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. Combining all 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. None of these problems can be seen as a simple incremental contribution since in both cases the tools required to analyze and solve them go beyond any standard overlapping of techniques used in the separated problems. 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 rely on tools borrowed from computational geometry that allow to handle polygonal barriers and neighbours that are second-order cone (SOC) representable, which 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

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

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 (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)).

The k-median problem, one of the most studied facility location models, deals with the optimal placement of one or several new facilities/plants to satisfy the demand of some customers. Here, the positions of both the customers and the potential new facilities are part of the input, as well as the travel costs between them. It was introduced by Hakimi (1965) as a generalisation of the concept of median. The goal of the k-median problem is the location of k facilities within a given region to minimise the total distance between those facilities and a set of given demand points.

One extension of this problem, studied in Blanco (2019), is the Ordered k-Median Problem with Neighbourhoods. It 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. The same rational behind this model also has interest on drone delivery and inspection problems. Indeed, neighbourhoods can represent regions that one 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 a 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).

On the other hand, another studied version of the k-median problem in the literature is the so called k-median problem with barriers (see Klamroth (2002)). In this case, there exist some areas that cannot be traversed, and it is necessary to compute the minimum distance between each pair of points in the graph induced by the facility locations and demand points. In the context of planar location modelling, this problem represents restrictions that appear in a real-life context. One of them can be the case in which there are regions (called forbidden regions) where the placement of a facility is forbidden, but transportation through them it is still possible. They can model parks or regions where by their geographic characteristics it is forbidden the construction of a facility. For a survey of location problems with forbidden regions, see Hamacher and Nickel (1995) or Nickel (1995). The case in which transportation is possible, but only at a higher cost (called congested regions) is studied in Butt and Cavalier (1996) or Mitchell and Papadimitriou (1991). In this case, in these regions, different travel

speeds or travel costs are considered. Finally, the case where transportation is not possible is justified by the existence of military areas, buildings, mountain ranges, lakes, big rivers, or highways that cannot be traversed. Examples of barrier regions involve circuit board design (LaPaugh, 1980), pipe network design for ships (Wangdahl et al., 1974), (**?**) or location and routing with robots (Lozano-Pérez and Wesley, 1979).

This paper introduces the k-median problem with neighbourhoods and barriers, that do not allow transportation through them, i.e., assuming that the trips between service facilities and demand points can not cross them. From a drone routing perspective, these barriers can simulate buildings in urban areas that drones cannot cross. Another application appears in modelling pedestrian routes that must avoid obstacles in urban or rural areas. The resulting problem inherits some elements from the k-median problem with neighbourhoods that must be exploited to partially overcome the difficulties of the solution approaches, but in addition requires new techniques and algorithms from computational geometry to handle the network design among neighbourhoods and with barriers. The combination of all these elements makes this problem an new, attractive challenge in the area of Operations Research.

Nowadays, the advance of technology permits to obtain maps with accurate representations of two-dimensional obstacles by means of various software tools designed for geographic information systems (GIS). An open-source software like Quantum GIS (QGIS Development Team (2009)) provides robust capabilities for creating, editing, and exporting maps with detailed two-dimensional building footprints and attributes. These tools enable users to incorporate building information into spatial analyses, urban planning, and environmental assessments. Some examples where QGIS has been successfully used are described in the following. The work of Arsanjani et al. (2013) explores the use of open-source GIS software for mapping and analyzing urban structures, emphasizing the importance of accurate two-dimensional building data. In drone routing, the work by Mangiameli et al. (2013) proposes an innovative approach for the construction of a map of the obstacles based on Geographic Information Systems (GIS) technologies. In this way, the path planning for the drone navigation is accurately managed by defining precise flight plans on a cartographic support where the obstacles are represented in a three-dimensional way. In summary, GIS software serves as a fundamental resource for exporting maps with two- and three-dimensional building information, contributing to a wide range of spatial analysis and planning endeavors.

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, which is already NP-hard, makes the solution of the problem under study a hard challenge. Off-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. Finally, this problem is applied to a real-world scenario of drone delivery of a small-size instance extracted from a neighbourhood of the city of Cordoba (in the

south of Spain).

The paper is organized in seven 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. A case study based on drone delivery is described in 6. Finally, some conclusions are presented in Section 7.

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.

  In this work, the set of neighbourhoods $\mathcal{N} = \mathcal{S} \cup \mathcal{T}$ are assumed to be compact, closed and convex sets. These sets can be represented by second-order cone constraints as explained in Section 3.3.
- $\mathcal{B}$: 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.

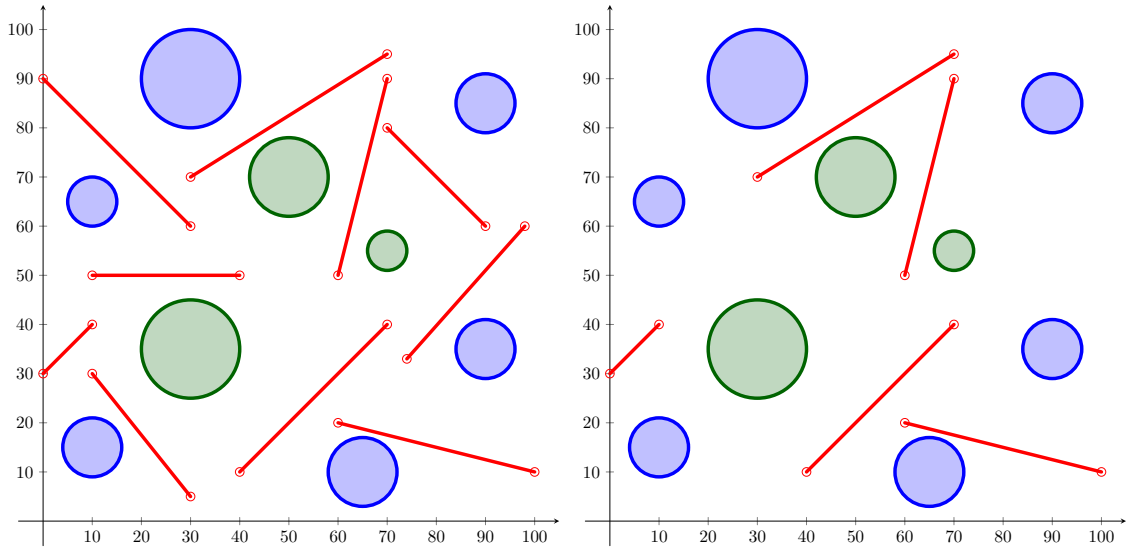


Figure 1: Problem data of the H-KMPHN and H-KMPN

Notation

The following terminology clarifies the notation applied throughout the paper:

- P and Q are referred to as generic points that, at the same time, are identified with their coordinates $P = (P_x, P_y)$ and $Q = (Q_x, Q_y)$, respectively.
- Given two points P^1 and P^2, the line segment that joins P^1 and P^2 is denoted by $\overline{P^1P^2}$.
- Given two points P^1 and P^2, the edge whose vertices are P^1 and P^2 is denoted by (P^1, P^2).
- Given two points P^1 and P^2, the vector pointing from P^1 to P^2 is denoted by $\overrightarrow{P^1P^2}$. It is computed as $\overrightarrow{P^1P^2} = P^2 - P^1$.
- Given three points P^1, P^2 and P^3, $\det(P^1|P^2P^3)$ denotes the following determinant:

$$\det(P^1|P^2P^3) = \det\left(\overrightarrow{P^1P^2} \;\middle|\; \overrightarrow{P^1P^3} \right) := \det\begin{pmatrix} P^2_x - P^1_x & P^3_x - P^1_x \\ P^2_y - P^1_y & P^3_y - P^1_y \end{pmatrix}.$$

  The sign of $\det(P^1|P^2P^3)$ gives the orientation of the point P^1 with respect to the line segment $\overline{P^2P^3}$. Note that $\det(P^1|P^2P^3) \neq 0$ by **A1**.

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, denoted by P_S, in the source set $\mathcal{S}$, at most one in each neighbourhood, and one point in each target set $\mathcal{T}$, denoted by P_T, that minimise 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 $\mathcal{T}$.
- $V_{\mathcal{N}} = V_{\mathcal{S}} \cup V_{\mathcal{T}}$. Set of vertices selected in the set of neighbourhoods $\mathcal{N}$.
- $E_{\mathcal{S}} = \{(P_S, P_B^i) : P_S \in V_{\mathcal{S}}, P_B^i \in V_{\mathcal{B}} \text{ and } \overline{P_S P_B^i} \cap B'' = \emptyset, \forall B'' \in \mathcal{B},\ i = 1, 2\}$. 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 $\mathcal{B}$. $E_{\mathcal{B}}^{int}$ denotes the edges represented by the linear barriers.
- $E_{\mathcal{T}} = \{(P_B^i, P_T) : P_B^i \in V_{\mathcal{B}}, P_T \in V_{\mathcal{T}} \text{ and } \overline{P_B^i P_T} \cap B'' = \emptyset, \forall B'' \in \mathcal{B},\ i = 1, 2\}$. 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 $T \in \mathcal{T}$ that do not cross any other barrier in $\mathcal{B}$.
- $E_{\mathcal{N}} = E_{\mathcal{S}} \cup E_{\mathcal{B}} \cup E_{\mathcal{T}} \cup E_{\mathcal{ST}}$. ***** OJO ***** Set of edges incidents to the set of neighbourhoods $\mathcal{N}$ that do not cross any other barrier in $\mathcal{B}$.

