The Hampered Travelling Salesman Problem with Neighbourhoods*

Justo Puerto[a,1], Carlos Valverde[b,1,*]

[a] *Department of Statistics and Operations Research, University of Seville, Seville, 41012, Spain*
[b] *Department of Statistics and Operations Research, University of Seville, Seville, 41012, Spain*

Abstract

This paper deals with two different route design problems in a continuous space with neighbours and barriers: the shortest path and the travelling salesman problems with neighbours and barriers. Each one of these two elements, neighbours and barriers, makes 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. We provide exact mathematical programming formulations for both problems assuming polygonal barriers and neighbours that are second-order cone (SOC) representable. These hypotheses give rise to mixed integer SOC formulations that we preprocess and strengthen with valid inequalities. The paper also reports computational experiments showing that our exact method can solve instances with 75 neighbourhoods and a range between 125-145 barriers.

Keywords: Routing, Travelling salesman, Networks, Conic programming and interior point methods

1. Introduction

Routing problems are considered classical problems at the core of combinatorial optimisation. Among them, the Travelling Salesman Problem (TSP) is one of its most important representations, and it has been studied extensively in many different forms. The problem is well known to be NP-hard. Its analysis has led in the last decades to methodological advances and new optimisation techniques that have allowed solving real life instances that few years ago were beyond solvable limits. The TSP has been studied in its geometric and pure combinatorial forms because of its algorithmic and practical implications. One very interesting extension arises when we require the tour to visit a set of regions rather than points. This version of the problem is called the TSP with Neighbours (see Arkin and Hassin (1994)) which is APX-hard to approximate even for regions that are (intersecting) line segments. They can represent regions that the drone must reach and where customers are willing to pick up the orders (they can be seen as uniform probability densities) in the delivery industry. Moreover, they can also be used for modelling some areas that must be inspected by the drone (whenever visiting a point of these areas, it suffices to consider them as inspected). On the other hand, the graphic TSP (see T. Moemke and O. Svensson

*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.

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

(22)), where distances are measures by geodesics (shortest paths in the respective metric space), has also attracted the attention of many researchers since it models, in a natural way, TSP routes among forbidden barriers, robot motion planning and automatic navigation in computer games.

For the above mentioned reasons, it is very important to analyse the mathematical implications of barriers to the geometry and computation of routes in a continuous space. Beyond the mathematical relevance due to the intrinsic difficulty of the resulting models (non-convexities), the importance of this topic also comes from its many real-life applications and, more specifically, drone delivery with uniformly distributed demand and drone inspection and surveillance in urban areas with buildings or similar barriers. Most of the classical methods in route design are based on an underlying network structure. Our approach is essentially different since, on top of assuming the existence of barriers, we also assume that the locations of targets to be visited are uniformly distributed in neighbourhoods, giving rise to challenging problems in the continuous space. Nevertheless, the resulting problems still retain geometric elements that must be exploited to partially overcome the difficulties of solution approaches and algorithms in the design of routes among neighbourhoods with barriers.

The well-known Travelling Salesman Problem with Neighbourhoods (TSPN) was introduced by Arkin and Hassin (1994). These authors addressed this problem by proposing heuristic procedures that construct tours and prove that the length of these tours is within a constant factor of the length of an optimal tour. In Gentilini et al. (2013), authors formulate this problem as a non-convex mixed integer non-linear program (MINLP) that yields a convex nonlinear program when the discrete variables are fixed. Moreover, Yuan and Zhang (2017) combine strategically metaheuristics and classical TSP solvers to produce high quality solutions for the TSPN with arbitrary neighbourhoods. In Glock and Meyer (2023), the concept of spatial coverage in routing and path planning is unified and defined, and a categorisation scheme of related problems is introduced. Other classical combinatorial problems, such as the Minimum Spanning Tree (MST) or the Crossing Postman Problem (XPP), which have been extended to deal with neighbourhoods, are developed in Blanco et al. (2017) and Puerto and Valverde (2022), respectively.

In Mennell (2009), the Close-Enough Travelling Salesman problem was introduced. This problem is a particular case of the TSPN where the neighbourhoods are represented by circles. In that dissertation, the author also performed extensive computational experiments by solving instances developed by them. The authors of Coutinho et al. (2016) proposed an exact algorithm based on branch-and-bound and second-order cone programming. Finally, a metaheuristic approach was developed that computes the upper and lower bounds of the optimal solution value. This metaheuristic used a discretization scheme and the Carousel Greedy algorithm to obtain high-quality solutions (see Carrabs et al. (2020) for more details).

As far as we are concerned, the use of barriers in location problems has been widely considered (see Klamroth (2002)) but the corresponding barrier routing problem has attracted less attention in the field of Operations Research. In spite of that, one can find connections with some problems in the area of computational geometry, such as the shortest path problem in polygons and the touring polytopes problem (see Mitchell (2017)), in navigation games competition as, for instance, the Physical Travelling

Salesman Problem (see D. Perez et al. (2014)), or in robot motion planning (see Y. K. Hwang and N. Ahuja (1992) and J. . -P. Laumond et al. (1994)). Different complexity results are known, and many effective heuristic algorithms have been developed for specific classes of problems. However, as far as we know, in all cases targets are points and no exact algorithms are provided. Moreover, at times, these problems can appear as subproblems of some other combinatorial problems, so that one would like to embed them into some mathematical programming formulation. Nevertheless, we are not aware of any mathematical programming formulation for these problems that permits it to be considered as a building block of more complex/integrated problems in real life applications, even without requiring target to be neighbourhoods.

Our goal in this paper is to deal with the TSP with Neighbourhoods and barriers that we call the Hampered Traveling Salesman with Neighbourhoods (H-TSPN). As commented above, each one of these two elements (neighbourhoods and barriers) makes the problem difficult. Thus, mixing both together results in a new problem that, as far as we know, has not been addressed before, but that has a lot of applications in the delivery industry and inspection and surveillance activities, as justified previously. Moreover, it also has implications from the methodological point of view because introduces non-fixed elements in network representation and the issue of obtacles (barriers) in the solution space in the same problem.

Our contribution is to provide exact mathematical programming formulations for the problem assuming polygonal barriers and neighbourhoods that are second-order cone (SOC) representable. These hypotheses give rise to mixed integer SOC formulations that we preprocess and strengthen with valid inequalities. In our way to the formulations we deal with the associated Hampered Shortest Path Problem with Neighbourhoods (H-SPPN) which is used as a building block for the more difficult H-TSPN. Although we prove that the H-SPPN is polynomially solvable, we also give a formulation for this problem that highlights the constraints necessary for H-TSPN. The respective H-TSPN is NP-hard. We give exact formulations using a geodesic shortest-path representation that allows us to solve medium-sized instances of this class of problems. Our computational tests show the difficulty of handling barriers and neighbourhoods together, but also that our formulation is useful for problems with 75 neighbourhoods and a range between 125-145 barriers.

The paper is organised into ~~8~~ 6 sections. The first section is the introduction. In the second section, the problems to be dealt with are described, and the notation followed in the rest of the paper is set. Section 3 develops formulations for the problems defined in the manuscript. These initial formulations are later reformulated using finite dominating sets described in the paper. Section 4 strengthens these formulations by preprocessing variables, developing families of valid inequalities to be added to the formulations and proposing some variable-fixing strategies. A computational experience is reported in Section 5. The paper ends with a section devoted to conclusions and extensions, where some interesting further research connected with the problems addressed in this paper is discussed.

2. Preliminaries and Problems' Description

This section is devoted to setting the hypotheses that describe the framework where we analyse the considered problems. We will deal with two problems in this paper: the Hampered Shortest Path Problem with Neighbourhoods H-SPPN and the Hampered Travelling Salesman Problem with Neighbourhoods H-TSPN. 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 *drone*.

2.1. Assumptions and notation

In both problems considered, we denote by $\mathcal{B}$ the set of line segments that model the barriers and adopt the assumptions listed below.

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

A2 The line segments of $\mathcal{B}$ are open sets, that is, it is possible that the drone visits the endpoints of the segments, but entering into 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 aligned overlapping barriers, we assume that there is only one barrier ~~given by the union of~~ containing both of them.

In this work, we also analyse a special case of the H-TSPN that assumes, besides the previous hypotheses, that:

A4 There is no rectilinear path connecting two neighbourhoods without crossing an obstacle.

This problem is called the Hampered Traveling Salesman Problem with Hidden Neighbourhoods H-TSPHN. The consideration of this more constrained subproblem appears as a natural extension of the H-SPPN. Indeed, in H-SPPN, the shortest path that joins two neighbourhoods is sought by assuming **A4**. Otherwise, it would be trivially solved by finding the standard shortest path between both neighbourhoods since no obstacles would block its use.

Figures 1, 2 and 3 show an example of each of the three problems that are being considered, respectively. Figure 1 shows an instance of the H-SPPN, where the blue neighbourhood represents the source, the green one represents the target, and the red line segments show the barriers that the drone cannot cross. In Figure 2, an instance of the H-TSPHN is shown, where the neighbourhoods are balls and the barriers are, again, the red line segments. Finally, Figure 3 illustrates an example of the H-TSPN, where the orange and blue balls can be joined by a rectilinear path.

To simplify the presentation, we introduce some general notation used throughout the paper.

