P_B^1|PQ)\right] L(P_B^1|PQ) \leq \det(P_B^1|PQ) \leq U(P_B^1|PQ)\, \alpha(P_B^1|PQ), && \forall P, Q \in V_X, \forall P_B^1 \in V_{\mathcal{B}},\\
&\left[1 - \alpha(P_B^2|PQ)\right] L(P_B^2|PQ) \leq \det(P_B^2|PQ) \leq U(P_B^2|PQ)\, \alpha(P_B^2|PQ), && \forall P, Q \in V_X, \forall P_B^2 \in V_{\mathcal{B}},
\end{aligned}$$

where L and U are lower and upper bounds for the value of the corresponding determinants, respectively. If a determinant is non-negative, then α must be one to make the second inequality feasible. Analogously, if the determinant is not positive, α must be zero to satisfy the correct condition.

The fact of considering the possibility of going directly from one to other neighbourhood leads to include product of continuous variables in the α constraints of the model that represent the determinants. These products make the H-KMPN to become non-convex. However, in the H-KMPHN, since two of the three arguments of the determinant are fixed, the α constraints become linear. It motivates the comparison between two formulations in terms of computational cost of considering the more general version, studied in Section 5.

Secondly, to check whether the sign of any pair

$$\det(P|P_B^1 P_B^2),\ \det(Q|P_B^1 P_B^2) \quad \text{or} \quad \det(P_B^1|PQ),\ \det(P_B^2|PQ) \tag{1}$$

of determinants is the same, the binary variable β is defined, that is one if the corresponding pair has the same sign, and zero otherwise.

Hence, the correct value of β variable can be expressed by the following constraint of the α variables

$$\beta(PQ|P_B^1P_B^2) = \alpha(P|P_B^1P_B^2)\alpha(Q|P_B^1P_B^2) + \left[1-\alpha(P|P_B^1P_B^2)\right]\left[1-\alpha(Q|P_B^1P_B^2)\right], \qquad \forall P,Q \in V_X, \forall P_B^1P_B^2 \in V_{\mathcal{B}},$$
$$\beta(P_B^1P_B^2|PQ) = \alpha(P_B^1|PQ)\alpha(P_B^2|PQ) + \left[1-\alpha(P_B^1|PQ)\right]\left[1-\alpha(P_B^2|PQ)\right], \qquad \forall P,Q \in V_X, \forall P_B^1P_B^2 \in V_{\mathcal{B}}.$$

This condition can be equivalently written by means of an auxiliary binary variable γ that models the product of the α variables:

$$\beta(PQ|P_B^1P_B^2) = 2\gamma(PQ|P_B^1P_B^2) - \alpha(P|P_B^1P_B^2) - \alpha(Q|P_B^1P_B^2) + 1, \qquad \forall P,Q \in V_X, \forall P_B^1P_B^2 \in V_{\mathcal{B}}, \qquad (\beta\text{-C})$$
$$\beta(P_B^1P_B^2|PQ) = 2\gamma(P_B^1P_B^2|PQ) - \alpha(P_B^1|PQ) - \alpha(P_B^2|PQ) + 1, \qquad \forall P,Q \in V_X, \forall P_B^1P_B^2 \in V_{\mathcal{B}}.$$

These γ variables can be linearized by using the following constraints:

$$\gamma(PQ|P_B^1P_B^2) \leq \alpha(P|P_B^1P_B^2), \qquad \forall P,Q \in V_X, \forall P_B^1P_B^2 \in V_{\mathcal{B}}, \qquad (\gamma\text{-C})$$
$$\gamma(PQ|P_B^1P_B^2) \leq \alpha(Q|P_B^1P_B^2), \qquad \forall P,Q \in V_X, \forall P_B^1P_B^2 \in V_{\mathcal{B}},$$
$$\gamma(PQ|P_B^1P_B^2) \geq \alpha(P|P_B^1P_B^2) + \alpha(Q|P_B^1P_B^2) - 1, \qquad \forall P,Q \in V_X, \forall P_B^1P_B^2 \in V_{\mathcal{B}},$$

$$\gamma(P_B^1P_B^2|PQ) \leq \alpha(P_B^1|PQ), \qquad \forall P,Q \in V_X, \forall P_B^1P_B^2 \in V_{\mathcal{B}},$$
$$\gamma(P_B^1P_B^2|PQ) \leq \alpha(P_B^2|PQ), \qquad \forall P,Q \in V_X, \forall P_B^1P_B^2 \in V_{\mathcal{B}},$$
$$\gamma(P_B^1P_B^2|PQ) \geq \alpha(P_B^1|PQ) + \alpha(P_B^2|PQ) - 1 \qquad \forall P,Q \in V_X, \forall P_B^1P_B^2 \in V_{\mathcal{B}}.$$

Thirdly, verifying whether there exists any coincidence of the sign of determinants is required, so a binary variable δ is defined assuming the value one if segments do not intersect and zero, otherwise. This condition can be modelled by adopting the following disjunctive constraints:

$$\frac{1}{2}\left[\beta(PQ|P_B^1P_B^2) + \beta(P_B^1P_B^2|PQ)\right] \leq \delta(PQ|P_B^1P_B^2) \leq \beta(PQ|P_B^1P_B^2) + \beta(P_B^1P_B^2|PQ), \quad \forall P,Q \in V_X, \forall P_B^1P_B^2 \in V_{\mathcal{B}}. \qquad (\delta\text{-C})$$

Indeed, the above restrictions state that if there exists a sign coincidence in any of the two pairs of determinants in (1), then δ is one to satisfy the left constraint, and the right one is always fulfilled. However, if none of the signs of any pairs of determinants is the same, then the second constraint is zero and δ must be null.