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.

It is interesting to note that this graph can be split into two parts: a fixed graph $G_{\mathcal{B}} = (V_{\mathcal{B}}, E_{\mathcal{B}})$ whose edges can be computed beforehand, and the sets $V_{\mathcal{N}}$, $E_{\mathcal{N}}$, that depend on where the points P_N

are located as shown in Figures 2 and 3. The two subfigures show how the graph G is generated. The blue dashed line segments represent the edges of $E_{\mathcal{S}}$, the green dashed lines, the edges of $E_{\mathcal{T}}$ and the red dashed lines, the edges of $E_{\mathcal{B}}$. This explains that the difficulty to determine the visibility graph focused on the sets of edges that depend on the location of the variable points on the neighboors, that is, $V_{\mathcal{N}}$ and $E_{\mathcal{N}}$. Hence, the aim of the following section is to find the tools to build a valid formulation and to reduce the size of the problem.

Figure 2: Generation of the visibility graph. Case 1

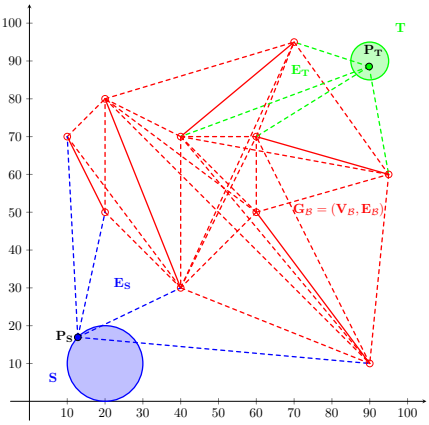
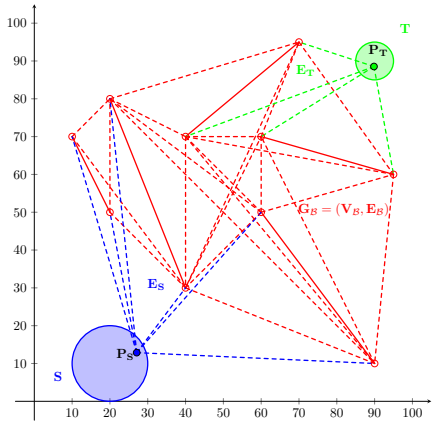


Figure 3: Generation of the visibility graph. Case 2



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_X is the k-median with geodesic distances. This problem is a version of the discrete k-median in which the distances are computed beforehand by solving a shortest-path problem for each pair source-target in G_X. Hence, it is NP-complete by a reduction to k-median (Kariv and Hakimi, 1979).

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 feasible region are set. Then, two different approaches to ensure connectivity between each pair source-target are presented. Later, we give the conic programming representation of the neighbourhoods and distance. Finally, we describe the constraints that check if a segment is included in the set of edges E_X with $X \in \{\text{KMPHN}, \text{KMPN}\}$. As far as we are concerned, this family of constraints is new and they are presented for the first time in this paper. The reader may realize that they are based on tools borrowed from computational geometry (de Berg et al., 1990; Daescu et al., 2008).

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

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

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

Binary and Integer Decision Variables

Name	Domain	Range	Description
y^S	$S \in \mathcal{S}$	$\mathbb{Z}_2$	1, if a facility is allocated in the source S in the solution of the model, 0, otherwise.
x^{ST}	$(S,T) \in \mathcal{S} \times \mathcal{T}$	$\mathbb{Z}_2$	1, if source S and target T are joined by a path in the solution of the model, 0, otherwise.
$f_{PP'}$	$(P,P') \in E_X$	$\mathbb{Z}_{\lvert\mathcal{T}\rvert}$	number of times that edge (P,P') is traversed in the solution.
$f^{ST}_{PP'}$	$(P,P',S,T) \in E_X \times \mathcal{S} \times \mathcal{T}$	$\mathbb{Z}_2$	1, if edge (P,P') is traversed in the path joining S and T, 0, otherwise.
$\alpha_{PQQ'}$	$(P,Q,Q') \in \mathcal{A} = (V_{\mathcal{N}} \times E^{int}_{\mathcal{B}}) \cup (V_{\mathcal{B}} \times E_{\mathcal{N}})$	$\mathbb{Z}_2$	1, if the determinant $\det(P\lvert QQ')$ is positive, 0, otherwise.
$\delta_{PP'QQ'}$	$(P,P',Q,Q') \in \mathcal{D} = (E_{\mathcal{N}} \times E^{int}_{\mathcal{B}}) \cup (E^{int}_{\mathcal{B}} \times E_{\mathcal{N}})$	$\mathbb{Z}_2$	1, if the determinants $\det(P\lvert QQ')$ and $\det(P'\lvert QQ')$ have the same sign, 0, otherwise.
$\varepsilon_{PP'QQ'}$	$(P,P',Q,Q') \in \mathcal{D}$	$\mathbb{Z}_2$	1, if the determinants $\det(P\lvert QQ')$ and $\det(P'\lvert QQ')$ are both positive, 0, otherwise.
$\zeta_{PP'QQ'}$	$(P,P',Q,Q') \in \mathcal{Z} = E_{\mathcal{N}} \times E^{int}_{\mathcal{B}}$	$\mathbb{Z}_2$	1, if the line segments $\overline{PP'}$ and $\overline{QQ'}$ do not intersect, 0, otherwise.
$\eta_{PP'}$	$(P,P') \in E_{\mathcal{N}}$	$\mathbb{Z}_2$	1, if the line segment $\overline{PP'}$ does not cross any barrier, 0, otherwise.
Continuous Decision Variables			
P_N	$N \in \mathcal{N} = \mathcal{S} \cup \mathcal{T}$	N	Coordinates representing the point selected in the neighbourhood N.
$d_{PP'}$	$(P,P') \in E_{\mathcal{N}}$	$\mathbb{R}^+$	Euclidean distance between the points P and P'.

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.

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 $(P, P') \in E_X$, an integer variable $f_{PP'}$ that counts the number of times that edge (P, P') is traversed in the solution is defined. Then, the flow conservation constraint that represents the path is the following:

$$\sum_{\{P' \in V_X : (P,P') \in E_X\}} f_{PP'} - \sum_{\{P' \in V_X : (P',P) \in E_X\}} f_{PP'} = \begin{cases} \sum_{T \in \mathcal{T}} x^{ST}, & \text{if } P \in V_{\mathcal{S}}, \\ 0, & \text{if } P \in V_{\mathcal{B}}, \\ -1, & \text{if } P \in V_{\mathcal{T}}, \end{cases} \quad \forall P \in V_X. \quad \text{(single-flow-C)}$$

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, requires the binary variables $f^{ST}_{PP'}$ to be defined for each $(P, P') \in E_X$, $S \in \mathcal{S}$ 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:

$$\sum_{\{P' \in V_X : (P,P') \in E_X\}} f^{ST}_{PP'} - \sum_{\{P' \in V_X : (P',P) \in E_X\}} f^{ST}_{PP'} = \begin{cases} x^{ST}, & \text{if } P \in S, \\ 0, & \text{if } P \in V_{\mathcal{B}}, \\ -x^{ST}, & \text{if } P \in T, \end{cases} \quad \forall P \in V_X, \forall S \in \mathcal{S}, \forall T \in \mathcal{T}.$$

(multi-flow-C)

These constraints model the path joining the source S and the target T whenever these two 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 P' in V_X, $X \in \{\text{KMPHN}, \text{KMPN}\}$, and the other one, the representation of source and target neighbourhoods, where the points are chosen.

Firstly, we define the non-negative continuous variable $d_{PP'}$ that represents the distance between P and P':

$$\|P - P'\| \leq d_{PP'}, \quad \forall (P, P') \in E_{\mathcal{N}}, \qquad (d\text{-C})$$

where $E_{\mathcal{N}}$ is the set of edges incident to the set of neighbourhoods $\mathcal{N}$ in 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:

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

where A_N^i, b_N^i, c_N^i and d_N^i are parameters of the constraint i, $n(N)$ denotes the number of constraints that appear in the block associated with the neighbourhood $N \in \mathcal{N}$ and P_N represents the coordinates of the variable point selected in N.

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 construct a test representable by linear constraints to check whether given two arbitrary vertices $P, P' \in V_X$, the edge (P, P') belongs to E_X, with $X \in \{\text{KMPHN}, \text{KMPN}\}$, i.e., whether the line segment $\overline{PP'}$ does not intersect with any barrier of $\mathcal{B}$. The reader may note that this is a new contribution to the field that can be used in future in different problems involving barriers.

The following result borrowed from computational geometry is instrumental to check if two line segments intersect.

Remark 1. *Let $\overline{PP'}$ and $\overline{QQ'}$ be two different line segments. If*

$$\text{sign}\left(\det(P|QQ')\right) = \text{sign}\left(\det(P'|QQ')\right) \quad \text{or} \quad \text{sign}\left(\det(Q|PP')\right) = \text{sign}\left(\det(Q'|PP')\right),$$

then $\overline{PP'}$ and $\overline{QQ'}$ do not intersect.

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, P', Q and Q'.