Figure 1: H-SPPN instance

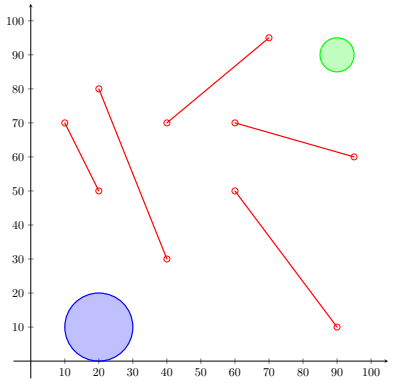


Figure 2: H-TSPHN instance

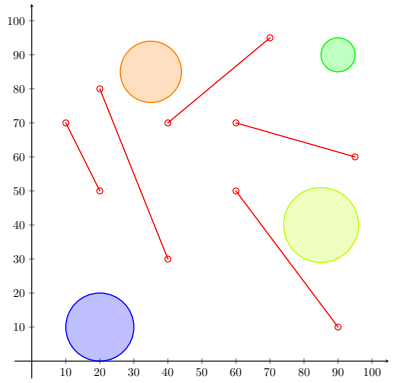
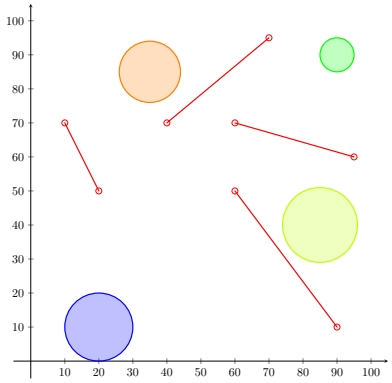


Figure 3: H-TSPN instance



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 $\overline{P^1P^2}$. It can be parameterized as follows:

$$\overline{P^1P^2} = \{P \in \mathbb{R}^2 : P = \lambda P^1 + (1-\lambda)P^2,\ \lambda \in [0,1]\}.$$

- 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**.

Once defined the problems treated in this work, the following subsections describe the construction of the *visibility graphs* that are used to develop some mathematical programming formulations that solve the problem exactly. These objects represent the set of possible rectilinear paths the drone can follow without crossing any barrier. For these problems, the visibility graphs are not fixed. They depend on the points selected in the neighbourhoods, as shown in Figures 4 and 5.

2.2. Description of the Hampered Shortest Path Problem with Neighbourhoods

In H-SPPN, we have a source neighbourhood $N_S \subset \mathbb{R}^2$ and a target neighbourhood $N_T \subset \mathbb{R}^2$, which we assume to be second-order cone-representable sets and a set $\mathcal{B}$ of line segments that play the role of barriers that the drone cannot cross.

The goal of the H-SPPN is to find the best pair of points $(P_S, P_T) \in N_S \times N_T$ in the source and target neighbourhoods that minimise the length of the path that joins both points without crossing any barrier of $\mathcal{B}$ and assuming **A1**-**A4**. To state the model, we define the following sets:

- $V_S = \{P_S\}$. Set composed by the point selected in the source neighbourhood N_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 the barriers in the problem.
- $V_T = \{P_T\}$. Set composed by the point selected in the target neighbourhood N_T.
- $E_S = \{(P_S, P_B^i) : P_B^i \in V_{\mathcal{B}} \text{ and } \overline{P_S P_B^i} \cap B'' = \emptyset, \forall B'' \in \mathcal{B},\ i = 1, 2\}$. Set of edges formed by the lines that join the selected point in the source neighbourhood N_S and each endpoint on 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~~ endnodes of $V_{\mathcal{B}}$ and do not cross any other barrier in $\mathcal{B}$.
- $E_T = \{(P_B^i, P_T) : P_B^i \in V_{\mathcal{B}} \text{ and } \overline{P_B^i P_T} \cap B'' = \emptyset, \forall B'' \in \mathcal{B},\ i = 1, 2\}$. Set of edges formed by the line segments that join the selected point in the target neighbourhood N_T and every endpoint on the barriers that do not cross any other barrier in $\mathcal{B}$.

The above sets allow us to define the visibility graph $G_{\text{SPP}} = (V_{\text{SPP}}, E_{\text{SPP}})$ induced by barriers and neighbourhoods, where $V_{\text{SPP}} = V_S \cup V_{\mathcal{B}} \cup V_T$ and $E_{\text{SPP}} = E_S \cup E_{\mathcal{B}} \cup E_T$.

Figures 4 and 5 show how the visibility graph G_{SPP} is generated for an instance of the H-SPPN. The blue dashed line segments represent the edges of E_S, the green dashed lines, the edges of E_T and the red dashed lines represent the edges of $E_{\mathcal{B}}$.

Figure 4: Generation of the visibility graph G_{SPP}. Case 1

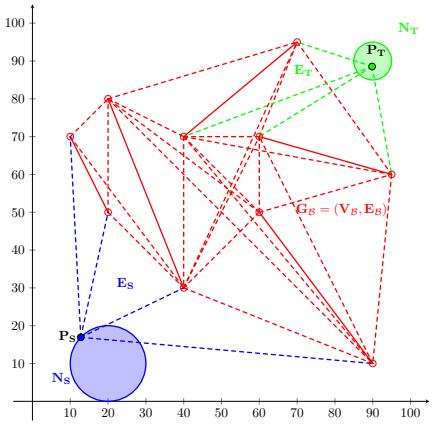
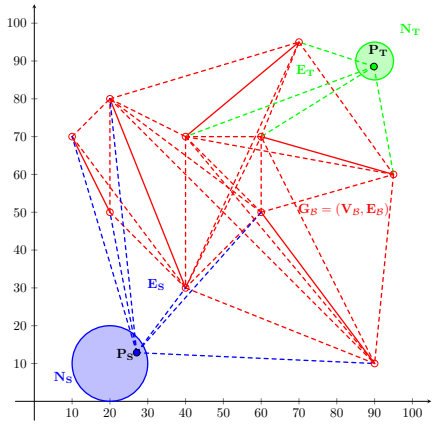


Figure 5: Generation of the visibility graph G_{SPP}. Case 2



A special case that can be highlighted occurs when the neighbourhoods, N_S and N_T, are 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. This idea is later exploited in Subsection 3.3.1.

2.3. Description of the Hampered Travelling Salesman Problem with Hidden Neighbourhoods

The H-TSPHN is an extension of the H-SPPN where the neighbourhood set $\mathcal{N}$ is considered to play the role of source and target in the H-SPPN and, moreover, a set of given targets must be visited. The aim of the H-TSPHN is to find the shortest tour that visits each neighbourhood $N \in \mathcal{N}$ exactly once without crossing any barrier $B \in \mathcal{B}$ and assuming again **A1**-**A4**.

To present our formulation for the H-TSPHN, the graph induced by the endpoints of the barriers and the neighbourhoods is different from the previous one for the H-SPPN. For its description, we introduce the following sets:

- $V_{\mathcal{N}} = \{P_N : N \in \mathcal{N}\}$. Set of points selected in the neighbourhoods $\mathcal{N}$ to be visited.
- $V_{\mathcal{B}} = \{P_B^1, P_B^2 : B = \overline{P_B^1 P_B^2} \in \mathcal{B}\}$. Set of vertices of the barriers of the problem.
- $E_{\mathcal{N}} = \{(P_N, P_B^i) : P_N \in V_{\mathcal{N}}, P_B^i \in V_{\mathcal{B}} \text{ and } \overline{P_N P_B^i} \cap B'' = \emptyset, \forall B'' \in \mathcal{B}, i = 1, 2\}$. Set of edges formed by line segments that join each point selected in the neighbourhoods of $\mathcal{N}$ with each endpoint on the barriers and do not cross any 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 line segments that join two vertices of $V_{\mathcal{B}}$ and do not cross any barrier in $\mathcal{B}$.

Following the same idea as before, we consider the visibility graph $G_{\text{TSPH}} = (V_{\text{TSPH}}, E_{\text{TSPH}})$ induced by barriers and neighbourhoods, where $V_{\text{TSPH}} = V_{\mathcal{N}} \cup V_{\mathcal{B}}$ and $E_{\text{TSPH}} = E_{\mathcal{N}} \cup E_{\mathcal{B}}$.

2.4. Description of the Hampered Travelling Salesman Problem with Neighbourhoods

For the general case, barriers are not required to completely separate all neighbourhoods, i.e., when moving from one neighbourhood to another, sometimes it is possible to follow a straight line without crossing any barrier. The main difference lies in the description of the edges of the graph induced by the neighbourhoods and endpoints of the barriers.

By following the same approach that before, the sets that describe the graph in this case are:

- $V_{\mathcal{N}} = \{P_N : N \in \mathcal{N}\}$. Set of points in the neighbourhoods $\mathcal{N}$ that must be visited.
- $V_{\mathcal{B}} = \{P_B^1, P_B^2 : B = \overline{P_B^1 P_B^2} \in \mathcal{B}\}$. Set of vertices of the barriers of the problem.
- $V_{\text{TSP}} = V_{\mathcal{N}} \cup V_{\mathcal{B}}$.
- $E_{\text{TSP}} = \{(P, P') : P, P' \in V_{TSP} \text{ and } \overline{PP'} \cap B'' = \emptyset, \forall B'' \in \mathcal{B}\}$. Set of edges formed by the line segments that join every pair of points in V_{TSP} that do not cross any barrier.

3. MINLP Formulations

This section proposes a mixed integer nonlinear programming formulation for the problems described in Section 2. First of all, we present the conic programming representation of the neighbourhoods and distance. Then, we set the constraints that check if a segment is included in the set of edges E_X

with $X \in \{SPP, TSPH, TSP\}$. Finally, the formulations for the H-SPPN, H-TSPHN and H-TSPN are described.

We would like to remark that having a mathematical programming representation for the H-SPPN is very important, even though we will prove in the following that this problem can be solved with a polynomial-time combinatorial algorithm. Indeed, we will need this type of representation to address some NP-hard problems that require computing shortest paths as building blocks, such as, for example, the location problem with barriers and neighbours and the H-TSPN. Therefore, we will start to provide two different mathematical programming formulations for the H-SPPN capable of being embedded in more complex problems.

First, we introduce the decision variables that represent the problem. These are summarised in Table 1.

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

Binary Decision Variables	
Name	**Description**
$\alpha(P\|QQ')$	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 $\overline{QQ'}$ intersect, 0, otherwise.
$\varepsilon(PP')$	1, if the line segment $\overline{PP'}$ does not cross any barrier, 0, otherwise.
$y(PQ)$	1, if the edge (P, Q) is selected in the solution of the model, 0, otherwise.
Continuous Decision Variables	
Name	**Description**
P_N	Coordinates representing the point selected in the neighbourhood N.
$d(PQ)$	Euclidean distance between the points P and Q.
$g(PQ)$	Amount of commodity passing through the edge (P, Q).