Finally, to check whether

$$\overline{PQ} \cap B'' = \emptyset, \quad \forall B'' \in \mathcal{B}, \quad \Longleftrightarrow \quad \delta(PQ|P_{B''}^1P_{B''}^2) = 1, \quad \forall B'' \in \mathcal{B},$$

the binary variable $\varepsilon(PQ)$ is introduced, and it is one if this condition is verified for all $B'' \in \mathcal{B}$. This variable can be expressed as:

$$\left[\sum_{B''\in\mathcal{B}} \delta(PQ|P_{B''}^1P_{B''}^2) - |\mathcal{B}|\right] + 1 \leq \varepsilon(PQ) \leq \frac{1}{|\mathcal{B}|}\sum_{B''\in\mathcal{B}} \delta(PQ|P_{B''}^1P_{B''}^2), \quad \forall P,Q \in V_X. \qquad (\varepsilon\text{-C})$$

If there exists, at least, a barrier $B'' \in \mathcal{B}$ that intersects the segment $\overline{PQ}$, then $\delta(PQ|P_{B''}^1P_{B''}^2)$ is zero and the second inequality enforces ε to be zero because the right hand side is fractional and the first inequality is non-positive. However, if no barrier intersects the segment $\overline{PQ}$, then ε is equals to one,

because the left hand side of the first inequality is one and the right hand side of the second inequality too.

It is possible to identify the set of actual edges of graph G_X by using the ε variables based on the above description, as follows:

$$E_X = \{(P,Q) : P,Q \in V_X \wedge \varepsilon(PQ) = 1, P \neq Q\}, \quad X \in \{\text{KMPHN}, \text{KMPN}\}.$$

This representation of E_X with $X \in \{\text{KMPHN}, \text{KMPN}\}$ will be applied in the formulations that are presented in the following subsections.

It is interesting to note that $E_{\mathcal{B}}$ is a fixed set whose edges can be computed by using the Remark 1. Then, ε variables can be prefixed in advance. However, edges in $E_X \setminus E_{\mathcal{B}}$ depend on the points selected in the neighbourhoods. A special case that can be highlighted happens when the set of neighbourhoods, $\mathcal{S}$ and $\mathcal{T}$, are represented by points. In that case, the induced graph is completely fixed and it is only necessary to find which edges are included by keeping in mind that the graph must be planar, i.e., without crossings.

Once edges E_X are represented by means of ε variables, some inequalities must be included to assure that the delivery from P to Q can be produced only if the segment $\overline{PQ}$ does not cross any barrier. These depend on the approach taken to model the paths.

For the single-commodity approach, we have:

$$f(PQ) \leq |\mathcal{T}|\varepsilon(PQ), \quad \forall P,Q \in V_X. \tag{single-f-C}$$

This constraint ensures that if there exists a rectilinear path joining P and Q, the maximum amount of commodity that flows in this path is the number of targets to serve.

For the multi-commodity approach, we state

$$f(PQ|ST) \leq \varepsilon(PQ), \quad \forall (P,Q) \in E_X, \forall S \in \mathcal{S}, \forall T \in \mathcal{T}. \tag{multi-f-C}$$

This inequality states that if target T is assigned to the source S, the vehicle can traverse edge (P,Q) only if it does not cross any barrier.

3.5. Formulations for the Hampered k-Median Problem with Neighbourhoods

The formulations of the H-KMPN are based on the structure of the well-known k-Median Problem where distances between each pair of source-target neighbourhoods are represented by the shortest path joining them without traversing any barrier, as stated before. Putting together all the constraints explained along this section, we can describe two formulations based on the two approaches to model the flow between sources and targets.

The single-commodity formulation adjusted to the induced graph G_X, $X \in \{\text{KMPHN}, \text{KMPN}\}$ is as follows:

$$\begin{aligned}
&\text{minimize} && \omega_E \sum_{(P,Q)\in E_X} d(PQ) f(PQ) + \frac{\omega_L}{2} \sum_{(P,Q)\in E_X} f(PQ) && \text{(H-KMPN-single)}\\
&\text{subject to} && \text{(k-median-C1)} - \text{(k-median-C3)},\\
& && \text{(single-flow-C)}, \text{(single-}f\text{-C)},\\
& && (\alpha\text{-C}), (\beta\text{-C}), (\gamma\text{-C}), (\delta\text{-C}), (\varepsilon\text{-C}),\\
& && (d\text{-C}), (N\text{-C}).
\end{aligned}$$