Firstly, the sign of each determinant in Remark 1 is modelled with (α-C). 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:

$$[1 - \alpha_{PQQ'}]\, L_{PQQ'} \leq \det(P|QQ') \leq U_{PQQ'}\, \alpha_{PQQ'}, \quad \forall (P, Q, Q') \in \mathcal{A}, \tag{α-C}$$

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 neighbourhood to another leads to include product of continuous variables, presented in the determinants of the α constraints of the model. Specifically, for the tuples $(P, Q, Q') \in V_{\mathcal{B}} \times E_{\mathcal{ST}}$, the determinant $\det(P|QQ')$ produces bilinear

terms. These products make the general H-KMPN to become non-convex. However, for the hidden version, since two of the three arguments of the determinant are fixed because $E_{\mathcal{ST}}$ is not considered, the α constraints become linear. This difference justifies the comparison between the two formulations in terms of computational cost studied in Section 5.

Secondly, to check whether the sign of any pair

$$\det(P|QQ'),\ \det(P'|QQ') \quad \text{or} \quad \det(Q|PP'),\ \det(Q'|PP') \tag{1}$$

of determinants is the same, a binary variable δ is defined which assumes the value one if the corresponding pair has the same sign, and zero otherwise.

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

$$\delta_{PP'QQ'} = \alpha_{PQQ'}\alpha_{P'QQ'} + [1-\alpha_{PQQ'}]\,[1-\alpha_{P'QQ'}]\,, \quad \forall (P,P',Q,Q') \in \mathcal{D}.$$

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

$$\delta_{PPQQ'} = 2\varepsilon_{PP'QQ'} - \alpha_{PQQ'} - \alpha_{P'QQ'} + 1, \quad \forall (P,P',Q,Q') \in \mathcal{D}. \tag{δ-C}$$

The product of binary variables that define ε can be linearised by means of the McCormick's envelope:

$$\begin{aligned}
&\varepsilon_{PP'QQ'} \leq \alpha_{PQQ'}, && \forall (P,P',Q,Q') \in \mathcal{D}, \qquad (\varepsilon\text{-C})\\
&\varepsilon_{PP'QQ'} \leq \alpha_{P'QQ'}, && \forall (P,P',Q,Q') \in \mathcal{D},\\
&\varepsilon_{PP'QQ'} \geq \alpha_{PQQ'} + \alpha_{P'QQ'} - 1, && \forall (P,P',Q,Q') \in \mathcal{D}.
\end{aligned}$$

Thirdly, verifying whether there exists any coincidence of the sign of determinants is required, so a binary variable ζ 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:

$$\frac{1}{2}\left[\delta_{PP'QQ'} + \delta_{QQ'PP'}\right] \leq \zeta_{PP'QQ'} \leq \delta_{PP'QQ'} + \delta_{QQ'PP'}, \quad \forall (P,P',Q,Q') \in \mathcal{Z}. \tag{ζ-C}$$

Indeed, the above restrictions state that if there exists a sign coincidence in any of the two pairs of determinants in (1), then ζ 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 ζ must be null.

Finally, to check whether

$$\overline{PP'} \cap \overline{QQ'} \neq \emptyset, \quad \forall \overline{QQ'} \in \mathcal{B}, \quad \Longleftrightarrow \quad \zeta_{PP'QQ'} = 1, \quad \forall \overline{QQ'} \in \mathcal{B},$$

the binary variable $\eta_{PP'}$ is introduced, and it is one if this condition is verified for all $\overline{QQ'} \in \mathcal{B}$. This variable can be expressed as:

$$\left[\sum_{\overline{QQ'}\in\mathcal{B}} \zeta_{PP'QQ'} - |\mathcal{B}|\right] + 1 \leq \eta_{PP'} \leq \frac{1}{|\mathcal{B}|}\sum_{\overline{QQ'}\in\mathcal{B}} \zeta_{PP'QQ'}, \quad \forall (P,P') \in \mathcal{E}. \tag{η-C}$$

If there exists, at least, a barrier $\overline{QQ'} \in \mathcal{B}$ that intersects the segment $\overline{PP'}$, then $\zeta_{PP'QQ'}$ is zero and the second inequality enforces η to be zero because the right hand side is fractional and the first inequality is non-positive. However, if no barrier intersects the segment $\overline{PP'}$, then η 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 η variables based on the above description, as follows:

$$E_X = \{(P, P') : P, P' \in V_X, \eta_{PP'} = 1, P \neq P'\}, \quad X \in \{\text{KMPHN}, \text{KMPN}\}.$$

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.

A special case that can be highlighted happens when the set of neighbourhoods $\mathcal{N}$ is 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 η variables, some inequalities must be included to assure that the delivery from P to P' can be produced only if the segment $\overline{PP'}$ does not cross any barrier. They depend on the approach taken to model the paths.

For the single-commodity approach, we have:

$$f_{PP'} \leq |\mathcal{T}|\eta_{PP'}, \quad \forall P, P' \in V_X. \tag{single-f-C}$$

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

For the multi-commodity approach, we state

$$f^{ST}_{PP'} \leq \eta_{PP'}, \quad \forall (P, P') \in E_X, \forall S \in \mathcal{S}, \forall T \in \mathcal{T}. \tag{multi-f-C}$$

This inequality ensures that if target T is assigned to the source S, the drone can traverse edge (P, P') 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 k-Median Problem but imposing that distances between each pair of source-target neighbourhoods are represented by the shortest path joining them without traversing any barrier, as stated before. The reader may realize that this is far from trivial since elements in neighbourhoos are variable and paths cannot cross. 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:

$$\text{minimize} \quad \omega_E \left(\sum_{(P,P')\in E_{\mathcal{N}}} d_{PP'} f_{PP'} + \sum_{(P,P')\in E_{\mathcal{B}}} \|P - P'\| f_{PP'} \right) + \frac{\omega_L}{2} \sum_{(P,P')\in E_X} f_{PP'} \tag{H-KMPN-single}$$

$$\begin{aligned}
\text{subject to} \quad & \text{(k-median-C1)} - \text{(k-median-C3)}, \\
& \text{(single-flow-C)}, \text{(single-}f\text{-C)}, \\
& (\alpha\text{-C}), (\delta\text{-C}), (\varepsilon\text{-C}), (\zeta\text{-C}), (\eta\text{-C}), \\
& (d\text{-C}), (\mathcal{N}\text{-C}), \\
& y \in \mathbb{Z}_2^{\mathcal{S}}, x \in \mathbb{Z}_2^{\mathcal{S}\times\mathcal{T}}, f \in \mathbb{Z}_{|\mathcal{T}|}^{E_X}, \\
& \alpha \in \mathbb{Z}_2^{\mathcal{A}}, \delta, \varepsilon \in \mathbb{Z}_2^{\mathcal{D}}, \zeta \in \mathbb{Z}_2^{\mathcal{Z}}, \eta \in \mathbb{Z}_2^{E_{\mathcal{N}}}.
\end{aligned}$$

The objective function takes into account both the Euclidean and link weighted distances to join the selected sources with their assigned targets. The term related with the Euclidean distance is split into two parts: the first one accounts for the variable distances, while the distances in the second one are known beforehand. Inspired in a drone routing application, the link distance refers to the rotation cost of the vehicle (see Maheshwari et al. (2000) for more details). The first group of constraints represents the classical k-median. The second group includes the flow conservation constraints and the constraint that relate the possibility of traversing one path with the amount of commodity that can flow in that path. The third group models the visibility graph. The fourth block defines the conic constraints used to model the distance between each pair of points and the representation of the sources and targets. Finally, lines five and six describe the domain of the variables.

In order to linearise the bilinear terms defined in the first part of the objective function, it is necessary to transform the integer variable as a linear combination of binary indicator variables, $f^t_{aux,PP'}$, that attains the value one if edge $(P,P') \in E_{\mathcal{N}}$ is visited t times. Hence, for each edge $(P,P') \in E_{\mathcal{N}}$, the representation of f in terms of f_{aux} variables can be described as follows:

$$\sum_{t=0}^{|\mathcal{T}|} t f^t_{aux,PP'} = f_{PP'}, \tag{f-representation-C1}$$

$$\sum_{t=0}^{|\mathcal{T}|} f^t_{aux,PP'} = 1. \tag{f-representation-C2}$$

Then, to deal with the product of the binary variable f^t_{aux} with the continuous variable d in the transformed objective function, the McCormick's envelope is used to linearise them. For each $(P,P') \in E_{\mathcal{N}}$ and $t = 0, \ldots, |\mathcal{T}|$, we define variables $prod^t \geq 0$ to represent the products and the following constraints are introduced:

$$prod^t_{PP'} \geq m_{PP'} f^t_{aux,PP'}, \tag{McCormick-C1}$$

$$prod^t_{PP'} \geq d_{PP'} - M_{PP'}(1 - f^t_{aux,PP'}), \tag{McCormick-C2}$$

where $m_{PP'}$ and $M_{PP'}$ are, respectively, lower and upper bounds of the distance variable $d_{PP'}$.