3.1. Conic programming constraints in the models

Let $\|\cdot\|_2$ denote the standard Euclidean norm derived from the dot product in $\mathbb{R}^n$, i.e., $\|u\|_2 = (u^t u)^{1/2}$. The second-order cone (or Lorentz cone) of dimension $k+1$ is defined as:

$$\mathcal{L}^{k+1} = \{(x, y) \in \mathbb{R}^k \times \mathbb{R} : \|x\|_2 \leq y\}.$$

Then, a second-order cone constraint is defined as

$$\|A_i x - b_i\| \leq c_i^t x - d_i \Leftrightarrow (A_i x - b_i, c_i^t x - d_i) \in \mathcal{L}^{k_i+1}, \quad i = 1, \ldots, m,$$

where $A_i \in \mathbb{R}^{k_i \times n}, b_i \in \mathbb{R}^{k_i}, c_i \in \mathbb{R}^n$ and $d_i \in \mathbb{R}$ are the problem parameters.

Second-order cone constraints can be used to represent common convex restrictions. If $A_i \equiv 0$ and $b_i \equiv 0$, for some $i \in 1, \ldots, m$, the corresponding second-order cone constraint is reduced to a linear. On the other hand, if $c_i \equiv 0$ and $d_i \leq 0$, the constraint is reduced to a convex quadratic constraint that can represent ellipsoids or hyperbolic constraints (see Lobo et al. (1998) and Boyd and Vandenberghe (2004) for more information). The advantages of second-order constraints are the fully symmetric duality, heavily utilised by solution algorithms, the existence of polynomial-time interior point methods and its extremely powerful expressive abilities (Nesterov and Nemirovski, 1994).

In the two problems considered, namely H-SPPN and H-TSPN, there exist two second-order cone constraints that model the distance between pairs of points P and Q, as well as the representation of neighbourhoods where the points can be chosen.

For modelling the distance between pairs of points, we introduce the nonnegative 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_{SPP}, E_{TSPH} or E_{TSP}.

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

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

where A_N^i, b_N^i, c_N^i and d_N^i are the constraints parameters, i and nc_N denotes the number of constraints that appear in the block associated with the neighbourhood N.

These type of elements could be extended further to unions of SOC representable sets by introducing binary variables. Each one determines the set where the point chosen to visit the union of these SOC representable sets is located.

Let $\mathcal{U}_N = \{C_N^1, \ldots, C_N^{m_N}\}$ be the second-order cone representable sets that define the neighbourhood N. Consider the binary variable χ_N^j that is one if $P_N \in C_N^j$, and zero otherwise. Therefore, for each $N \in \mathcal{N}$,

$$P_N \in \mathcal{U}_N \Longleftrightarrow \begin{cases} \|A_N^{ij} P_N + b_N^{ij}\| & \leq & (c_N^{ij})^T P_N + d_N^{ij} + U_N^j (1 - \chi_N^j), \quad i = 1, \ldots, nc_N, \quad j = 1, \ldots, m_N, \\ \sum_{j=1}^{m_N} \chi_N^j & = & 1, \end{cases} \tag{U-C}$$

where U_N^j is a big M constant on the maximal distance between two points in the union of sets. The reader may observe that (N-C) can be replaced by (U-C) without altering the validity of the formulations. However, for the sake of simplicity, only the SOC-representable sets are considered in the paper.

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

The goal of this subsection is to provide a test to check whether, given two arbitrary vertices $P, Q \in V_X$, the edge $(P, Q) \in E_X$, with $X \in \{SPP, TSPH, TSP\}$, that is, whether the line segment $\overline{PQ}$ does not intersect 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*

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

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

Let $P, Q \in V_X$, where V_X denotes the set of vertices V_{SPP}, V_{TSPH} or V_{TSP}. It is essential to model the conditions of the Remark 1 by using binary variables that check the sign of determinants, the equality of signs, and the disjunctive condition, since these determinants depend on the location of P and Q.

To model the sign of each determinant in Remark 1, we introduce the binary variable α, which assumes the value one if the determinant is positive and zero, otherwise. Note that the case where the determinants are null does not need to be considered because the segments are located in a general position.

It is possible to represent the sign condition by including the following constraints:

$$\begin{aligned}
\left[1-\alpha(P|P_B^1P_B^2)\right] L(P|P_B^1P_B^2) \le \det(P|P_B^1P_B^2) \le U(P|P_B^1P_B^2)\,\alpha(P|P_B^1P_B^2), \qquad (\alpha\text{-C})\\
\left[1-\alpha(Q|P_B^1P_B^2)\right] L(Q|P_B^1P_B^2) \le \det(Q|P_B^1P_B^2) \le U(Q|P_B^1P_B^2)\,\alpha(Q|P_B^1P_B^2),\\
\left[1-\alpha(P_B^1|PQ)\right] L(P_B^1|PQ) \le \det(P_B^1|PQ) \le U(P_B^1|PQ)\,\alpha(P_B^1|PQ),\\
\left[1-\alpha(P_B^2|PQ)\right] L(P_B^2|PQ) \le \det(P_B^2|PQ) \le U(P_B^2|PQ)\,\alpha(P_B^2|PQ),
\end{aligned}$$

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

Now, we check whether the pairs of determinants

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

have the same sign, we introduce the binary variable β, that is one if the corresponding pair has the same sign, and zero otherwise.

Therefore, the correct value of the variable β can be enforced by the following constraint of the variables α.

$$\begin{aligned}
\beta(PQ|P_B^1P_B^2) &= \alpha(P|P_B^1P_B^2)\alpha(Q|P_B^1P_B^2) + \left[1-\alpha(P|P_B^1P_B^2)\right]\left[1-\alpha(Q|P_B^1P_B^2)\right],\\
\beta(P_B^1P_B^2|PQ) &= \alpha(P_B^1|PQ)\alpha(P_B^2|PQ) + \left[1-\alpha(P_B^1|PQ)\right]\left[1-\alpha(P_B^2|PQ)\right].
\end{aligned}$$

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

$$\begin{aligned}\beta(PQ|P_B^1P_B^2) &= 2\gamma(PQ|P_B^1P_B^2) - \alpha(P|P_B^1P_B^2) - \alpha(Q|P_B^1P_B^2) + 1, \\ \beta(P_B^1P_B^2|PQ) &= 2\gamma(P_B^1P_B^2|PQ) - \alpha(P_B^1|PQ) - \alpha(P_B^2|PQ) + 1,\end{aligned} \tag{β-C}$$

We observe that γ can be linearised using the following constraints:

$$\begin{aligned}&\gamma(PQ|P_B^1P_B^2) \leq \alpha(P|P_B^1P_B^2), && \gamma(P_B^1P_B^2|PQ) \leq \alpha(P_B^1|PQ), \\ &\gamma(PQ|P_B^1P_B^2) \leq \alpha(Q|P_B^1P_B^2), && \gamma(P_B^1P_B^2|PQ) \leq \alpha(P_B^2|PQ), \\ &\gamma(PQ|P_B^1P_B^2) \geq \alpha(P|P_B^1P_B^2) + \alpha(Q|P_B^1P_B^2) - 1, && \gamma(P_B^1P_B^2|PQ) \geq \alpha(P_B^1|PQ) + \alpha(P_B^2|PQ) - 1.\end{aligned} \tag{γ-C}$$

Later, we need to check whether there exists any coincidence of the sign of determinants, so we define the binary variable δ, which is one if the segments do not intersect and zero, otherwise. This condition can be modelled by using the following disjunctive constraints:

$$\frac{1}{2}\left[\beta(PQ|P_B^1P_B^2) + \beta(P_B^1P_B^2|PQ)\right] \leq \delta(PQ|P_B^1P_B^2) \leq \beta(PQ|P_B^1P_B^2) + \beta(P_B^1P_B^2|PQ). \tag{δ-C}$$

Indeed, the above constraints 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 the pairs of determinants is the same, then the second constraint is zero and δ is null.

Finally, we need to check that

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

Hence, if we denote by $\varepsilon(PQ)$ the binary variable that is one if the previous condition is satisfied for all $B'' \in \mathcal{B}$ and zero otherwise, this variable can be represented by means of the following inequalities:

$$\left[\sum_{B''\in\mathcal{B}} \delta(PQ|P_{B''}^1P_{B''}^2) - |\mathcal{B}|\right] + 1 \leq \varepsilon(PQ) \leq \frac{1}{|\mathcal{B}|}\sum_{B''\in\mathcal{B}} \delta(PQ|P_{B''}^1P_{B''}^2). \tag{ε-C}$$

If there is, 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 forces ε to be zero because the right hand side is fractional and the first inequality is nonpositive. Nevertheless, if no barrier intersects the segment $\overline{PQ}$, then ε is equal to one, because the left-hand side of the first inequality is one and the right-hand side of the second inequality is also one.

Based on the description above, we can identify the set of actual edges of the graph G using the variables ε as follows:

$$E_X = \{(P,Q) : P,Q \in V_X, \varepsilon(PQ) = 1, P \neq Q\}, \quad X \in \{SPP, TSPH, TSP\}.$$

This representation of E_X with $X \in \{SPP, TSPH, TSP\}$ will be exploited in the formulations described in the following subsections.

It is interesting to note that $E_{\mathcal{B}}$ is a fixed set whose edges can be computed using Remark 1. Then, the variables ε can be prefixed in advance. However, the edges in $E_X \setminus E_{\mathcal{B}}$ depend on the points selected in the neighbourhoods, as explained before.

3.3. A formulation for the H-SPPN

The idea of the formulation of the H-SPPN is to extend the classical formulation of the Shortest Path Problem by taking into account the description of the graph G_{SPP} in Subsection 2.2.

The formulation describes the path that the drone can follow by taking into account the edges of the induced graph. Let $P, Q \in V_{\text{SPP}}$ and let $y(PQ)$ be the binary variable that is one if the drone goes from P to Q. Then, the inequalities

$$y(PQ) \leq \varepsilon(PQ), \tag{y-C}$$

assure that the drone can go from P to Q only if the segment $\overline{PQ}$ does not cross any barrier.