$$\begin{aligned}
&\text{minimize} && \omega_E \sum_{(P,Q)\in E_X} \sum_{S\in\mathcal{S}} \sum_{T\in\mathcal{T}} d(PQ) f(PQ|ST) + \frac{\omega_L}{2} \sum_{(P,Q)\in E_X} \sum_{S\in\mathcal{S}} \sum_{T\in\mathcal{T}} f(PQ|ST) && \text{(H-KMPN-multi)}\\
&\text{subject to} && \text{(k-median-C1)} - \text{(k-median-C3)},\\
& && \text{(multi-flow-C)}, \text{(multi-}f\text{-C)},\\
& && (\alpha\text{-C}), (\beta\text{-C}), (\gamma\text{-C}), (\delta\text{-C}), (\varepsilon\text{-C}),\\
& && (d\text{-C}), (N\text{-C}).
\end{aligned}$$

$$\begin{aligned}
&\text{minimize} && \alpha_E \sum_{(P,Q)\in E_{\text{KMPHN}}} \sum_{S\in\mathcal{S}} \sum_{T\in\mathcal{T}} d(PQ) f(PQ|ST) + \frac{\alpha_L}{2} \sum_{(P,Q)\in E_{\text{KMPHN}}} \sum_{S\in\mathcal{S}} \sum_{T\in\mathcal{T}} f(PQ|ST) && \text{(H-KMPHN)}\\
&\text{subject to} && \sum_{S\in\mathcal{S}} y(S) = k,\\
& && x(ST) \le y(S), && \forall S\in\mathcal{S},\quad \forall T\in\mathcal{T},\\
& && \sum_{S\in\mathcal{S}} x(ST) = 1, && \forall T\in\mathcal{T},\\
& && \sum_{\{Q\in V_{\text{KMPHN}}:(P,Q)\in E_{\text{KMPHN}}\}} f(PQ|ST) - \sum_{\{Q\in V_{\text{KMPHN}}:(Q,P)\in E_{\text{KMPHN}}\}} f(PQ|ST) = \begin{cases} x(ST), & \text{if } P\in S,\\ 0, & \text{if } P\in V_{\mathcal{B}},\\ -x(ST), & \text{if } P\in T. \end{cases} && \forall S\in\mathcal{S}, \forall T\in\mathcal{T},\\
& && (\alpha\text{-C}), (\beta\text{-C}), (\gamma\text{-C}), (\delta\text{-C}) \quad \forall P,Q\in V_{\text{KMPHN}},\quad \forall P_B^1, P_B^2\in V_{\mathcal{B}},\\
& && (\varepsilon\text{-C}), (??), (d\text{-C}) \quad \forall P,Q\in V_{\text{KMPHN}},\\
& && (N\text{-C}) \quad \forall P\in V_{\mathcal{S}}\cup V_{\mathcal{T}}.
\end{aligned}$$

The objective function takes into account both the Euclidean and link weighted distances to join the selected sources with their assigned targets. The first group of constraints represents the classical k-median. The second group includes the flow conservation constraints and the constraint that relates the possibility of traversing one path with the amount of commodity that can flow in that path. The third group models the visibility graph. Finally, the fourth defines the conic constraints used to model the distance between each pair of points and the neighbourhoods representing sources and targets.

To deal with the bilinear terms that appear in the objective function, the McCormick's envelope are used to linearize them by including variables $p(PQ|ST) \ge 0$ that represent the products and introducing the following constraints:

$$\begin{aligned}
p(PQ|ST) &\ge m(PQ) f(PQ|ST),\\
p(PQ|ST) &\ge d(PQ) - M(PQ)(1 - f(PQ|ST)),
\end{aligned}$$

where $m(PQ)$ and $M(PQ)$ are, respectively, lower and upper bounds of the distance variable $d(PQ)$.

Proposition 1. *The* H-KMPN *is NP-complete.*

Note that, once a point is fixed in each neighbourhood, the problem that results in the induced graph G_{KMPHN} is the k-median with geodesic distances, that is NP-complete. Figure 2 shows the solutions of the problem data in Figure 1 for $k = 1, 2, 3$, respectively.

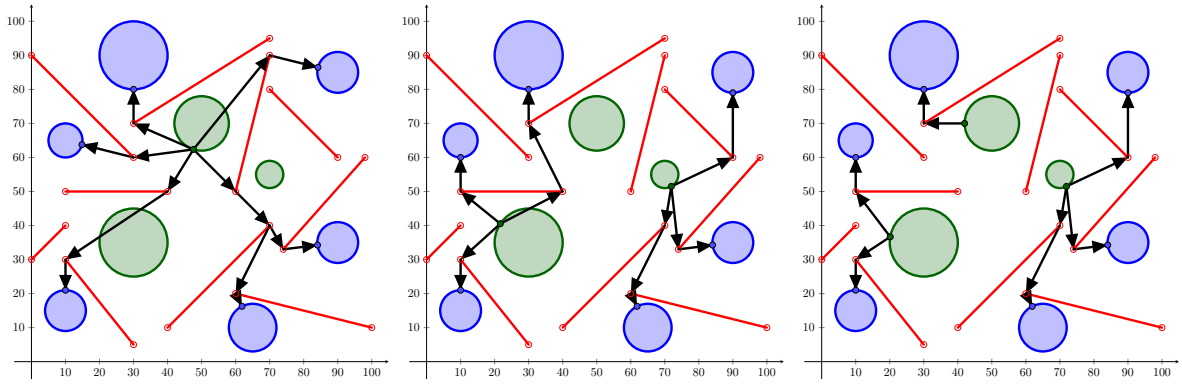

Figure 2: Optimal solution for the H-KMPHN

Again, the problem data in Figure 1, with a smaller number of barriers, is used to illustrate the solutions of the H-KMPN for $k = 1, 2, 3$, respectively.