The equivalent formulation that is obtained when the objective function is linearised is stated as:

$$\text{minimize} \quad \omega_E \left(\sum_{t=0}^{|\mathcal{T}|} \sum_{(P,P') \in E_{\mathcal{N}}} prod^t_{PP'} + \sum_{(P,P') \in E_{\mathcal{B}}} \|P - P'\| f_{PP'} \right) + \frac{\omega_L}{2} \sum_{(P,P') \in E_X} f_{PP'} \tag{H-KMPN-single}$$

$$\begin{aligned}
\text{subject to} \quad & \text{(k-median-C1)} - \text{(k-median-C3)}, \\
& \text{(single-flow-C)}, \text{(single-}f\text{-C)}, \\
& (\alpha\text{-C}), (\delta\text{-C}), (\varepsilon\text{-C}), (\zeta\text{-C}), (\eta\text{-C}), \\
& (d\text{-C}), (\mathcal{N}\text{-C}), \\
& (f\text{-representation-C1}) - (f\text{-representation-C2}), \\
& \text{(McCormick-C1)} - \text{(McCormick-C2)}, \\
& y \in \mathbb{Z}_2^{\mathcal{S}}, x \in \mathbb{Z}_2^{\mathcal{S} \times \mathcal{T}}, f \in \mathbb{Z}_{|\mathcal{T}|}^{E_X}, f^t_{aux}, prod^t \in \mathbb{Z}_2^{E_{\mathcal{N}} \times \mathcal{T}}, \\
& \alpha \in \mathbb{Z}_2^{\mathcal{A}}, \delta, \varepsilon \in \mathbb{Z}_2^{\mathcal{D}}, \zeta \in \mathbb{Z}_2^{\mathcal{Z}}, \eta \in \mathbb{Z}_2^{E_{\mathcal{N}}}.
\end{aligned}$$

The H-KMPN can be also modelled by means of a multi-commodity approach. It can be described as follows:

$$\text{minimize} \quad \omega_E \sum_{S \in \mathcal{S}} \sum_{T \in \mathcal{T}} \left(\sum_{(P,P') \in E_{\mathcal{N}}} d_{PP'} f^{ST}_{PP'} + \sum_{(P,P') \in E_{\mathcal{B}}} \|P - P'\| f^{ST}_{PP'} \right) + \frac{\omega_L}{2} \sum_{S \in \mathcal{S}} \sum_{T \in \mathcal{T}} \sum_{(P,P') \in E_X} f^{ST}_{PP'} \tag{H-KMPN-multi}$$

$$\begin{aligned}
\text{subject to} \quad & \text{(k-median-C1)} - \text{(k-median-C3)}, \\
& \text{(multi-flow-C)}, \text{(multi-}f\text{-C)}, \\
& (\alpha\text{-C}), (\delta\text{-C}), (\varepsilon\text{-C}), (\zeta\text{-C}), (\eta\text{-C}), \\
& (d\text{-C}), (\mathcal{N}\text{-C}), \\
& y \in \mathbb{Z}_2^{\mathcal{S}}, x \in \mathbb{Z}_2^{\mathcal{S} \times \mathcal{T}}, f \in \mathbb{Z}_2^{E_X \times \mathcal{S} \times \mathcal{T}}, \\
& \alpha \in \mathbb{Z}_2^{\mathcal{A}}, \delta, \varepsilon \in \mathbb{Z}_2^{\mathcal{D}}, \zeta \in \mathbb{Z}_2^{\mathcal{Z}}, \eta \in \mathbb{Z}_2^{E_{\mathcal{N}}}.
\end{aligned}$$

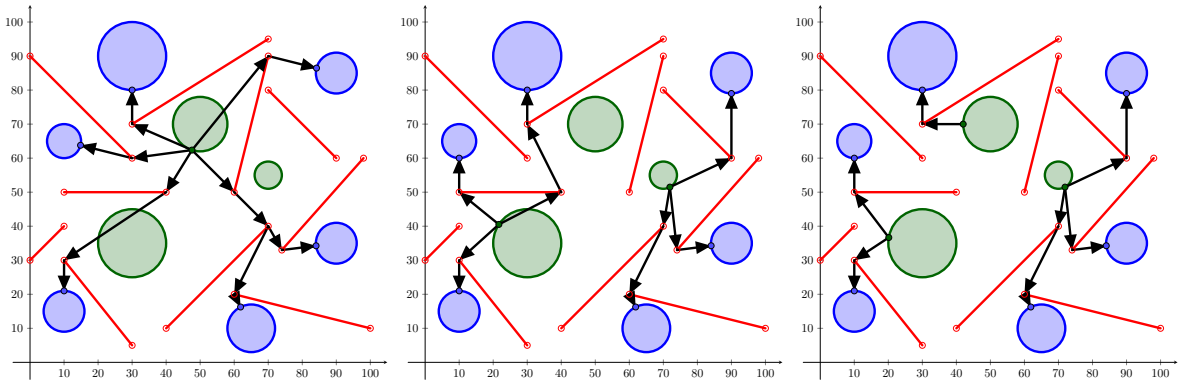
In this case, the objective function considers both Euclidean and link-weighted distances to connect the chosen sources with their designated targets. The initial set of constraints are the k-median constraints, while the second set encompasses flow conservation and the traversability of a path with the quantity of commodity that can flow through it. The third set characterizes the visibility graph, and the fourth describes conic constraints adopted to represent the distance between every pair of points and the domain of sources and targets. Lastly, constraints in lines 5 and 6 describe variables domains. Once more, the bilinear terms in the objective function can be linearised by using McCormick's envelope by defining the corresponding continuous variable $prod^{ST}_{PP'} \geq 0$. Thus, an equivalent formulation is:

$$\text{minimize} \quad \omega_E \sum_{S\in\mathcal{S}}\sum_{T\in\mathcal{T}}\left(\sum_{(P,P')\in E_{\mathcal{N}}} prod^{ST}_{PP'} + \sum_{(P,P')\in E_{\mathcal{B}}} \|P-P'\| f^{ST}_{PP'}\right) + \frac{\omega_L}{2}\sum_{S\in\mathcal{S}}\sum_{T\in\mathcal{T}}\sum_{(P,P')\in E_X} f^{ST}_{PP'} \tag{H-KMPN-multi}$$

$$\begin{aligned}
\text{subject to} \quad & \text{(k-median-C1)} - \text{(k-median-C3)},\\
& \text{(multi-flow-C)}, \text{(multi-}f\text{-C)},\\
& (\alpha\text{-C}), (\delta\text{-C}), (\varepsilon\text{-C}), (\zeta\text{-C}), (\eta\text{-C}),\\
& (d\text{-C}), (\mathcal{N}\text{-C}),\\
& \text{(McCormick-C1)} - \text{(McCormick-C2)},\\
& y \in \mathbb{Z}_2^{\mathcal{S}}, x \in \mathbb{Z}_2^{\mathcal{S}\times\mathcal{T}}, f \in \mathbb{Z}_2^{E_X\times\mathcal{S}\times\mathcal{T}}, prod \in \mathbb{Z}_2^{E_{\mathcal{N}}\times\mathcal{S}\times\mathcal{T}},\\
& \alpha \in \mathbb{Z}_2^{\mathcal{A}}, \delta, \varepsilon \in \mathbb{Z}_2^{\mathcal{D}}, \zeta \in \mathbb{Z}_2^{\mathcal{Z}}, \eta \in \mathbb{Z}_2^{E_{\mathcal{N}}}.
\end{aligned}$$

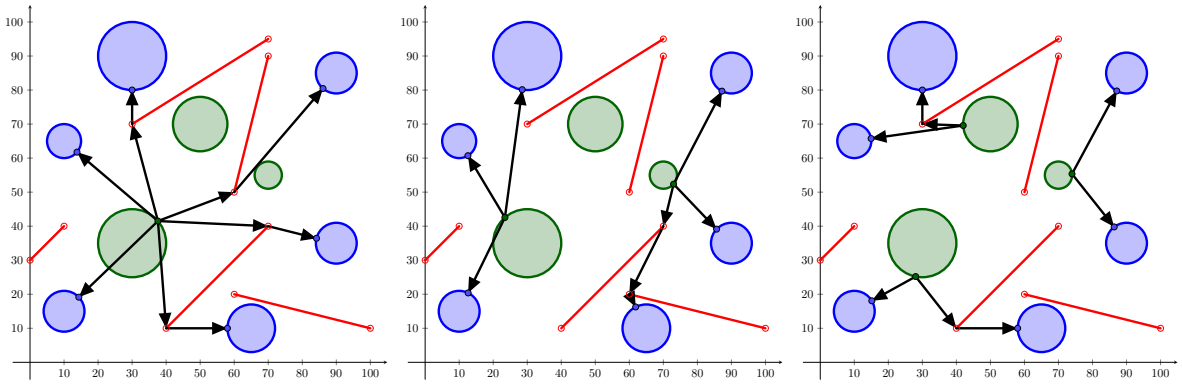
Figure 4 shows the solutions of the problem data in Figure 1 for $k = 1, 2, 3$, respectively.

Figure 4: 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 5: 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 70 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, wlog, 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 $\mathcal{N} = \{\}$; $points = \{\}$; $\mathcal{B} = \{\overline{(0,0)(100,0)}, \overline{(100,0)(100,100)}, \overline{(100,100)(0,100)}, \overline{(0,100)(0,0)}\}$.