Taking into account the constraints explained in the subsections above, the following MINLP formulation is valid for H-SPPN.

$$\text{minimize} \quad \sum_{(P,Q)\in E_{\text{SPP}}} d(PQ)y(PQ) \tag{H-SPPN}$$

$$\text{subject to} \quad \sum_{\{Q:(P,Q)\in E_{\text{SPP}}\}} y(PQ) - \sum_{\{Q:(Q,P)\in E_{\text{SPP}}\}} y(QP) = \begin{cases} 1, & \text{if } P \in V_S, \\ 0, & \text{if } P \in V_{\mathcal{B}}, \\ -1, & \text{if } P \in V_T. \end{cases},$$

$$(\alpha\text{-C}), (\beta\text{-C}), (\gamma\text{-C}), (\delta\text{-C}) \quad \forall P, Q \in V_{\text{SPP}}, \quad \forall P_B^1, P_B^2 \in V_{\mathcal{B}},$$

$$(\varepsilon\text{-C}), (\text{y-C}), (\text{d-C}) \quad \forall P, Q \in V_{\text{SPP}},$$

$$(\text{N-C}) \quad \forall P \in V_S \cup V_T.$$

The objective function minimises the length of the path followed by the drone at the edges of the induced graph G_{SPP}. The first constraints are the flow conservation constraints that ensure the path is connected, the second constraints represent the sets E_S and E_T and the third ones state that the selected points must be in their respective neighbourhoods.

3.3.1. Reformulating the H-SPPN

The formulation for the H-SPPN presented above can be simplified taking into account the following observation.

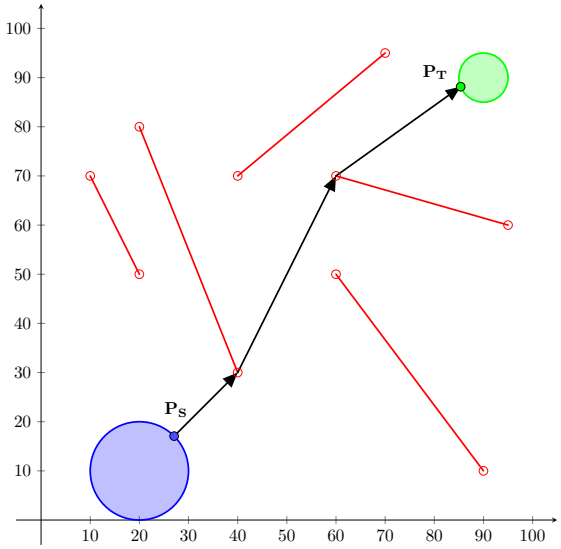
Proposition 1. *There exist two finite dominating sets, N_S^* and N_T^*, of possible candidates to be starting points in N_S and terminal points in N_T, respectively. Moreover,*

$$N_S^* = \{P_S(P_B^i) : P_S(P_B^i) = \underset{P_S\in N_S}{\arg\min} \|P_S - P_B^i\|, \, (P_S, P_B^i) \in E_S \text{ and } P_B^i \in V_{\mathcal{B}}\},$$

$$N_T^* = \{P_T(P_B^i) : P_T(P_B^i) = \underset{P_T\in N_T}{\arg\min} \|P_B^i - P_T\|, \, (P_B^i, P_T) \in E_T \text{ and } P_B^i \in V_{\mathcal{B}}\}.$$

Proof. Note that the points chosen in N_S and N_T in an optimal solution for H-SPPN must be those that give the minimum distances to the points of the first and last barriers visited in any optimal solution,

Figure 6: Optimal solution for this instance of the H-SPPN



respectively. Therefore, N_S^* and N_T^* must contain, at most, the points in the sets

$$\{P_S(P_B^i) : P_S(P_B^i) = \arg\min_{P_S \in N_S} \|P_S - P_B^i\|, (P_S, P_B^i) \in E_S \text{ and } P_B^i \in V_{\mathcal{B}}\},$$

$$\{P_T(P_B^i) : P_T(P_B^i) = \arg\min_{P_T \in N_T} \|P_B^i - P_T\|, (P_B^i, P_T) \in E_T \text{ and } P_B^i \in V_{\mathcal{B}}\},$$

as claimed. □

Therefore, we can compute, *'a priori'*, the sets N_X^*, $X \in \{S, T\}$. For each $P_B^i \in V_{\mathcal{B}}$, the corresponding point in the source neighbourhood $P_X(P_B^i)$ is computed by solving the following convex problem:

$$\begin{aligned} P_X(P_B^i) = \quad & \arg\min_{P_X \in N_X} \quad d(P_X P_B^i) && (N_X^*) \\ & \text{subject to} \quad (\alpha\text{-C}), (\beta\text{-C}), (\gamma\text{-C}), (\delta\text{-C}), \quad \forall P_B^1, P_B^2 \in V_{\mathcal{B}}, \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (\varepsilon\text{-C}), \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \varepsilon(P_X P_B^i) = 1, \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \|P_X - P_B^i\| \leq d(P_X P_B^i). \end{aligned}$$

Then, the visibility graph is constructed by means of these dominating sets. This graph, required to address the new formulation, uses the previous graph G_{SPP} ~~and then~~ in which we increase the new edges that connect the initial and terminal points $P_S \in N_S$ and $P_T \in N_T$, respectively, with the corresponding points in N_S^* and N_T^*. Moreover, we also augment the edges connecting the points in the dominating sets with the extreme points of the segments where the minimal distances are attained. The reader may note that since the optimisation model minimises the overall distance, in an optimal solution P_S must belong to N_S^* and P_T to N_T^*.

Therefore, the following sets are needed to build the new visibility graph induced by the dominating sets.

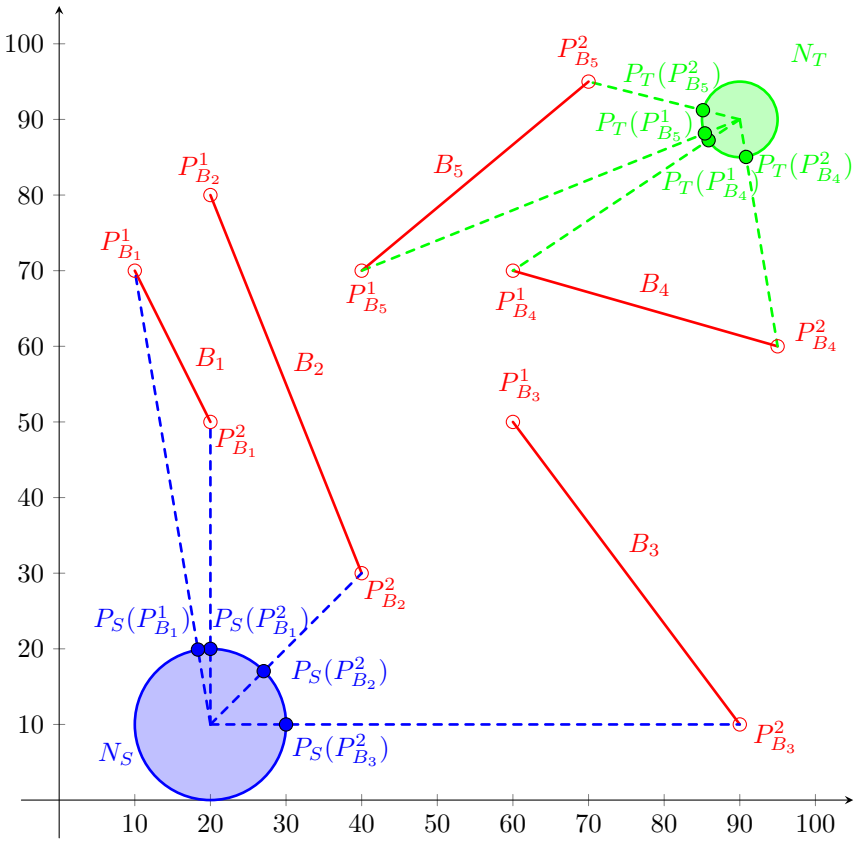
- $V_S^* = N_S^*$. Dominating set of points associated with the neighbourhood N_S.

- $V_T^* = N_T^*$. Dominating set of points associated with the neighbourhood N_T.
- $E_S^{\text{int}} = \{(P_S, P_S(P_B^i)) : P_S \in V_S \text{ and } P_B^i \in V_{\mathcal{B}}\}$. Set of edges that join the selected point $P_S \in N_S$ with each point $P_S(P_B^i)$ in the dominating set N_S^*.
- $E_S^{\text{ext}} = \{(P_S(P_B^i), P_B^i) : P_B^i \in V_{\mathcal{B}}\}$. Set of edges joining the point $P_S(P_B^i)$ with its respective P_B^i in $V_{\mathcal{B}}$.
- $E_S^* = E_S^{\text{int}} \cup E_S^{\text{ext}}$.
- $E_T^{\text{int}} = \{(P_T(P_B^i), P_T) : P_B^i \in V_{\mathcal{B}} \text{ and } P_T \in V_T\}$. Set of edges that join each point $P_S(P_B^i)$ in the dominating set N_S^* with the point selected $P_T \in N_T$.
- $E_T^{\text{ext}} = \{(P_B^i, P_T(P_B^i)) : P_B^i \in V_{\mathcal{B}}$. Set of edges joining the point P_B^i in $V_{\mathcal{B}}$ with its respective $P_T(P_B^i)$.
- $E_T^* = E_T^{\text{int}} \cup E_T^{\text{ext}}$.

We define the graph $G_{\text{SPP}}^* = (V_{\text{SPP}}^*, E_{\text{SPP}}^*)$, where $V_{\text{SPP}}^* = V_S^* \cup V_{\text{SPP}} \cup V_T^*$ and $E_{\text{SPP}}^* = E_S^* \cup E_{\mathcal{B}} \cup E_T^*$.

The major advantage of this approach is that, once the sets N_S^* and N_T^* are calculated beforehand, the entire graph G_{SPP}^* is fixed because the incident edges can be computed for each point $P_S(P_B^i)$ and $P_T(P_B^i)$, $P_B^i \in V_{\mathcal{B}}$, separately. Figure 7 shows how the dominating sets N_S^* and N_T^* are calculated for an instance of the H-SPPN.