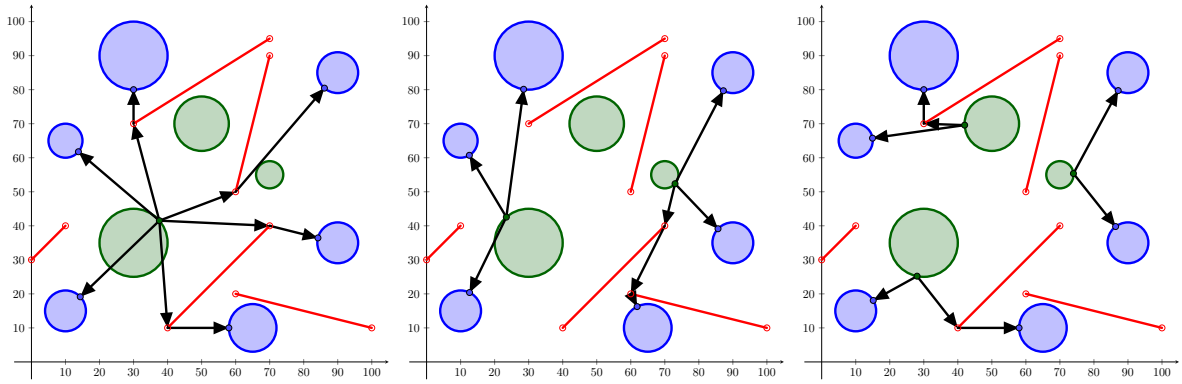

Figure 3: Optimal solution for the H-KMPN

4. Matheuristic Algorithm

The two considered problems are computationally very hard, and, at times, finding even feasible solutions becomes a challenge for large-size instances. In order to provide initial solutions to our algorithms, we have developed an easy procedure based on a reduction to the classical K-median problem with barriers.

In this section, we describe a matheuristic approach based on the formulations presented above, which is able to handle larger instances of the problem. This matheuristic provides solutions of H-KMPHN or H-KMPN, that can be used to initialise the solver for any of the formulations of these models proposed above.

The basic idea of this procedure is to consider only the centres of each neighbourhood as the points chosen in each of them. By fixing those points, the graphs G_{KMPHN} and G_{KMPN} induced in our construction are also fixed, their respective edges can be preprocessed, and the resulting problems become mixed-integer linear. This reduction allows us to obtain feasible solutions for them using any of the available solvers for mixed-integer programming. Furthermore, since the sizes we can handle for the general

problems H-KMPHN and H-KMPN are in the range of 50 to 80 neighbourhoods, solving the point-wise versions takes a short time.

5. Computational Experiments

This section is dedicated to evaluating the performance of the formulations H-KMPHN and H-KMPN and the behaviour of the matheuristic applied to the two problems considered. First, we present details of the data generation. Second, the design of the experiments is stated. Finally, we report the results obtained in our computational experiments.

5.1. Data generation

Assumptions **A1-A4** stated in Section 2 are assumed to generate the instances of the experiments. In this case, w.l.o.g., the neighbourhoods generated are circles. The sketch of the procedure is as follows.

1. Random sampling of points in a square.
2. Generation of bisectors that separate any pair of points.
3. Generation of neighbourhoods that satisfy **A4**.

The following pseudocode describes the details of the construction of the instances.

Algorithm 1: Generation of instances of H-KMPHN

Initialization: Let $|\mathcal{N}|$ be the number of neighbourhoods to generate. Let $r_{\text{init}} = 10$ be half of the initial length of the barriers. Set $\mathcal{N} = \{\}$; $points = \{\}$; $\mathcal{B} = \{\overline{(0,0)(100,0)}, \overline{(100,0)(100,100)}, \overline{(100,100)(0,100)}, \overline{(0,100)(0,0)}\}$.