1 Generate $|\mathcal{N}|$ points uniformly distributed in the square $[0,100]^2$ and include them in $points$.
2 **for** $P, P' \in points$ **do**
3 **if** $\overline{PP'} \cap B = \emptyset$, $\forall B \in \mathcal{B}$ **then**
4 Compute $\overrightarrow{d} = \overrightarrow{PP'}$.
5 Compute $M = P + \frac{1}{2}\overrightarrow{d}$.
6 Compute the unitary vector $\overrightarrow{n_u}$ perpendicular to $\overrightarrow{d}$.
7 Set $r = r_{\text{init}}$.
8 Generate the barrier $B(r) = \overline{P_B^+ P_B^-}$ where $P_B^{\pm} = M \pm r\overrightarrow{n_u}$.
9 **while** $B(r) \cap B' \neq \emptyset$ *for some* $B' \in \mathcal{B}$ **do**
10 Set $r := r/2$.
11 Generate the barrier $B(r)$.
12 Include $B(r)$ in $\mathcal{B}$.
13 **for** $P \in points$ **do**
14 Set $r_{\max} = \min_{\{P_B \in B : B \in \mathcal{B}\}} d(P, P_B)$.
15 Generate a random $radii$ uniformly distributed in the interval $\left[\frac{1}{2}r_{\max}, r_{\max}\right]$.
16 Set the ball N whose centre is P and radii is $radii$.
17 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 in bold. 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 $|\boldsymbol{\mathcal{N}}| \in \{10, 20, 30, 50, 70\}$ have been generated following Algorithm 1. We recall that this algorithm generates a number of barriers $|\mathcal{B}|$ required to ensure that the neighbourhoods are hidden. To avoid **A4**, the ceil of a sample in terms of the percentage $\lceil \%|\boldsymbol{\mathcal{B}}| \rceil$ of all barriers generated initially by Algorithm 1 is selected for $\%|\boldsymbol{\mathcal{B}}| \in \{10, 20, 50\}$. Furthermore, the number of neighbourhood facilities k, for each

problem, is 1 and a percentage $\lceil \%\boldsymbol{k} \rceil$, for $\%\boldsymbol{k} \in \{10\%|\mathcal{N}|, 25\%|\mathcal{N}|\}$ of the whole set $\mathcal{N}$. Finally, we set $\omega_E = 1$ and $\omega_L \in \{0, 50\}$ to assess the influence of the link distance in the resolution of the problem.

It is assumed that $\mathcal{S} = \mathcal{T} = \mathcal{N}$, i.e., each neighbourhood can be chosen as a source and each one must be associated with a facility. For each combination of the above factors, both approaches for the 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 600 seconds for the matheuristic.

Table 2 summarises all the parameters that are combined to study the behaviour of the problem studied in this manuscript.

Table 2: Table of parameters used in the experiments

symbol	description	values
Approach	formulation approach	single, multi
Initialisation	solver is initialised by means of the matheuristic	without, with
$\lvert\mathcal{N}\rvert$	number of neighbourhoods	10, 20, 30, 50, 70
$\boldsymbol{\omega_E}$	weight for the Euclidean distance	1
$\boldsymbol{\omega_L}$	weight for the link distance	0, 50
$\lceil \%\lvert\boldsymbol{\mathcal{B}}\rvert \rceil$	ceil of the percentage of all barriers generated by Algorithm	H-KMPN: 10%, 20%, 50%; H-KMPHN: 100%
$\boldsymbol{k}$	number of neighbourhoods selected	**1**, $\lceil 10\%\lvert\mathcal{N}\rvert \rceil$, $\lceil 25\%\lvert\mathcal{N}\rvert \rceil$

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:

- **% Gap**: gap between the best incumbent solution with respect to the best bound found by the solver.
- **Runtime**: time (in seconds) spent by the solver to obtain the best solution.
- **% Unfound**: number of instances in which the solver could find a solution.
- **% 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 matheuristic to build the first incumbent solution.

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. If

we consider Gap_{build}, we can study how much the solver can improve the first solution that it builds starting from the solution generated by the matheuristic.

5.3. Results of the experiments *** Not revised yet ****

In this subsection, the results obtained for these experiments are analysed in terms of the values reported by Gurobi, described in the previous subsection. The average gaps, runtimes and percentage of instances whose a feasible is not found that have been obtained for each size of neighbourhoods and approach can be found in Tables 3, 5, 6. 4 also reports the average relative gaps for the best solution found by the matheuristic and the best solution after the time limit, and the relative gap between the solution provided by the matheuristic and the one built by the solver from the matheuristic. It also reports the spent time to build both solutions. Each table refers to a single parameter and the subcolumns, each value of this parameter. The idea is to check if by changing the value of a single parameter, we can find a difference in terms of performance of each size and each approach. These tables are summarised by means of the boxplots in Figures 6 and Figures 7.

Table 3: Results depending of ω_L

		% Gap		Runtime		% Unfound	
	ω_L	**0**	**50**	**0**	**50**	**0**	**50**
$\|\mathcal{N}\|$	**Approach**						
10	**single**	0	0	53	140.14	0	0
	multi	0	0	20.63	102.94	0	0
20	**single**	1.58	0.33	1876.44	1275.27	0	0
	multi	0.61	0.12	1132.57	752.92	0	0
30	**single**	11.77	9.99	3336.06	2629.27	5.83	6.67
	multi	11.44	12.73	2558.18	2092.37	10.83	9.17
50	**single**	43.21	39.91	3600.52	3569.27	34.17	35
	multi	36.13	32.79	3579.62	3281.97	25.83	31.67
70	**single**	59.82	50.46	3600.57	3600.72	38.33	37.5
	multi	62.95	57.32	3606.94	3472.67	40	40.83

For each fixed parameter of the model (see Table 2), we test if there exist a statistical significant difference for the values obtained by Gurobi for each combination of the parameters values. Since these values are not normally distributed, the Mann-Whitney U rank test (see McKnight and Najab (2010) for more information) is used as a test of difference in location between distributions. We assume a significance level of 5%. Tables 7, 8, 9, 10, 11 and 12 reports the alternative hypothesis that is accepted together with the p-values for the significant tests obtained with this test for **% Gap**, **Runtime**, **% Gap_math**, **Runtime_math**, **% Gap_build**, **Runtime_build**, respectively. The reader should note that the results described in the following are made for those instances in which a feasible solution has been founded.

An example that illustrates how these tables can be interpreted is exposed in the following example. On the one hand, by observing the second column of the first row of Table 7 it is possible to conclude

Table 4: Results depending of matheuristic initialisation

		% Gap		Runtime		% Unsolved		% Gap_math		Runtime_math		% Gap_build		Runtime_build	
	Initialisation	without	with	without	with	without	with	without	with	without	with	without	with	without	with
$\|\mathcal{N}\|$	Approach														
10	single	0	0	99.31	93.83	0	0	-	5.13	-	0.09	-	4.92	-	2.79
	multi	0	0	47.74	75.82	0	0	-	5.13	-	0.36	-	4.57	-	2.03
20	single	1.45	0.45	1612.39	1539.32	0	0	-	3.34	-	0.84	-	3.06	-	120.88
	multi	0.63	0.1	926.85	958.64	0	0	-	3.4	-	3.87	-	3.18	-	58.52
30	single	6.59	14.64	3058.48	2917.56	12.5	0	-	1.82	-	21.34	-	1.64	-	524.23
	multi	1.6	20.49	2378.5	2270.2	20	0	-	1.89	-	18.8	-	1.71	-	404.84
50	single	26.28	46.28	3600.41	3578.19	69.17	0	-	0.48	-	147.04	-	0.48	-	270.19
	multi	16.6	42.15	3437.89	3450.96	57.5	0	-	0.74	-	114.56	-	0.73	-	307.44
70	single	29.82	61.22	3600.64	3600.65	75.83	0	-	0.14	-	240.77	-	0.14	-	270.72
	multi	28.14	66.29	3618.04	3509.31	80.83	0	-	0.07	-	277.42	-	0.07	-	1056.43

Table 5: Results depending of the percentage of barriers considered

		% Gap				Runtime				% Unfound			
	$\lceil \%\|\mathcal{B}\| \rceil$	10%	20%	50%	100%	10%	20%	50%	100%	10%	20%	50%	100%
$\|\mathcal{N}\|$	Approach												
10	single	0	0	0	0	4.27	140.45	191.94	49.62	0	0	0	0
	multi	0	0	0	0	2.48	70.34	151.07	23.24	0	0	0	0
20	single	0	0.08	3.25	0.48	216.82	1161.68	2667.93	2256.99	0	0	0	0
	multi	0	0	1.28	0.18	79.45	327.37	1966.59	1397.58	0	0	0	0
30	single	1.76	7.72	38.35	2.57	1997.93	3144.75	3548.36	3530.7	0	0	25	0
	multi	0.07	4.57	62.11	0.66	967.18	2490.23	3573.27	3208.04	0	1.67	38.33	0
50	single	31.26	62.19	77.14	18.8	3538.99	3600.39	3600.5	3600.63	41.67	46.67	50	0
	multi	15.65	49.42	73.83	20.67	3096.6	3601.08	3606.63	3611.11	23.33	41.67	50	0
70	single	51.07	74.48	90.39	29.38	3600.55	3600.73	3600.64	3600.67	50	50	50	1.67
	multi	55.45	67.62	89.1	41.69	3321.47	3533.57	3613.92	3620.2	46.67	50	50	15