Figure 7: Illustration of the dominating sets associated with an instance of the H-SPPN.



Defining again the variables d and y for the edges in E^*_{SPP}, the new formulation for the H-SPPN can be expressed as the following simplified form:

$$\text{minimize} \quad \sum_{(P,Q)\in E^*_{\text{SPP}}} d(PQ)y(PQ) \tag{H-SPPN*}$$

$$\text{subject to} \quad \sum_{\{Q:(P,Q)\in E^*_{\text{SPP}}\}} y(PQ) - \sum_{\{Q:(Q,P)\in E^*_{\text{SPP}}\}} y(QP) = \begin{cases} 1, & \text{if } P \in V_S, \\ 0, & \text{if } P \in V^*_S \cup V_{\mathcal{B}} \cup V^*_T, \\ -1, & \text{if } P \in V_T. \end{cases},$$

$$\text{(d-C)} \quad \forall P, Q \in V^*_{\text{SPP}},$$

$$\text{(N-C)} \quad \forall P \in V_S \cup V_T.$$

Proposition 2. *Let $n = |\mathcal{B}|$. The* H-SPPN *can be solved in polynomial time $O(n^3)$.*

Proof. The proof follows using the finite dominating sets N^*_S and N^*_T. First, it is clear that the cardinality of these finite dominating sets is $O(n)$ since there is one point in each set for each vertex of a segment in $\mathcal{B}$. Next, we recall that given a set of polygonal obstacles with $O(n)$ vertices, the shortest linear size map from any given point can be computed in $O(n \log n)$ time. Once you have that map, finding the shortest path to any other point in the region is done in $O(n)$ time (Mitchell, 2017). Therefore, the H-SPPN can be solved as follows.

For each point $P_S(P^i_B) \in N^*_S$ we compute the shortest path map of linear size and solve the problem of the shortest path with respect to all $O(n)$ points in N^*_T. In general, this operation takes $O(n \log n + n^2)$. The same operation has to be repeated for all $O(n)$ points in N^*_S. Therefore, the overall complexity to solve H-SPPNis $O(n^3)$.

□

3.4. A formulation for the H-TSPHN

The rationale of the formulation for the H-TSPHN is to consider the variant called Steiner TSP (STSP) (see Letchford et al. (2013)), where some nodes in $V_{\mathcal{B}}$ do not have to be visited, but if necessary, they can be visited more than once.

It is well known that it is possible to convert any instance of the STSP into an instance of the standard TSP, by computing the shortest paths between every pair of required nodes, when these nodes are fixed. However, in our problem, since the points in the neighbourhoods are not fixed, the H-SPPN cannot be used directly to derive an optimal solution for the H-TSPHN, although it may produce a good approximation to generate lower bounds for the H-TSPHN that allows to solve larger instances.

Our approach relies on adjusting a single-commodity flow formulation to ensure connectivity. We can assume that the neighbourhood N_1 is required and that the drone departs from that depot (assuming that it is N_1) with $|\mathcal{N}| - 1$ units of commodity. The idea is that the model must deliver one unit of commodity to each of the required neighbourhoods. Then, for each edge $(P, Q) \in E_{\text{TSPH}}$, we define the following variables:

- $y(PQ)$, binary variable equal to one if the drone goes from P to Q.

- $g(PQ)$, non-negative continuous variable that represents the amount of commodity passing through the edge (P, Q).

Hence, we can adjust the single-commodity flow formulation to the induced graph G_{TSPH} as follows:

$$\text{minimize} \quad \sum_{(P,Q)\in E_{\text{TSPH}}} d(PQ)y(PQ) \tag{H-TSPHN}$$

$$\text{subject to} \quad \sum_{\{Q:(P_N,Q)\in E_{\mathcal{N}}\}} y(P_NQ) \geq 1, \qquad \forall P_N \in V_{\mathcal{N}},$$

$$\sum_{\{Q:(P,Q)\in E_{\text{TSPH}}\}} y(PQ) = \sum_{\{Q:(Q,P)\in E_{\text{TSPH}}\}} y(QP), \quad \forall P \in V_{\text{TSPH}},$$

$$\sum_{\{Q:(Q,P_N)\in E_{\mathcal{N}}\}} g(QP_N) - \sum_{\{Q:(P_N,Q)\in E_{\mathcal{N}}\}} g(P_NQ) = 1, \qquad \forall P_N \in V_{\mathcal{N}} \setminus \{P_{N_1}\},$$

$$\sum_{\{Q:(Q,P_B^i)\in E_{\text{TSPH}}\}} g(QP_B^i) - \sum_{\{Q:(P_B^i,Q)\in E_{\text{TSPH}}\}} g(P_B^iQ) = 0, \qquad \forall P_B^i \in V_{\mathcal{B}},$$

$$g(PQ) \leq (|\mathcal{N}|-1)y(PQ), \qquad \forall (P,Q) \in E_{\text{TSPH}},$$

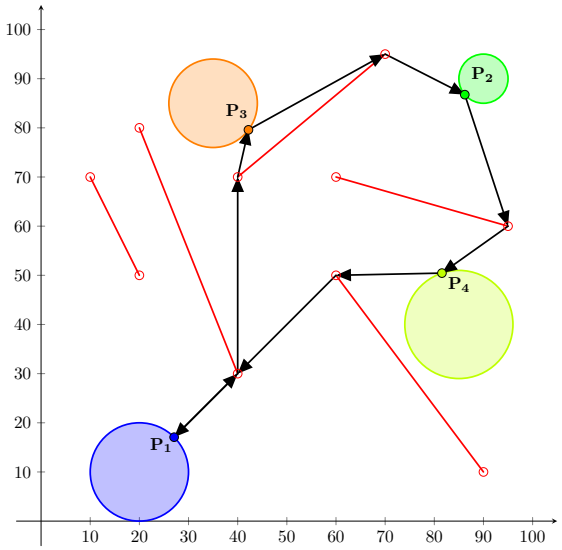
$$(\alpha\text{-C}), (\beta\text{-C}), (\gamma\text{-C}), (\delta\text{-C}) \quad \forall P, Q \in V_{\text{TSPH}}, \quad \forall P_B^1, P_B^2 \in V_{\mathcal{B}},$$

$$(\varepsilon\text{-C}), (\text{y-C}), (\text{d-C}) \quad \forall P, Q \in V_{\text{TSPH}},$$

$$(\text{N-C}) \quad \forall P_N \in V_{\mathcal{N}}.$$

The first group of constraints imposes that the drone departs from each neighbourhood. The second block of constraints is the flow-conservation constraints. The third inequalities ensure that one unit of commodity is delivered to each of the required neighbourhood. The fourth ensures that the fictitious nodes at the end of the barriers do not consume commodity. Finally, the last inequalities enforce that some commodity goes throughout an edge only if this edge is used in the tour. The inequalities (α-C), (β-C), (γ-C), (δ-C), (ε-C), (y-C), (d-C), (N-C) require the variables of the problem to be well defined.

Figure 8: Solution for the instance of the H-TSPHN



Proposition 3. *The* H-TSPHN *is NP-complete.*

Note that, once a point is fixed in each neighbourhood, the problem that results in the induced graph G_{TSPH} is the Steiner TSP (STSP), which is NP-complete.

3.4.1. Reformulating the H-TSPHN

The idea of computing a dominating set that represents each of the neighbourhoods in the H-SPPN can be adopted to reformulate the H-TSPHN in the same way. The dominating sets are stated in the next proposition.

Proposition 4. *Given a neighbourhood $N \in \mathcal{N}$, there exists a finite dominating set, N^* of possible candidates to be in N. Moreover,*

$$N^* = \{P_N(P_B^i, P_{B'}^j) : P_N(P_B^i, P_{B'}^j) = \underset{P_N \in N}{\arg\min} \|P_B^i - P_N\| + \|P_N - P_{B'}^j\| \text{ and } (P_B^i, P_N), (P_N, P_{B'}^j) \in E_{\mathcal{N}}\}.$$

Proof. The way a drone visits a neighbourhood N is

$$P_B^i \longrightarrow P_N \longrightarrow P_{B'}^j,$$

for some points P_B^i, $P_{B'}^j \in V_{\mathcal{B}}$, since, by **A4**, there does not exist a rectilinear path that joins any pair of neighbourhoods. Therefore, the points chosen in an optimal solution for H-TSPHN must be those that produce the minimum distances to the points of the barriers visited previously and next visited in the optimal solution. Therefore, N^* must be composed, at most, of the points in the set

$$\{P_N(P_B^i, P_{B'}^j) = \underset{P_N \in N}{\arg\min} \|P_B^i - P_N\| + \|P_N - P_{B'}^j\| : (P_B^i, P_N), (P_N, P_{B'}^j) \in E_{\mathcal{N}}\},$$

which completes the proof. □

Again, we can compute the respective dominating set N^* of a neighbourhood $N \in \mathcal{N}$ by solving a convex problem for each pair of barrier endpoints:

$$N^* = \{P_N(P_B^i, P_{B'}^j) : P_N(P_B^i, P_{B'}^j) = \underset{P_N \in N}{\arg\min} \|P_B^i - P_N\| + \|P_N - P_{B'}^j\|, \varepsilon(P_B^i P_N) = 1 \text{ and } \varepsilon(P_N P_{B'}^j) = 1\}.$$

The graph induced by the precomputed dominating sets is described in terms of the following sets:

- $V_{\mathcal{N}}^* = \{N^* : N \in \mathcal{N}\}$. Union of the dominating sets associated with each neighbourhood.
- $E_N^{\text{int}} = \{(P_N(P_B^i, P_{B'}^j), P_N), (P_N, P_N(P_B^i, P_{B'}^j)) : P_B^i \in V_{\mathcal{B}}, P_N \in N \text{ and } P_{B'}^j \in V_{\mathcal{B}}\}$. Set of edges joining the point selected $P_N \in N$ with each point $P_N(P_B^i, P_{B'}^j)$ in the dominating set N^*.
- $E_N^{\text{ext}} = \{(P_B^i, P_N(P_B^i, P_{B'}^j)), (P_N(P_B^i, P_{B'}^j), P_{B'}^j) : P_B^i \in V_{\mathcal{B}} \text{ and } P_{B'}^j \in V_{\mathcal{B}}\}$. Set of edges joining the point $P_N(P_B^i, P_{B'}^j)$ with its respective P_B^i and $P_{B'}^j$ in $V_{\mathcal{B}}$.
- $E_N^* = E_N^{\text{int}} \cup E_N^{\text{ext}}$. Set of edges associated with the neighbourhood N.
- $E_{\mathcal{N}}^* = \{E_N^* : N \in \mathcal{N}\}$. Union of the edges of every neighbourhood.