Generate $|\mathcal{N}|$ points uniformly distributed in the square $[0,100]^2$ and include them in $points$.
for $P, P' \in points$ **do**
  if $\overline{PP'} \cap B = \emptyset$, $\forall B \in \mathcal{B}$ **then**
    Compute $\overrightarrow{d} = \overrightarrow{PP'}$.
    Compute $M = P + \frac{1}{2}\overrightarrow{d}$.
    Compute the unitary vector $\overrightarrow{n_u}$ perpendicular to $\overrightarrow{d}$.
    Set $r = r_{\text{init}}$.
    Generate the barrier $B(r) = \overline{P_B^+ P_B^-}$ where $P_B^{\pm} = M \pm r\overrightarrow{n_u}$.
    while $B(r) \cap B' \neq \emptyset$ *for some* $B' \in \mathcal{B}$ **do**
      Set $r := r/2$.
      Generate the barrier $B(r)$.
    Include $B(r)$ in $\mathcal{B}$.
for $P \in points$ **do**
  Set $r_{\max} = \min_{\{P_B \in B : B \in \mathcal{B}\}} d(P, P_B)$.
  Generate a random *radii* uniformly distributed in the interval $\left[\frac{1}{2}r_{\max}, r_{\max}\right]$.
  Set the ball N whose centre is P and radii is *radii*.
  Include N in $\mathcal{N}$.

Lines 9-11 ensure that neighbourhoods are not enclosed inside of the bisectors so that paths can exit from any neighbourhood to visit another one. Lines 13-17 set a maximum radii for the balls that ensure that they are hidden behind barriers.

Note that, since H-KMPN does not assume **A4**, it is only required to remove some bisectors for the instances generated before to ensure that it is possible to go directly from one neighbourhood to another.

5.2. Configuration of the experiments

To explore the behaviour of the formulations described in the paper, we report on a series of experiments that vary most of the parameters that describe the models. Once they are solved, it is important to give some measures that make them comparable with any others that may be available in the literature. First, the experimental parameters are reported. Then, the computer framework where the experiments were performed is described. Finally, the reported solution values are explained in detail.

Since there are no benchmark instances available for this problem in the literature, five instances for each $|\mathcal{N}| \in \{10, 20, 30, 50, 80\}$ have been generated following Algorithm 1. For each instance, both sources and targets are the whole $\mathcal{N}$ set, i.e., each neighbourhood can be chosen as a source and each one must be associated with a facility. Furthermore, the number of neighbourhood facilities k, for each problem, is 1 and a percentage perc_k, for $\text{perc}_k \in \{10, 25\}$ of the whole set $\mathcal{N}$. To avoid **A4**, a percentage $\text{perc}_{|\mathcal{B}|}$ of all barriers is selected for $\text{perc}_{|\mathcal{B}|} \in \{10, 20, 50\}$. Finally, we set $\alpha_E = 1$ and $\alpha_L = 50$ to take into account both terms of the distances considered in the objective function.

For each combination of the above factors, both H-KMPHN and H-KMPN formulations are tested without and with the initial solution provided by the matheuristic described in Section 4. A time limit of 3600 seconds was set in the experiments for the formulation and 100 seconds for the matheuristic.

Formulations were coded in Python 3.9.2 (G. van Rossum (Guido) (1995)) and solved in Gurobi 9.1.2 (Gurobi Optimization LLC (2022)) on an AMD® Epyc 7402p 8-core processor.

The values obtained by Gurobi that are reported in our tables are:

- $\#Found$: number of instances in which the solver could find a solution.
- Gap: gap between the best incumbent solution with respect to the best bound found by the solver.
- $Runtime$: time spent by the solver to obtain the best solution found.
- Gap_{math}: relative gap between the best incumbent solution given by the matheuristic and the best solution found by the solver after the time limit.
- $Runtime_{math}$: time spent by the matheuristic to obtain the best solution.
- Gap_{build}: relative gap between the value of the first incumbent solution built by the solver and the solution provided by the matheuristic.
- $Runtime_{build}$: time spent by the solver to build the initial solution provided by the matheuristic.

The above information allows us to compare the difficulty of the problem, and, in addition, to test the matheuristic performance to obtain solutions for instances generated as stated in Subsection 5.1.

5.3. Results of the experiments

Analysing the results in Tables 2 and 3, we first note that the solver is capable of finding feasible solutions up to 80 neighbourhoods when the formulation H-KMPHN is considered. Furthermore, these solutions are optimal up to a number of 30 neighbourhoods. Moreover, the solver reports maximum gaps of $14,46\%$ and $31,4\%$ for a number of 50 and 80 neighbourhoods, respectively. It is noteworthy that these gaps are obtained when a 10% of the entire set must be chosen as facilities. For $k = 1$, the maximum gap reported by the solver is $7,84\%$ after the time limit. In addition, this case is considerably the fastest (and the easiest) one to obtain the optimal solution with Gurobi, whenever it is found. Finally, one can remark that the initial solution found by the matheuristic helps to certify optimality in less time and also to get better gaps whenever the optimal solution is not found within the time limit. Note that the maximum gap between the best incumbent solution obtained by the matheuristic in a time limit of 100 seconds and the best by the exact formulation after an hour is 5.96%. This is an indication that the mathheuristic is a good alternative to the exact method whenever the problem size grows.