Table 6: Results depending of the k selected

		% Gap			Runtime			% Unfound		
	k	1	$\lceil 10\%\|\mathcal{N}\| \rceil$	$\lceil 25\%\|\mathcal{N}\| \rceil$	1	$\lceil 10\%\|\mathcal{N}\| \rceil$	$\lceil 25\%\|\mathcal{N}\| \rceil$	1	$\lceil 10\%\|\mathcal{N}\| \rceil$	$\lceil 25\%\|\mathcal{N}\| \rceil$
$\|\mathcal{N}\|$	Approach									
10	single	0	0	0	40.73	41.03	207.96	0	0	0
	multi	0	0	0	16.42	15.36	153.57	0	0	0
20	single	0.92	1.19	0.75	1920.76	1806.85	999.95	0	0	0
	multi	0.78	0.21	0.11	1041.49	1050.52	736.23	0	0	0
30	single	9.63	17.34	6.27	3242.46	2958.16	2782.9	6.25	11.25	1.25
	multi	12.31	14.12	9.87	2634	2290.9	2046.29	8.75	11.25	10
50	single	53.69	47.85	22.71	3553.27	3600.27	3600.65	31.25	37.5	35
	multi	37.19	42.18	24.16	3200.56	3585	3547.27	25	31.25	30
70	single	61.86	63.45	40.2	3600.73	3600.53	3600.66	37.5	38.75	37.5
	multi	56.82	67.81	55.99	3493.04	3593.03	3556.74	40	41.25	40

Figure 6: Boxplots for % Gap, Runtime and % Unfound for instances with different ω_L (left side) and initialisation (right side)

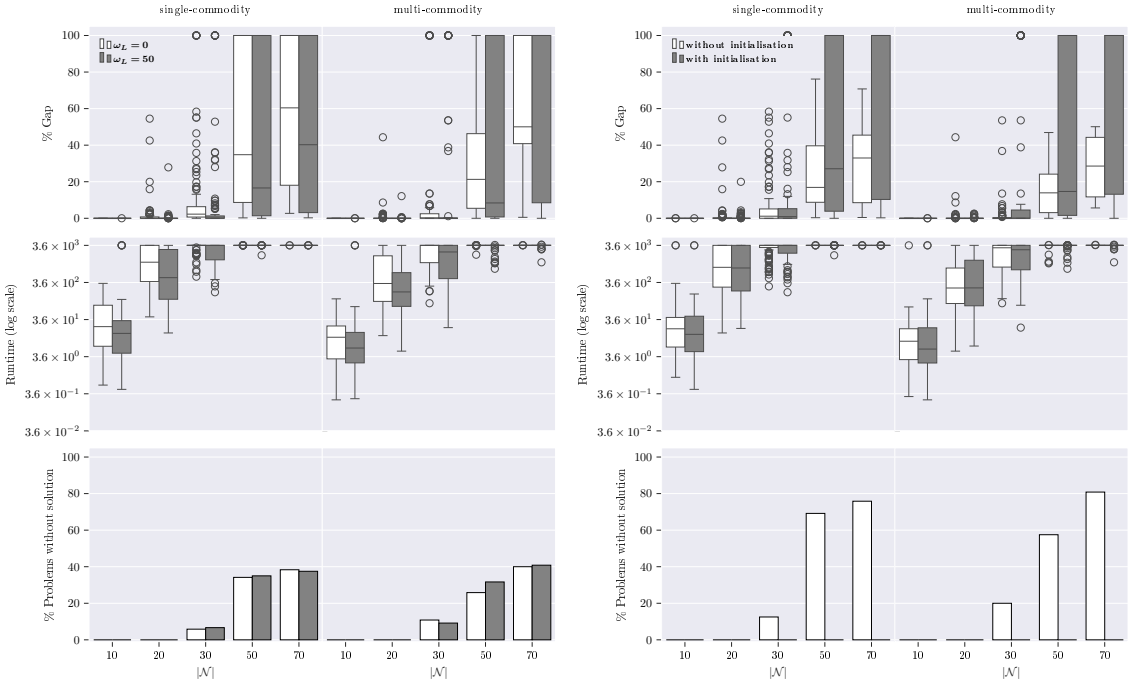
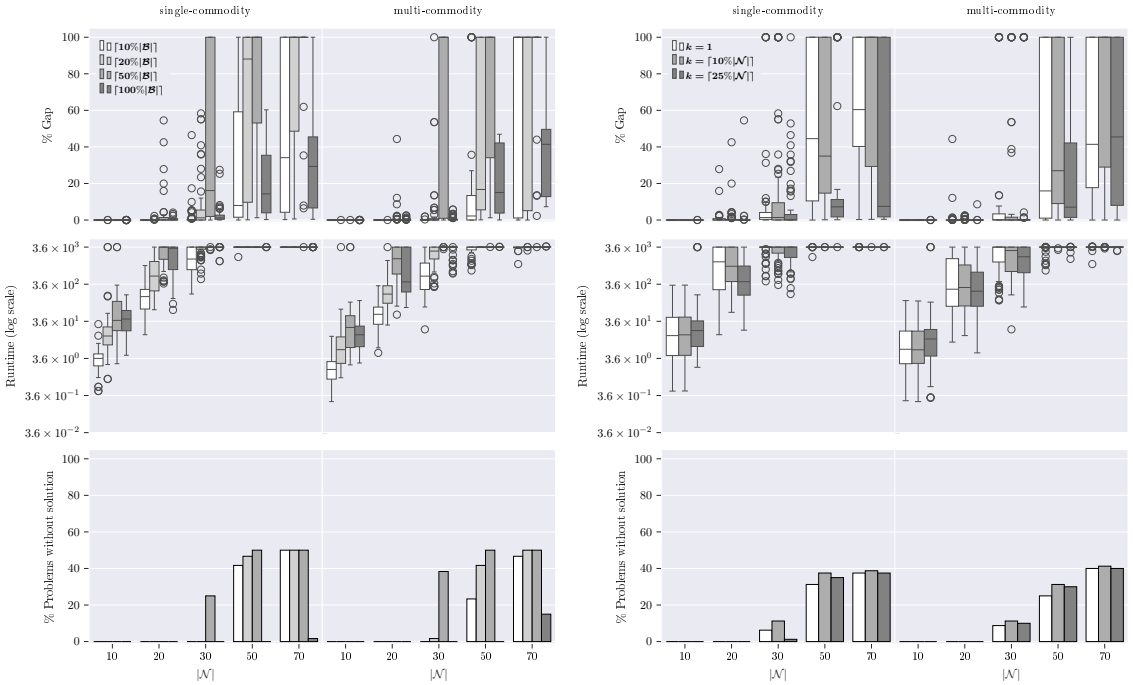


Figure 7: Boxplots for % Gap, Runtime and % Unfound for instances with different $\%|\mathcal{B}|$ (left side) and k (right side)



that the gap for instances solved by using the single-commodity approach is significant higher (>) for instances with $\omega_L = 0$ with respect to those with $\omega_L = 50$. On the other hand, we can say that the solver reports a significant lower gap for instances with $\lceil 20\%|\mathcal{B}| \rceil$ with respect to instances with $\lceil 50\%|\mathcal{B}| \rceil$, as shown in the sixth column of the row one (<). However, the previous column of the same row concludes that there is not a significant difference in gap terms between the $\lceil 10\%|\mathcal{B}| \rceil$ and the $\lceil 20\%|\mathcal{B}| \rceil$. The same

interpretation can be done for each metric represented in each table.

Analysing the results in Table 7, we can conclude that for most of the configurations, the gap reported for the single-commodity approach is higher than the one for the multi-commodity. In addition, the instances that does not take into account the link distances reports more gap that those weighting the link distance. This behaviour can be justified by the difficulty of dealing with the bounds that McCormick's envelope arises. Furthermore, the higher is the number of barriers in the problem the higher is the final gap reported by the solver. The most relevant fact is that the gap is lower even when it is compared with the convex case. It is easily derived from the relationship between the size of the visibility graph and the number of variables involved in the problem. However, considering the initialisation of the solver or changing the number of selected neighbourhoods does not produce any difference for most of the cases. Similar results can be derived in terms of the runtime spent by the solver to close the gap. However, it seems that the runtime spent by the solver when it is initialised by the matheuristic is higher that the model without initialisation for the nonconvex case, the single-commodity approach, $\omega_L = 0$ and the number of selected neighbourhoods is high.

Table 7: Table of p-values for gap difference

		Approach	ω_L	Initialisation	$\lceil \%\|\mathcal{B}\| \rceil$						k		
		single-multi	0-50	without-with	10%-20%	10%-50%	10%-100%	20%-50%	20%-100%	50%-100%	1-10%	1-25%	10%-25%
Approach	single		> (0.0)	< (0.0)	< (0.02)	< (0.0)	< (0.0)	< (0.0)	< (0.049)	> (0.009)	−−	> (0.0)	> (0.007)
	multi		> (0.008)	< (0.0)	< (0.001)	< (0.0)	< (0.0)	< (0.0)	< (0.001)	> (0.009)	−−	−−	−−
ω_L	0	> (0.023)		< (0.0)	−−	< (0.0)	< (0.0)	< (0.0)	< (0.007)	> (0.008)	−−	> (0.004)	> (0.032)
	50	−−		< (0.0)	< (0.0)	< (0.0)	< (0.0)	< (0.0)	< (0.005)	> (0.031)	−−	−−	−−
Initialisation	with	−−	> (0.011)		< (0.0)	< (0.0)	< (0.0)	< (0.0)	−−	> (0.0)	−−	> (0.029)	> (0.017)
	without	> (0.013)	> (0.0)		−−	< (0.0)	< (0.0)	< (0.001)	< (0.0)	< (0.0)	−−	> (0.016)	−−
$\lceil \%\|\mathcal{B}\| \rceil$	10%	> (0.03)	> (0.0)	< (0.0)							−−	−−	−−
	20%	−−	−−	< (0.0)							−−	−−	−−
	50%	−−	−−	< (0.0)							−−	−−	−−
	100%	−−	> (0.0)	−−							−−	> (0.0)	> (0.0)
k	1	> (0.004)	> (0.0)	< (0.0)	< (0.006)	< (0.0)	< (0.0)	< (0.0)	< (0.041)	> (0.036)			
	10%	−−	> (0.031)	< (0.0)	< (0.022)	< (0.0)	< (0.0)	< (0.0)	< (0.021)	> (0.023)			
	25%	−−	> (0.033)	< (0.0)	< (0.045)	< (0.0)	< (0.002)	< (0.0)	< (0.042)	> (0.01)			