We define the graph $G^*_{\text{TSPH}} = (V^*_{\text{TSPH}}, E^*_{\text{TSPH}})$, where $V^*_{\text{TSPH}} = V_{\text{TSPH}} \cup V^*_{\mathcal{N}}$ and $E^*_{\text{TSPH}} = E^*_{\mathcal{N}} \cup E_{\mathcal{B}}$.

Again, the sets N^* for each $N \in \mathcal{N}$ can be computed in advance so that the whole graph G^*_{TSPH} is fixed. The new formulation for the H-TSPHN can be represented as the next simplified program:

$$\text{minimize} \quad \sum_{(P,Q)\in E^*_{\text{TSPH}}} d(PQ)y(PQ) \qquad \text{(H-TSPHN}^*\text{)}$$

$$\begin{aligned}
\text{subject to} \quad & \sum_{\{Q:(P_N,Q)\in E^{\text{int}}_N\}} y(P_NQ) \geq 1, && \forall P_N \in V_{\mathcal{N}},\\
& \sum_{\{Q:(P,Q)\in E^*_{\text{TSPH}}\}} y(PQ) = \sum_{\{Q:(Q,P)\in E^*_{\text{TSPH}}\}} y(QP), && \forall P \in V^*_{\text{TSPH}},\\
& \sum_{\{Q:(Q,P_N)\in E^{\text{int}}_N\}} g(QP_N) - \sum_{\{Q:(P_N,Q)\in E^{\text{int}}_N\}} g(P_NQ) = 1, && \forall P_N \in V_{\mathcal{N}} \setminus \{P_{N_1}\},\\
& \sum_{\{Q:(Q,P)\in E^*_{\text{TSPH}}\}} g(QP) - \sum_{\{Q:(P,Q)\in E^*_{\text{TSPH}}\}} g(PQ) = 0, && \forall P \in V_{\mathcal{B}} \cup V^*_{\mathcal{N}},\\
& g(PQ) \leq (|\mathcal{N}|-1)y(PQ), && \forall (P,Q) \in E^*_{\text{TSPH}},\\
& \text{(d-C)} \quad \forall P,Q \in V^*_{\text{TSPH}},\\
& \text{(N-C)} \quad \forall P_N \in V_{\mathcal{N}}.
\end{aligned}$$

The reader may note that the maximum number of dominating points associated with an instance $(\mathcal{N}, \mathcal{B})$ of the H-TSPHN is $\frac{1}{2}|\mathcal{N}||\mathcal{B}|(|\mathcal{B}|-1)$. Therefore, to obtain all sets N^* for each $N \in \mathcal{N}$, it is required to solve, at most, $\frac{1}{2}|\mathcal{N}||\mathcal{B}|(|\mathcal{B}|-1)$ convex programs which may be computationally expensive but polynomial.

3.5. Relaxing the assumptions of the problem: The H-TSPN

In this subsection, we analyse the differences between H-TSPHN and H-TSPN, where it is possible to move from one neighbourhood to another one without crossing any barrier. The main difference lies in the description of the edges of the graph induced by the neighbourhoods and the endpoints of the barriers.

By taking the same approach as before, the sets that describe the graph in the new case are $V_{\mathcal{N}}$, $V_{\mathcal{B}}$, V_{TSP} and E_{TSP}, as described in Subsection 2.3.

$$\text{minimize} \quad \sum_{(P,Q)\in E_{\text{TSP}}} d(PQ)y(PQ) \tag{H-TSPN}$$

$$\begin{aligned}
\text{subject to} \quad & \sum_{\{Q:(P_N,Q)\in E_{\mathcal{N}}\}} y(P_NQ) \geq 1, && \forall P_N \in V_{\mathcal{N}},\\
& \sum_{\{Q:(P,Q)\in E_{\text{TSP}}\}} y(PQ) = \sum_{\{Q:(Q,P)\in E_{\text{TSP}}\}} y(QP), && \forall P \in V_{\text{TSP}},\\
& \sum_{\{Q:(Q,P_N)\in E_{\mathcal{N}}\}} g(QP_N) - \sum_{\{Q:(P_N,Q)\in E_{\mathcal{N}}\}} g(P_NQ) = 1, && \forall P_N \in V_{\mathcal{N}} \setminus \{P_{N_1}\},\\
& \sum_{\{Q:(Q,P_B^i)\in E_{\text{TSP}}\}} g(QP_B^i) - \sum_{\{Q:(P_B^i,Q)\in E_{\text{TSP}}\}} g(P_B^iQ) = 0, && \forall P_B^i \in V_{\mathcal{B}},\\
& g(PQ) \leq (|\mathcal{N}|-1)y(PQ), && \forall (P,Q) \in E_{\text{TSP}},\\
& (\alpha\text{-C}), (\beta\text{-C}), (\gamma\text{-C}), (\delta\text{-C}) && \forall P,Q \in V_{\text{TSP}}, \quad \forall P_B^1, P_B^2 \in V_{\mathcal{B}},\\
& (\varepsilon\text{-C}), (\text{y-C}), (\text{d-C}) && \forall P,Q \in V_{\text{TSP}},\\
& (\text{N-C}) && \forall P_N \in V_{\mathcal{N}}.
\end{aligned}$$

The formulation described above is analogous to those detailed in (H-TSPHN). The difference between the set of edges in the H-TSPN with respect to the graph in H-TSPHN is that, in the former case, the edges that join each pair of neighbourhoods must be considered. This fact leads to including the product of continuous variables in the constraints α of the model that represent the determinants that determine whether the segment joining the two variable points in the neighbourhoods crosses any barrier or not. These products make the problem a non-convex, quadratically constrained program.

4. Strengthening the formulations

4.1. Preprocessing

In this subsection, a pre-processing result is proposed that allows one to fix some variables. It is based on analysing the relative position between the neighbourhoods and the barriers. Specifically, we present a sufficient condition that ensures that there are some barriers whose endpoints cannot be incident at the edges of $E_{\mathcal{N}}$ so that it is not necessary to include them in $E_{\mathcal{N}}$.

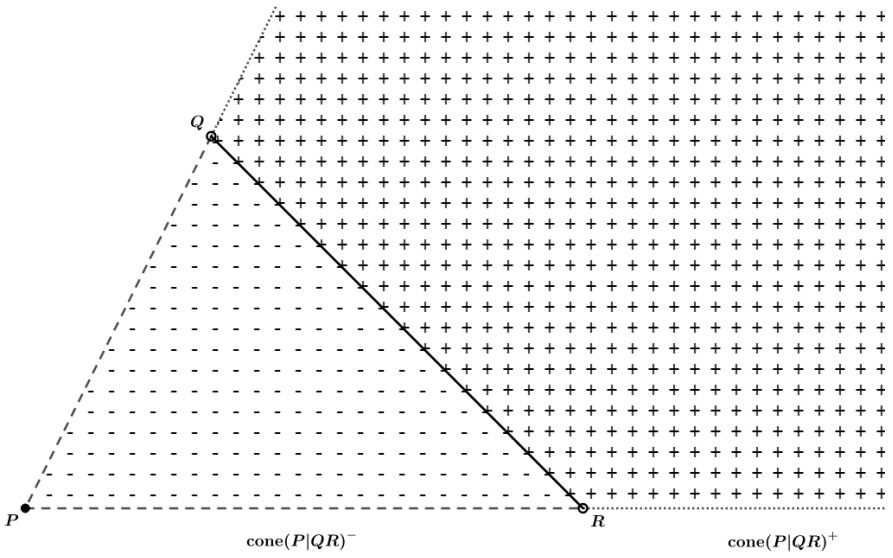
Let us denote

$$\begin{aligned}
\text{cone}(P|QR) &:= \{\mu_1\overrightarrow{PQ} + \mu_2\overrightarrow{PR} : \mu_1, \mu_2 \geq 0\},\\
\text{cone}(P|QR)^- &:= \{\mu_1\overrightarrow{PQ} + \mu_2\overrightarrow{PR} : \mu_1, \mu_2 \geq 0,\ \mu_1 + \mu_2 \leq 1\},\\
\text{cone}(P|QR)^+ &:= \{\mu_1\overrightarrow{PQ} + \mu_2\overrightarrow{PR} : \mu_1, \mu_2 \geq 0,\ \mu_1 + \mu_2 \geq 1\}.
\end{aligned}$$

Note that $\text{cone}(P|QR)$ is the union of $\text{cone}(P|QR)^-$ and $\text{cone}(P|QR)^+$. It is also important to note that $\text{cone}(P|QR)^+$ represents the subset of points P' in the plane whose segments $\overline{PP'}$ cross the barrier $\overline{QR}$ (see Figure 9), i.e.

$$\text{cone}(P|QR)^+ = \{P' : \overline{PP'} \cap \overline{QR} \neq \emptyset\}.$$

Figure 9: Representation of the cone generated by the point P and the line segment $\overline{QR}$.



Let $B = \overline{P_B^1 P_B^2} \in \mathcal{B}$ a barrier. In the following proposition, we give a sufficient condition to not include the edge (P_N, P_B^i) in $E_{\mathcal{N}}$.