With regard to the more general problem modelled with formulation H-KMPN, the non-convex nature of the model makes the problem even harder. This can be seen in terms of the resulting gap after the time limit (see Figure 4). The solver starts to not close this gap already for 20 neighbourhoods and half of the original barriers considered. In addition, for instances with 50 and 80 neighbourhoods, the solver cannot find even a feasible solution when it is not initialised. Moreover, Gurobi cannot even build, in some cases, the solution provided by the matheuristic. The case where only one neighbourhood is selected ($k = 1$, e.g., the single facility case) is the hardest one to be solved. This is counterintuitive and is the opposite of the behaviour shown by the formulation H-KMPHN. This case has become the hardest, in our computational experience, because choosing only one point to serve all the required neighbourhoods implies to jointly satisfy a large bunch of non-convex constraints with the same point. This seems to be very difficult for the solver. Another remarkable fact is that the higher the percentage of barriers that are set, the greater the difficulty of the problem, taking into account that the location of the neighbourhood remains the same for all cases. It can be explained in terms of constraints and variables that depend on the number of barriers that are considered. Finally, we can conclude that the matheuristic works well for medium to large-size instances. For those, the matheuristic algorithm always finds a feasible solution that can not be improved by the solver, as can be observed in terms of the relative gap. In addition, in all cases the feasible solutions given by the matheuristic are obtained in a maximum computing time of 100 seconds. This makes the matheuristic an efficient method to provide good-quality solutions in short computing time.

6. Concluding remarks

This paper has dealt with the H-KMPHN and its general version called H-KMPN, in which the assumption that neighbourhoods are not visible to one another is removed. This last version leads to non-convex mixed-integer problems whereas the first one results in second-order cone mixed-integer problems. The two problems, beyond its similarity, show deep differences in terms of computational

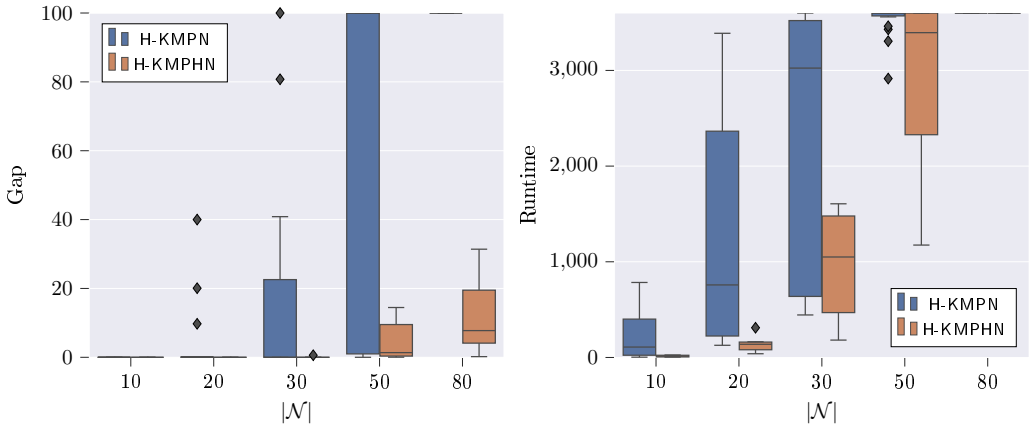

Figure 4: Gap and Runtime reported by Gurobi when the model is initialised by the matheuristic

difficulty, as explained in Section 5. However, the proposed mathematical programming approaches allow a formal treatment that allows one to optimally solve small to medium-size instances. For larger size instances, this approach also inspires a matheuristic algorithm providing good quality solutions, in short computing time, by exploiting the structure of the problem. It is still an open question whether there is some kind of finite dominating set with polynomial cardinality for the version H-KMPHN, which certainly will simplify the underlying graph structures and the solution of the problem. Moreover, given the complexity of the problem, studying valid inequalities that reduce the space of feasible solutions will be instrumental in solving larger instances efficiently.

In addition, one can consider an extension of these problems assuming limited lengths for the paths between the source and its associated target. It would also be interesting to combine in the same model different typologies of barriers such as piecewise linear and second-order cone-representable sets. Besides, it will deserve some attention to study three-dimensional barriers that simulate buildings that planar paths cannot traverse, thus approaching even more real-life applications in the drone delivery industry.

All of the problems mentioned above are natural extensions of those considered in this paper and may attract the attention of researchers in the future.

Acknowledgements

This research has been partially supported by the Agencia Estatal de Investigación (AEI) and the European Regional Development Fund (ERDF): PID2020-114594GB-C21; and Regional Government of Andalusia: project P18-FR-1422.

References

Blanco, V. (2019). Ordered p-median problems with neighbourhoods. *Computational Optimization and Applications*, 73(2):603–645.

Blanco, V., Fernández, E., and Puerto, J. (2017). Minimum Spanning Trees with neighborhoods: Mathematical programming formulations and solution methods. *European Journal of Operational Research*, 262(3):863–878.

Boyd, S. P. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press, Cambridge, UK ; New York.

Daescu, O., Mitchell, J. S. B., Ntafos, S., Palmer, J. D., and Yap, C. K. (2008). An Experimental Study of Weighted k-Link Shortest Path Algorithms. In Akella, S., Amato, N. M., Huang, W. H., and Mishra, B., editors, *Algorithmic Foundation of Robotics VII: Selected Contributions of the Seventh International Workshop on the Algorithmic Foundations of Robotics*, Springer Tracts in Advanced Robotics, pages 187–202. Springer, Berlin, Heidelberg.

de Berg, M., van Kreveld, M., Nilsson, B. J., and Overmars, M. H. (1990). Finding shortest paths in the presence of orthogonal obstacles using a combined L1 and link metric. In Gilbert, J. R. and Karlsson, R., editors, *SWAT 90*, Lecture Notes in Computer Science, pages 213–224, Berlin, Heidelberg. Springer.