Table 8: Table of p-values for runtime difference

		Approach	ω_L	Initialisation	$\lceil \%\|\mathcal{B}\| \rceil$						k		
		single-multi	0-50	without-with	10%-20%	10%-50%	10%-100%	20%-50%	20%-100%	50%-100%	1-10%	1-25%	10%-25%
Approach	single		> (0.005)	< (0.017)	< (0.001)	< (0.0)	< (0.0)	< (0.014)	< (0.0)	< (0.0)	> (0.034)	−−	−−
	multi		> (0.021)	−−	< (0.003)	< (0.0)	< (0.0)	−−	< (0.0)	< (0.0)	−−	−−	−−
ω_L	0	−−		< (0.035)	< (0.005)	< (0.0)	< (0.0)	< (0.02)	< (0.0)	< (0.0)	−−	−−	−−
	50	−−		−−	< (0.0)	< (0.0)	< (0.0)	< (0.046)	< (0.0)	< (0.0)	−−	−−	−−
Initialisation	with	−−	> (0.006)		< (0.003)	< (0.0)	< (0.0)	< (0.046)	< (0.0)	< (0.0)	−−	−−	−−
	without	−−	> (0.009)		< (0.0)	< (0.0)	< (0.0)	< (0.017)	< (0.0)	< (0.0)	−−	−−	−−
$\lceil \%\|\mathcal{B}\| \rceil$	10%	> (0.025)	> (0.002)	< (0.005)							−−	−−	−−
	20%	> (0.027)	> (0.011)	< (0.013)							−−	−−	−−
	50%	−−	> (0.011)	< (0.029)							−−	−−	−−
	100%	< (0.0)	−−	−−							−−	−−	−−
k	1	−−	> (0.006)	−−	< (0.014)	< (0.001)	< (0.0)	−−	< (0.0)	< (0.0)			
	10%	−−	> (0.031)	−−	< (0.007)	< (0.002)	< (0.0)	−−	< (0.0)	< (0.0)			
	25%	−−	−−	< (0.003)	< (0.005)	< (0.0)	< (0.0)	< (0.021)	< (0.0)	< (0.0)			

Table 9: Table of p-values for matheuristic gap difference

		Approach	ω_L			$\lceil\%\lvert\mathcal{B}\rvert\rceil$				k		
		single-multi	0-50	10%-20%	10%-50%	10%-100%	20%-50%	20%-100%	50%-100%	1-10%	1-25%	10%-25%
Approach	multi		$>$ (0.0)	$>$ (0.018)	$>$ (0.0)	--	$>$ (0.003)	--	$<$ (0.0)	$<$ (0.023)	$<$ (0.0)	$<$ (0.0)
	single		$>$ (0.0)	--	$>$ (0.0)	--	$>$ (0.016)	--	$<$ (0.0)	--	$<$ (0.0)	$<$ (0.0)
ω_L	0	--		--	$>$ (0.0)	--	$>$ (0.009)	$<$ (0.023)	$<$ (0.0)	$<$ (0.003)	$<$ (0.0)	$<$ (0.0)
	50	--		$>$ (0.028)	$>$ (0.0)	$>$ (0.017)	$>$ (0.005)	--	$<$ (0.003)	--	$<$ (0.0)	$<$ (0.0)
$\lceil\%\lvert\mathcal{B}\rvert\rceil$	10%	--	$>$ (0.0)							--	$<$ (0.0)	$<$ (0.003)
	20%	--	$>$ (0.0)							--	$<$ (0.0)	$<$ (0.004)
	50%	--	$>$ (0.012)							--	$<$ (0.002)	$<$ (0.005)
	100%	--	$>$ (0.0)							--	$<$ (0.0)	$<$ (0.0)
k	1	--	$>$ (0.002)	$>$ (0.034)	$>$ (0.0)	--	--	--	$<$ (0.004)			
	10%	--	$>$ (0.0)	--	$>$ (0.0)	--	$>$ (0.008)	--	$<$ (0.001)			
	25%	--	$>$ (0.0)	--	$>$ (0.001)	--	$>$ (0.017)	--	$<$ (0.0)			

Table 10: Table of p-values for matheuristic runtime difference

		Approach	ω_L			$\lceil\%\lvert\mathcal{B}\rvert\rceil$				k		
		single-multi	0-50	10%-20%	10%-50%	10%-100%	20%-50%	20%-100%	50%-100%	1-10%	1-25%	10%-25%
Approach	single		--	--	--	$<$ (0.0)	--	$<$ (0.003)	$<$ (0.014)	$<$ (0.0)	$<$ (0.0)	$>$ (0.023)
	multi		--	$<$ (0.0)	$<$ (0.0)	$<$ (0.0)	$<$ (0.0)	$<$ (0.0)	$<$ (0.001)	--	--	--
ω_L	0	$<$ (0.002)		$<$ (0.016)	$<$ (0.0)	$<$ (0.0)	$<$ (0.016)	$<$ (0.0)	$<$ (0.018)	$<$ (0.0)	$<$ (0.001)	--
	50	$<$ (0.0)		$<$ (0.02)	$<$ (0.0)	$<$ (0.0)	$<$ (0.024)	$<$ (0.0)	$<$ (0.001)	$<$ (0.0)	$<$ (0.006)	--
$\lceil\%\lvert\mathcal{B}\rvert\rceil$	10%	--	--							$<$ (0.007)	$<$ (0.037)	--
	20%	$<$ (0.037)	--							$<$ (0.021)	--	--
	50%	$<$ (0.0)	--							$<$ (0.006)	$<$ (0.044)	--
	100%	$<$ (0.0)	--							$<$ (0.0)	$<$ (0.003)	--
k	1	$<$ (0.0)	--	$<$ (0.022)	$<$ (0.0)	$<$ (0.0)	$<$ (0.028)	$<$ (0.002)	--			
	10%	--	--	--	$<$ (0.004)	$<$ (0.0)	--	$<$ (0.0)	$<$ (0.009)			
	25%	$<$ (0.024)	--	$<$ (0.024)	$<$ (0.0)	$<$ (0.0)	$<$ (0.03)	$<$ (0.0)	$<$ (0.002)			

Table 11: Table of p-values for gap difference of the solution built by the solver

		Approach	ω_L			$\lceil\%\lvert\mathcal{B}\rvert\rceil$				k		
		single-multi	0-50	10%-20%	10%-50%	10%-100%	20%-50%	20%-100%	50%-100%	1-10%	1-25%	10%-25%
Approach	multi		$<$ (0.018)	--	$<$ (0.028)	--	--	--	$>$ (0.048)	--	--	--
	single		--	$<$ (0.03)	$<$ (0.001)	$<$ (0.0)	--	$<$ (0.0)	$<$ (0.0)	--	--	--
ω_L	0	$>$ (0.0)		--	$<$ (0.001)	$<$ (0.0)	$<$ (0.025)	$<$ (0.005)	--	--	$>$ (0.006)	--
	50	--		$<$ (0.029)	$<$ (0.017)	$<$ (0.001)	--	--	--	--	--	--
$\lceil\%\lvert\mathcal{B}\rvert\rceil$	10%	--	--							--	--	--
	20%	--	$<$ (0.038)							--	--	--
	50%	--	--							--	--	--
	100%	$>$ (0.0)	--							--	--	--
k	1	$>$ (0.01)	--	--	$<$ (0.013)	$<$ (0.001)	--	--	--			
	10%	--	--	$<$ (0.025)	$<$ (0.002)	$<$ (0.0)	--	$<$ (0.019)	--			
	25%	$>$ (0.011)	$<$ (0.007)	--	--	--	--	--	--			

Table 12: Table of p-values for runtime difference of the solution built by the solver

		Approach	ω_L	$\lceil \%\|\mathcal{B}\| \rceil$						k		
		single-multi	0-50	10%-20%	10%-50%	10%-100%	20%-50%	20%-100%	50%-100%	1-10%	1-25%	10%-25%
Approach	single		−−	< (0.0)	< (0.0)	< (0.0)	< (0.0)	< (0.02)	> (0.002)	> (0.002)	> (0.0)	−−
	multi		−−	< (0.0)	< (0.0)	< (0.0)	< (0.001)	< (0.0)	−−	−−	> (0.028)	−−
ω_L	0	< (0.0)		< (0.001)	< (0.0)	< (0.0)	< (0.0)	< (0.0)	−−	> (0.007)	> (0.0)	−−
	50	< (0.0)		< (0.0)	< (0.0)	< (0.0)	< (0.0)	< (0.038)	> (0.016)	> (0.027)	> (0.004)	−−
$\lceil \%\|\mathcal{B}\| \rceil$	10%	< (0.014)	−−							−−	> (0.015)	−−
	20%	< (0.002)	< (0.04)							> (0.026)	> (0.007)	−−
	50%	−−	< (0.015)							> (0.049)	> (0.004)	−−
	100%	< (0.0)	−−							−−	> (0.003)	−−
k	1	−−	−−	< (0.002)	< (0.0)	< (0.0)	< (0.009)	−−	−−			
	10%	< (0.0)	−−	< (0.002)	< (0.0)	< (0.0)	< (0.001)	< (0.002)	−−			
	25%	< (0.0)	< (0.021)	< (0.0)	< (0.0)	< (0.0)	< (0.001)	< (0.011)	−−			