Proposition 5. *Let* $B' = \overline{P_{B'}^1 P_{B'}^2} \in \mathcal{B}$ *and* $\text{cone}(P_B^i | P_{B'}^1 P_{B'}^2)^+$ *the conical hull generated by these points. If*

$$N \subset \bigcup_{B' \in \mathcal{B}} \text{cone}(P_B^i | P_{B'}^1 P_{B'}^2)^+,$$

then $(P_N, P_B^i) \notin E_{\mathcal{N}}$.

Proof. If $P_N \in N$, then there exists a $B' \in \mathcal{B}$ such that $P_N \in \text{cone}(P_B^i | P_{B'}^1 P_{B'}^2)^+$. Therefore, $\overline{P_B^i P_N} \cap B' \neq \emptyset$ and $(P_N, P_B^i) \notin E_{\mathcal{N}}$. □

One can easily check computationally the condition of the previous proposition by using the following procedure. First, we deal with the case where neighbourhoods are segments. Let $N = \overline{P_N^1 P_N^2}$ be a line segment and r_N be the straight line that contains the line segment N represented as:

$$r_N : P_N^1 + \lambda \overrightarrow{P_N^1 P_N^2}, \qquad \lambda \in \mathbb{R}.$$

Algorithm 1: Checking computationally whether $(P_N, P_B^i) \notin E_{\mathcal{N}}$ when N is a segment.

Initialization: Let P_B^i be the point of the edge (P_N, P_B^i) to check whether $(P_N, P_B^i) \notin E_{\mathcal{N}}$.

Set $points = \{P_N^1, P_N^2\}$, $lambdas = \{0, 1\}$.

for $B'' \in \mathcal{B}$ **do**

  for $j \in \{1, 2\}$ **do**

    Compute the straight line

$$r(P_B^i, P_{B''}^j) = P_B^i + \mu_{B''}^j \overrightarrow{P_B^i P_{B''}^j},$$

    that contains the points P_B^i and $P_{B''}^j$.

    Intersect $r(P_B^i, P_{B''}^j)$ and r_N at the point $Q_{B''}^j$ and compute $\overline{\mu}_{B''}^j$ such that

$$Q_{B''}^j = P_B^i + \overline{\mu}_{B''}^j \overrightarrow{P_B^i P_{B''}^j}.$$

    if $|\overline{\mu}_{B''}^j| \geq 1$ **then**

      Compute $\lambda_{B''}^j$ such that

$$Q_{B''}^j = P_N^1 + \lambda_{B''}^j \overrightarrow{P_N^1 P_N^2}.$$

      if $\overline{\mu}_{B''}^j \geq 1$ **then**

        Include $\lambda_{B''}^j$ in *lambdas*.

      else

        if $\lambda_{B''}^j \geq 0$ **then**

          Set $\lambda_{B''}^j = M << 0$ and include it in *lambdas*.

        else

          Set $\lambda_{B''}^j = M >> 0$ and include it in *lambdas*.

Order the set *lambdas* in non-decreasing order.

If it is satisfied that

$$\min\{\lambda_{B'}^1, \lambda_{B'}^2\} \leq 0 \leq \max\{\lambda_{B'}^1, \lambda_{B'}^2\}, \quad \text{for some } B' \in \mathcal{B},$$

$$\min\{\lambda_{B'}^1, \lambda_{B'}^2\} \leq 1 \leq \max\{\lambda_{B'}^1, \lambda_{B'}^2\}, \quad \text{for some } B' \in \mathcal{B},$$

$$\min\{\lambda_{B'}^1, \lambda_{B'}^2\} \leq \lambda_{B''}^j \leq \max\{\lambda_{B'}^1, \lambda_{B'}^2\}, \quad \text{for some } B' \in \mathcal{B} \setminus \{B''\}, \quad \forall \lambda_{B''}^j \in lambdas \setminus \{M\},$$

or

$$\min\{\lambda_{B'}^1, \lambda_{B'}^2\} \leq 0, 1 \leq \max\{\lambda_{B'}^1, \lambda_{B'}^2\}, \quad \text{for some } B' \in \mathcal{B},$$

then $(P_N, P_B^i) \notin E_{\mathcal{N}}$.

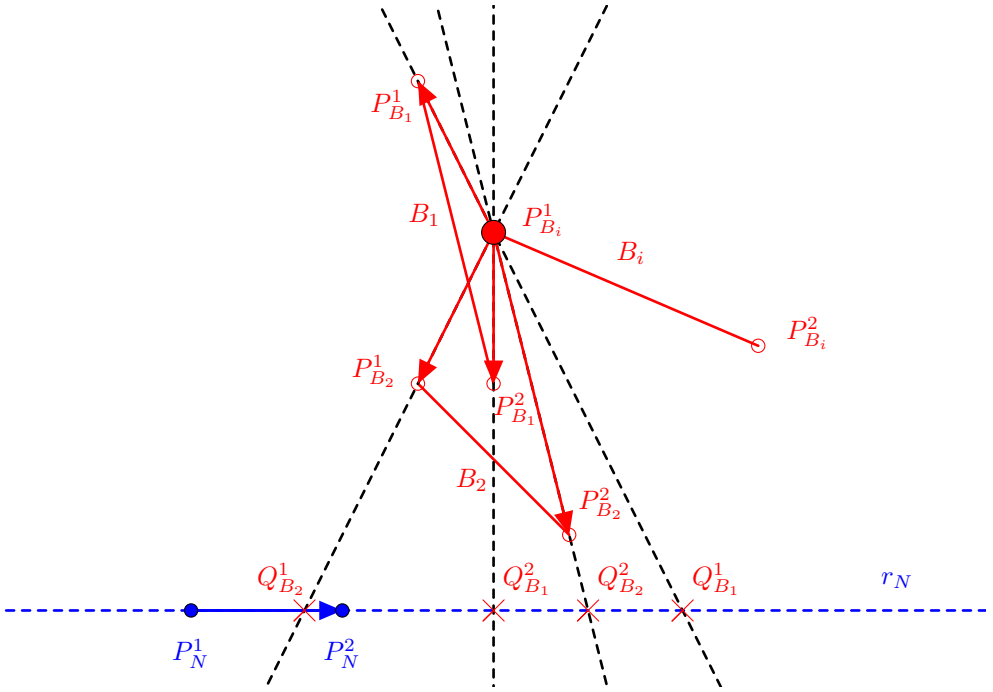
In Figure 10 an example of the Algorithm 1 is described to check if there exists a rectilinear path that joins the solid point P_B^i in the red barrier defined by the extreme points P_B^i and P_B^j with the segment $\overline{P_N^1 P_N^2}$. First, the dashed straight lines $r(P_B^i, P_{B''}^j)$ generated by P_B^i and each point of the barrier are computed. Second, the straight line r_N is determined by P_N^1 and P_N^2. Then, each of the straight lines $r(P_B^i, P_{B''}^j)$ is intersected with r_N obtaining the points Q_1^1, Q_2^1, Q_1^2 and Q_2^2. Each of these points has two

associated parameter values μ and λ with respect to the straight lines $r(P_B^i, P_{B''}^j)$ and r_N, respectively. Finally, the points are ordered in nondecreasing sequence with respect to the λ values. Since

$$M = \lambda_1^1 \leq 0, 1 \leq \lambda_1^2 = 2,$$

the segment $\overline{P_N^1 P_N^2}$ is fully included in cone$(P_B^i | P_1^1 P_1^2)^+$ and, therefore, (P_B^i, P_N) is not included in $E_{\mathcal{N}}$.

Figure 10: Example of the Algorithm 1



	$Q_{B_1}^1$	P_N^1	$Q_{B_2}^1$	P_N^2	$Q_{B_1}^2$	$Q_{B_2}^2$
μ	$-\frac{5}{2}$	$\times$	$\frac{5}{2}$	$\times$	$\frac{5}{2}$	$\frac{5}{4}$
λ	$M << 0$	0	$\frac{3}{4}$	1	2	$\frac{9}{4}$

Note that this algorithm also allows us to decide whether the drone can access a barrier point from any point in the neighbourhood N. It is enough to check in (15) that

$$0 \notin \left[\min\{\lambda_{B'}^1, \lambda_{B'}^2\}, \max\{\lambda_{B'}^1, \lambda_{B'}^2\}\right] \quad \text{and} \quad 1 \notin \left[\min\{\lambda_{B'}^1, \lambda_{B'}^2\}, \max\{\lambda_{B'}^1, \lambda_{B'}^2\}\right], \quad \forall B' \in \mathcal{B}.$$

For the case where N is an ellipse, the same rationale can be followed. The idea is to generate the largest line segment contained in the ellipse and to repeat the procedure in Algorithm 1. Let F_1 and F_2 be the focal points of N.

Algorithm 2: Checking computationally whether $(P_N, P_B^i) \notin E_{\mathcal{N}}$ when N is an ellipse.

Initialization: Let P_B^i be the point whose edge (P_B^i, P_N) is going to check whether $(P_N, P_B^i) \notin E_{\mathcal{N}}$.
Set $points = \{\}$, $lambdas = \{\}$.

1 Compute the straight line $r(F^1, F^2)$.

2 Intersect $r(F^1, F^2)$ and the boundary of N, ∂N, in the points P_N^1 and P_N^2.

3 Include P_N^1 and P_N^2 in *points*.

4 Apply Algorithm 1.

4.2. Valid inequalities

This subsection is devoted to showing some results that adjust the big M constants that appear in the previous formulation, specifically, in constraints (α-C), where modelling of the sign requires computing the lower and upper bounds L and U, respectively. We are going to determine these bounds explicitly for the cases where the neighbourhoods are ellipses and segments.