Drezner, Z. and Hamacher, H. W., editors (2004). *Facility Location: Applications and Theory*. Springer, Berlin.

G. van Rossum (Guido) (1995). Python reference manual. Issue: R 9525 Publication Title: Department of Computer Science [CS].

Gentilini, I., Margot, F., and Shimada, K. (2013). The travelling salesman problem with neighbourhoods: MINLP solution. *Optimization Methods and Software*, 28(2):364–378. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/10556788.2011.648932.

Gurobi Optimization LLC (2022). Gurobi Optimizer Reference Manual.

Klamroth, K. (2002). *Single-Facility Location Problems with Barriers*. Springer, New York, NY.

Kuehn, A. A. and Hamburger, M. J. (1963). A Heuristic Program for Locating Warehouses. *Management Science*, 9(4):643–666. Publisher: INFORMS.

Lobo, M. S., Vandenberghe, L., Boyd, S., and Lebret, H. (1998). Applications of second-order cone programming. *Linear Algebra and its Applications*, 284(1):193–228.

Mitchell, J. S. B. (2017). Shortest Paths and Networks. In *Handbook of Discrete and Computational Geometry*. Chapman and Hall/CRC, 3 edition. Num Pages: 38.

Nickel, S. and Puerto, J. (2007). S. Nickel and J. Puerto: Location theory: a unified approach. *Mathematical Methods of Operations Research*, 66(2):369–371.

Puerto, J. (2008). A New Formulation of the Capacitated Discrete Ordered Median Problems with $\{0, 1\}$-Assignment. In Kalcsics, J. and Nickel, S., editors, *Operations Research Proceedings 2007*, Operations Research Proceedings, pages 165–170, Berlin, Heidelberg. Springer.

Puerto, J. and Valverde, C. (2022). Routing for unmanned aerial vehicles: Touring dimensional sets. *European Journal of Operational Research*, 298(1):118–136.

Ulukan, Z. and Demircioğlu, E. (2015). A Survey of Discrete Facility Location Problems. *World Academy of Science, Engineering and Technology, International Journal of Industrial and Manufacturing Engineering.*

Yuan, B. and Zhang, T. (2017). Towards Solving TSPN with Arbitrary Neighborhoods: A Hybrid Solution. In *ACALCI*, pages 204–215.

$\|\mathcal{N}\|$	A_4	$\text{perc}_{\|\mathcal{B}\|}$	Initialization	k	$\#Found$	Gap	$Runtime$	Gap_{math}	$Runtime_{math}$	Gap_{build}	$Runtime_{build}$
10	no	10	no	1	5	0	362,95	-	-	-	-
				25%	5	0	6,97	-	-	-	-
			yes	1	5	0	2,94	40,72	0,01	27,94	0,61
				25%	5	0	9,54	54,76	0,01	39,15	0,88
		20	no	1	5	0	387,67	-	-	-	-
				25%	5	0	27,96	-	-	-	-
			yes	1	5	0,01	440,19	32,44	0,01	18,99	10,14
				25%	5	0	27,92	46,2	0,02	13,09	17,97
		50	no	1	5	0,02	452,6	-	-	-	-
				25%	5	0	103,53	-	-	-	-
			yes	1	5	0,01	783,8	28,97	0,02	13,19	70,59
				25%	5	0,01	114	37,36	0,04	29,09	19,08
	yes	100	no	1	5	0	6,16	-	-	-	-
				25%	5	0	25,72	-	-	-	-
			yes	1	5	0	4,19	5,28	0,03	2,97	0,8
				25%	5	0	21,99	4,7	0,26	1,83	1,01
20	no	10	no	1	5	0	127,18	-	-	-	-
				10%	5	0,01	132,99	-	-	-	-
				25%	5	0	175,68	-	-	-	-
			yes	1	5	0,02	775,61	36,23	0,04	8,82	57,03
				10%	5	0	173,2	42,66	0,18	28,25	18,98
				25%	5	0	150,86	56,08	0,03	24,36	60,94
		20	no	1	5	0	740,64	-	-	-	-
				10%	5	0	659,85	-	-	-	-
				25%	5	0	370,03	-	-	-	-
			yes	1	5	0	776,45	25,36	0,06	20,65	42,9
				10%	5	0	1039,52	37,39	0,65	28,61	268,46
				25%	5	0	667,29	50,57	0,15	20,16	257,77
		50	no	1	5	0,19	3044,11	-	-	-	-
				10%	5	0,13	2804,13	-	-	-	-
				25%	5	0,01	1747,37	-	-	-	-
			yes	1	5	9,73	3387,96	12,4	0,12	12,4	62,52
				10%	5	20,07	2852,9	21,62	1,32	20,11	87,53
				25%	5	40	2570,65	25,72	1,06	25,4	41,2
	yes	100	no	1	5	0	66,27	-	-	-	-
				10%	5	0	310,58	-	-	-	-
				25%	5	0	151,38	-	-	-	-
			yes	1	5	0	38,96	5,31	0,16	4,45	3,79
				10%	5	0	164,02	5,96	3,03	4,64	2,92
				25%	5	0	124,26	3,97	4,5	1,77	3,05
30	no	10	no	1	5	0	611,18	-	-	-	-
				10%	5	0	560,1	-	-	-	-
				25%	5	0	454,2	-	-	-	-
			yes	1	5	0	552	30,3	0,2	28,55	35,12
				10%	5	0	718,61	43,19	0,43	33,02	172,55
				25%	5	0	444,33	54,66	0,03	37,28	113,71
		20	no	1	3	0,01	3472,27	-	-	-	-
				10%	3	0	3024,87	-	-	-	-
				25%	4	0	2208,42	-	-	-	-
			yes	1	5	23,01	3240,75	15,44	0,25	13,11	104,78
				10%	5	21,24	3023,67	29,52	7,01	25,05	230,37
				25%	5	0	1933,3	47,09	0,58	47,09	194,12
		50	no	1	1	27,7	3600	-	-	-	-
				10%	1	10,59	3600	-	-	-	-
				25%	1	0	3451,41	-	-	-	-
			yes	1	5	100	3600	0	0,37	0	-
				10%	5	80,75	3600	5,6	22,04	5,6	-
				25%	5	40,85	3536,92	22,57	3,98	22,57	-
	yes	100	no	1	5	0	324,44	-	-	-	-
				10%	5	0	1572,14	-	-	-	-
				25%	5	0,6	1606,06	-	-	-	-
			yes	1	5	0	181,53	2,54	0,37	2,26	8,01
				10%	5	0	902,58	2,93	69,94	2,29	11,44
				25%	5	0	1197,54	2,76	46,68	2,08	12,13