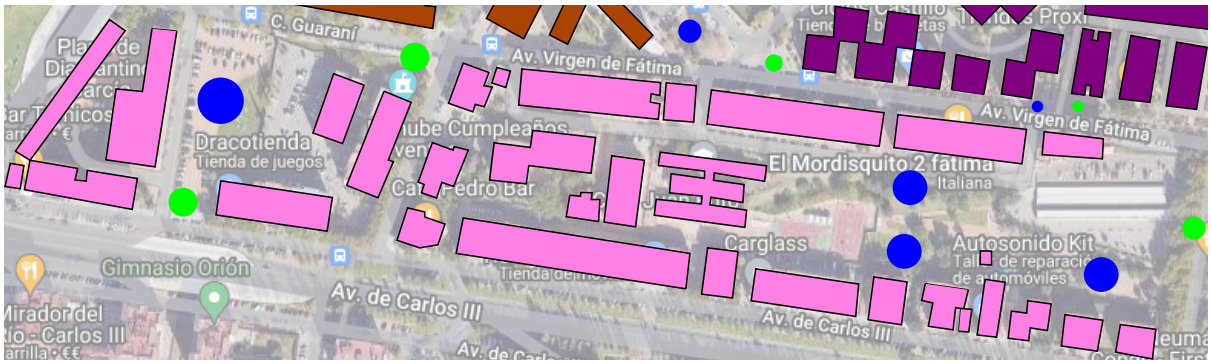
6. Case study

In this section, we describe a realistic application of the problem studied in this paper to perform last-mile delivery. We focus on the problem of allocating k vans from where some drones depart to delivery some products to the population represented. In particular, we consider a neighbourhood in the city of Córdoba called Fátima, that is located in the northeast part of the city as shown in Figure 8. This map is extracted from Google Maps.

Figure 8: Map that represents a part of Fátima, Córdoba. Scale 1:1000

In Figure 9, we plot the sources that are represented by green circles that simulate zones in which vans can park. The targets are also modelled by blue circles that define the zones in which customers are willing to pick up the delivery. The barriers represent buildings that drones cannot cross. To simplify the problem, we assume the drones have enough endurance to go from any source to any target and they cannot ascend to avoid the buildings and, once the number of vans is fixed, we can know the number of required drones assigned with each van.

Figure 9: Buildings that are presented in this part of Fátima



We run the single-commodity model in this scenario, presented in Section 3, for a different number of vans $k \in \{1, 2, 3, 4\}$. It is assumed that the link distance is negligible in comparison with the Euclidean distance, i.e., $\omega_E = 1$ and $\omega_L = 0$.

The optimal value, measured in meters, and runtime spent to obtain the optimal solution obtained for each $k \in \{1, 2, 3, 4\}$ are reported in Table 13.

Table 13: Optimal solution and runtime for each configuration

k	Optimal value	Runtime
1	289.192	1208.53
2	176.669	1417.14
3	124.861	998.25
4	110.001	427.47

The optimal solutions are represented in Figures 10, 11, 12 and 13, respectively. Studying the values in Table 13, we can conclude that increasing the number of vans to be located reduce the total cost spent by the drones to visit all the customers at the cost of including an additional van. Although this work does not consider this "allocation cost", the reader may note that this issue can be dealt straightforwardly by adding an additional term in the objective function described in Section 3.

Figure 10: Solution for the case study ($k = 1$)

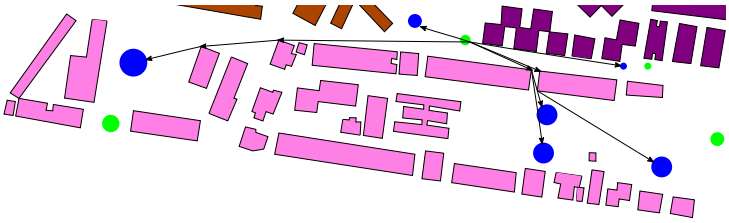


Figure 11: Solution for the case study ($k = 2$)

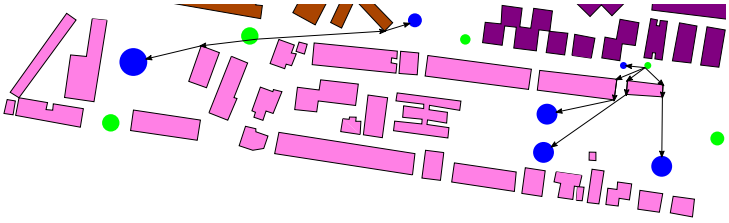


Figure 12: Solution for the case study ($k = 3$)

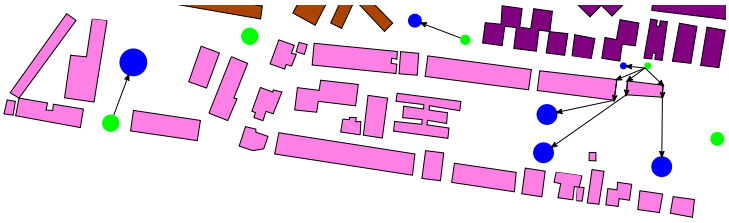
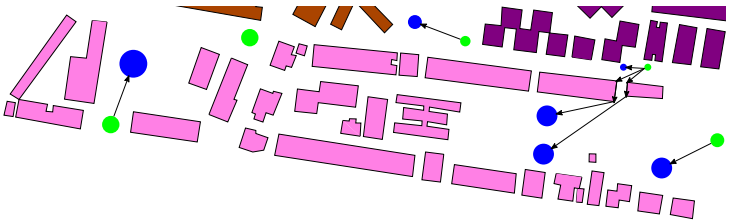


Figure 13: Solution for the case study ($k = 4$)



7. 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. The resulting problems inherit some elements from the k-median problem with neighbourhoods that must be exploited to partially overcome the difficulties of the solution approaches, but in addition require new techniques and algorithms from computational geometry to handle the network design among neighbourhoods and with barriers. The H-KMPN version leads to non-convex mixed-integer problems whereas H-KMPHN results in second-order cone mixed-integer problems. The two problems, beyond its similarity, show deep differences in terms of computational difficulty, as explained in Section 5. However, the proposed mathematical programming approaches permit 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

Arsanjani, J. J., Helbich, M., and de Noronha Vaz, E. (2013). Spatiotemporal simulation of urban growth patterns using agent-based modeling: The case of Tehran. *Cities*, 32:33–42.

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. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press, Cambridge.

Butt, S. E. and Cavalier, T. M. (1996). An efficient algorithm for facility location in the presence of forbidden regions. *European Journal of Operational Research*, 90(1):56–70.

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. (2004). *Facility Location: Applications and Theory*. Springer Science & Business Media.

G. van Rossum (Guido) (1995). Python reference manual.

Gentilini, I., Margot, F., and Shimada, K. (2013). The travelling salesman problem with neighbourhoods: MINLP solution. *Optimization Methods and Software*, 28(2):364–378.

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

Hakimi, S. L. (1965). Optimum Distribution of Switching Centers in a Communication Network and Some Related Graph Theoretic Problems. *Operations Research*, 13(3):462–475.

Hamacher, H. W. and Nickel, S. (1995). Restricted planar location problems and applications. *Naval Research Logistics (NRL)*, 42(6):967–992.

Kariv, O. and Hakimi, S. L. (1979). An Algorithmic Approach to Network Location Problems. II: The p-Medians. *SIAM Journal on Applied Mathematics*, 37(3):539–560.

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.

LaPaugh, A. S. (1980). *Algorithms for Integrated Circuit Layout: An Analytic Approach*. PhD thesis, Massachusetts Institute of Technology.

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.

Lozano-Pérez, T. and Wesley, M. A. (1979). An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 22(10):560–570.

Maheshwari, A., Sack, J.-R., and Djidjev, H. N. (2000). Chapter 12 - Link Distance Problems. In Sack, J. R. and Urrutia, J., editors, *Handbook of Computational Geometry*, pages 519–558. North-Holland, Amsterdam.

Mangiameli, M., Muscato, G., Mussumeci, G., and Milazzo, C. (2013). A GIS application for UAV flight planning. *IFAC Proceedings Volumes*, 46(30):147–151.

McKnight, P. E. and Najab, J. (2010). Mann-Whitney U Test. In *The Corsini Encyclopedia of Psychology*, pages 1–1. John Wiley & Sons, Ltd.

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

Mitchell, J. S. B. and Papadimitriou, C. H. (1991). The weighted region problem: Finding shortest paths through a weighted planar subdivision. *Journal of the ACM*, 38(1):18–73.

Nickel, S. (1995). *Discretization of Planar Location Problems*. Shaker Verlag, Aachen.

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.

QGIS Development Team (2009). QGIS Geographic Information System. Open Source Geospatial Foundation.

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*.

Wangdahl, G. E., Pollock, S. M., and Woodward, J. B. (1974). Minimum-Trajectory Pipe Routing. *Journal of Ship Research*, 18(01):46–49.

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