Table 2: Computational results for 10, 20 and 30 neighbourhoods

$\|\mathcal{N}\|$	A_4	perc$_{\|\mathcal{B}\|}$	Initialization	k	$\#Found$	Gap	$Runtime$	Gap_{math}	$Runtime_{math}$	Gap_{build}	$Runtime_{build}$
50	no	10	no	1	1	0	3559,43	-	-	-	-
				10%	1	0	3429,18	-	-	-	-
				25%	4	0	2914,82	-	-	-	-
			yes	1	5	100	3600	0	1,43	0	-
				10%	5	80,01	3600	8,78	7,16	8,78	-
				25%	5	40,07	3461,54	32,39	0,12	32,39	-
		20	no	1	0	-	3600	-	-	-	-
				10%	0	-	3600	-	-	-	-
				25%	2	0,99	3305,34	-	-	-	-
			yes	1	5	100	3600	0	1,67	0	-
				10%	5	100	3600	0	100	0	-
				25%	5	100	3600	0	3,76	0	-
		50	no	1	0	-	3600	-	-	-	-
				10%	0	-	3600	-	-	-	-
				25%	0	-	3600	-	-	-	-
			yes	1	5	100	3600	0	1,94	0	-
				10%	5	100	3600	0	100	0	-
				25%	5	100	3600	0	56,46	0	-
	yes	100	no	1	5	0,03	2041	-	-	-	-
				10%	5	14,46	3600	-	-	-	-
				25%	5	1,39	3600	-	-	-	-
			yes	1	5	0,03	1174,67	2,73	2,54	2,67	26,62
				10%	5	12,22	3600	5,15	100	4,77	16,7
				25%	5	1,32	3189,1	1,44	100	1	26,51
80	no	10	no	1	0	-	3600	-	-	-	-
				10%	0	-	3600	-	-	-	-
				25%	0	-	3600	-	-	-	-
			yes	1	5	100	3600	0	8,55	0	-
				10%	5	100	3600	0	82,05	0	-
				25%	5	100	3600	0	0,42	0	-
		20	no	1	0	-	3600	-	-	-	-
				10%	0	-	3600	-	-	-	-
				25%	0	-	3600	-	-	-	-
			yes	1	5	100	3600	0	8,43	0	-
				10%	5	100	3600	0	100	0	-
				25%	5	100	3600	0	16,16	0	-
		50	no	1	0	-	3600	-	-	-	-
				10%	0	-	3600	-	-	-	-
				25%	0	-	3600	-	-	-	-
			yes	1	5	100	3600	0	10,52	0	-
				10%	5	100	3600	0	100	0	-
				25%	5	100	3600	0	61,43	0	-
	yes	100	no	1	5	7,84	3600	-	-	-	-
				10%	5	31,4	3600	-	-	-	-
				25%	5	7,67	3600	-	-	-	-
			yes	1	5	0,19	3600	2,07	5,48	2,01	60,36
				10%	5	23,38	3600	0,64	100	0,46	33,48
				25%	5	2,93	3600	1,8	100	1,36	30,58

Table 3: Computational results for 50 and 80 neighbourhoods