Let $\overline{P_{B'}^1 P_{B'}^2} = B' \in \mathcal{B}$ be a barrier, and $P_N \in N$. Let $\det(P_N | P_{B'}^1 P_{B'}^2)$ also be the determinant whose value must be bounded. Clearly, the solution of the following problem gives a lower bound of the determinant:

$$\overline{L} = \min_{P_N=(x,y)\in N} F(x,y) := \det(P_N | P_{B'}^1 P_{B'}^2) = \begin{vmatrix} P_{B'_x}^1 - x & P_{B'_x}^2 - x \\ P_{B'_y}^1 - y & P_{B'_y}^2 - y \end{vmatrix}. \tag{L-Problem}$$

4.2.1. Lower and upper bounds when the neighbourhoods are line segments

In this case, the segment whose endpoints are P_N^1 and P_N^2 can be expressed as the following convex set:

$$N = \{(x,y) \in \mathbb{R}^2 : (x,y) = \mu P_N^1 + (1-\mu) P_N^2,\ 0 \le \mu \le 1\}.$$

Since we optimise a linear function in a compact set, we can conclude that the objective function in (L-Problem) achieves its minimum and maximum at the extreme points of N, that is, in P_N^1 and P_N^2.

4.2.2. Lower and upper bounds when the neighbourhoods are ellipses

The next case considered is that when N is an ellipse, that is, N is represented by the following inequality:

$$N = \{(x,y) \in \mathbb{R}^2 : ax^2 + by^2 + cxy + dx + ey + f \le 0\},$$

where a, b, c, d, e, f are coefficients of the ellipse. In extended form, we need to find:

$$\begin{aligned} \text{minimize} \quad & F(x,y) = \begin{vmatrix} P_{B'_x}^1 - x & P_{B'_x}^2 - x \\ P_{B'_y}^1 - y & P_{B'_y}^2 - y \end{vmatrix} = xP_{B'_y}^1 - xP_{B'_y}^2 + yP_{B'_x}^2 - yP_{B'_x}^1 + P_{B'_x}^1 P_{B'_y}^2 - P_{B'_y}^1 P_{B'_x}^2, \\ \text{subject to} \quad & ax^2 + by^2 + cxy + dx + ey + f \le 0. \end{aligned} \tag{L-Ellipse}$$

Since we minimise a linear function in a convex set, we can conclude that the extreme points are located in the frontier, so we can use the Lagrangian function to compute these points.

$$F(x,y;\lambda) = xP^1_{B'_y} - xP^2_{B'_y} + yP^2_{B'_x} - yP^1_{B'_x} + P^1_{B'_x}P^2_{B'_y} - P^1_{B'_y}P^2_{B'_x} + \lambda(ax^2 + by^2 + cxy + dx + ey + f).$$

$$\nabla F(x,y;\lambda) = 0 \Longleftrightarrow \begin{cases} \frac{\partial F}{\partial x} &= P^1_{B'_y} - P^2_{B'_y} + 2ax\lambda + cy\lambda + d\lambda &= 0, \\ \frac{\partial F}{\partial y} &= P^2_{B'_x} - P^1_{B'_x} + 2by\lambda + cx\lambda + e\lambda &= 0, \\ \frac{\partial F}{\partial \lambda} &= ax^2 + by^2 + cxy + dx + ey + f &= 0. \end{cases}$$

From the first two equations, we obtain the following:

$$\lambda = \frac{P^2_{B'_y} - P^1_{B'_y}}{2ax + cy + d} = \frac{P^1_{B'_x} - P^2_{B'_x}}{2by + cx + e}.$$

From this equality, we obtain the following general equation of the straight line:

$$(P^2_{B'_y} - P^1_{B'_y})(2by + cx + e) - (P^1_{B'_x} - P^2_{B'_x})(2ax + cy + d) = 0,$$

$$\left[c(P^2_{B'_y} - P^1_{B'_y}) - 2a(P^1_{B'_x} - P^2_{B'_x})\right]x + \left[2b(P^2_{B'_y} - P^1_{B'_y}) - c(P^1_{B'_x} - P^2_{B'_x})\right]y + \left[e(P^2_{B'_y} - P^1_{B'_y}) - d(P^1_{B'_x} - P^2_{B'_x})\right] = 0,$$

$$\left[(2a,c)\cdot\overrightarrow{P^1_{B'}P^2_{B'}}\right]x + \left[(c,2b)\cdot\overrightarrow{P^1_{B'}P^2_{B'}}\right]y + \left[(d,e)\cdot\overrightarrow{P^1_{B'}P^2_{B'}}\right] = 0,$$

where $\cdot$ denotes the scalar product of two vectors. Solving the quadratic system:

$$\begin{cases} \left[(2a,c)\cdot\overrightarrow{P^1_{B'}P^2_{B'}}\right]x + \left[(c,2b)\cdot\overrightarrow{P^1_{B'}P^2_{B'}}\right]y + \left[(d,e)\cdot\overrightarrow{P^1_{B'}P^2_{B'}}\right] &= 0, \\ ax^2 + by^2 + cxy + dx + ey + f &= 0, \end{cases}$$

they arise two solutions $x^{\pm}$ and $y^{\pm}$ that are evaluated in the objective function to obtain the lowest and highest value, respectively, according to $L(P_N|P^1_{B'}P^2_{B'})$ and $U(P_N|P^1_{B'}P^2_{B'})$, respectively. The reader may note that the same approach can be adopted to obtain the bounds for the rest of the determinants that appear in (α-C).

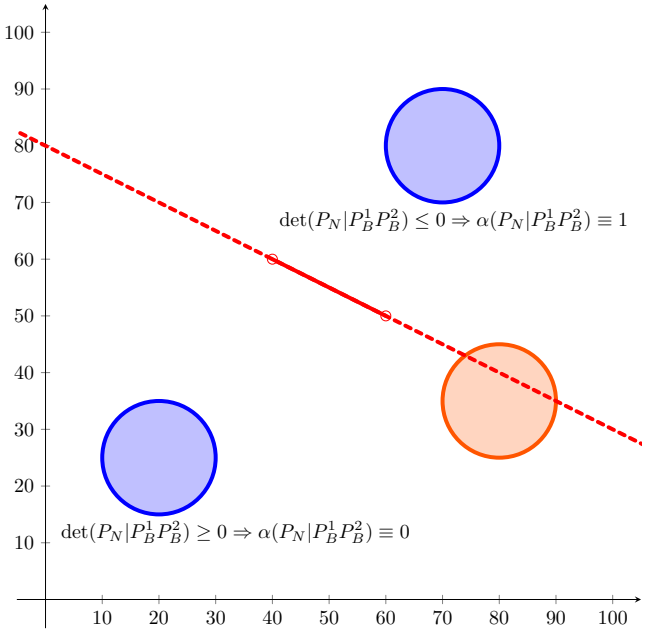
4.2.3. Variable Fixing

In this subsection, the geometry of the problem is exploited to fix the variables. In particular, when a neighbourhood N is in the half-space generated by a barrier B, the sign of the determinant $\det(P_N|P^1_BP^2_B)$ does not change for any point $P_N \in N$. Therefore, a relevant number of variables α (hence β, γ, δ and ε) that model the sign of this determinant can be fixed 'a priori'. It is sufficient to check whether both bounds $L(P_N|P^1_{B'}P^2_{B'})$ and $U(P_N|P^1_{B'}P^2_{B'})$ computed in Subsection 4.2.2 have the same sign or not. Figure 11 shows an example where variables α can be fixed. Each of the blue neighbourhoods is completely contained in the half-spaces generated by the barrier, and α is fixed. However, the variable α corresponding to the orange neighbourhood depends on the half-space in which P_N is located.

5. Computational experiments

This section is devoted to studying the performance of (H-TSPHN) and (H-TSPN) formulations proposed in Section 3. In the first subsection, the procedure for generating the considered random instances is described. The second subsection details the experiments that have been conducted. The third subsection reports the results obtained in these experiments.

Figure 11: Fixing α variables when the whole neighborhood lies in one of the half-spaces the generated by the barrier



5.1. Data generation

To generate the instances of our experiments, we assume assumptions **A1-A4** stated in Section 2. The following proposition gives an upper bound for the number of balls that can be generated given an instance with $n = |\mathcal{B}|$ barriers:

Proposition 6. *Under assumptions **A1-A4**, the maximum number of balls that can be considered in* H-TSPHN *is* $O(n^2)$.

Proof. Let $G_{\mathcal{B}} = (V_{\mathcal{B}}, E_{\mathcal{B}})$ the visibility graph of the barrier configuration in $\mathcal{B}$. The proof is based on the properties of this visibility graph described in Mitchell (2017). It is clear that the maximum number of balls coincides with the number of faces, $f_{G_{\mathcal{B}}}$, of $G_{\mathcal{B}}$. Recall that the number of vertices in $\mathcal{B}$ is n. Next, by the Euler formula, the number of faces $f_{G_{\mathcal{B}}}$ is $2 + E_{G_{\mathcal{B}}} - n$. Since $E_{\mathcal{B}} = O(n^2)$, it follows that $f_{G_{\mathcal{B}}} = O(n^2)$. □

Algorithm 3 describes a general way to construct instances where neighbourhoods are balls.

Algorithm 3: General scheme of the instances generation

Initialization: Let $|\mathcal{N}|$ be the number of neighbourhoods to generate.

Let $R = [LB_x, UB_x] \times [LB_y, UB_y] \subseteq \mathbb{R}^2$ be the rectangle where centers will be generated.

Set $points = \{\}$; $\mathcal{B} = \{\}$; $\mathcal{N} = \{\}$.

1 Generate $|\mathcal{N}|$ points uniformly distributed in R and include them in $points$.

2 Generate pairwise disjoint barriers that separate the points and include them in $\mathcal{B}$.

3 Generate neighbourhoods around $points$ and include them in $\mathcal{N}$.

The two following subsections develop Steps 2 and 3 of the Algorithm 3, respectively. Specifically, the Algorithm 4, related with Step 2, details how barriers are generated assuming **A1-A4**, while the

Algorithm 5, related with Step 3, describes the way neighbourhoods are designed. In practice, Algorithms 3, 4 and 5 are implemented to be run sequentially and generate the battery of instances used for testing the formulations.

Barriers generation

In this subsection, we focus on how to generate line segments located in general position without crossings. The idea is to build bisectors that separate each pair of points in the set *points*. The initial length of each bisector is r_{init}. This length is reduced until it does not intersect any of the previously generated line segments. The Algorithm 4 reports the pseudocode to generate barriers.

Algorithm 4: Generation of the barriers

Initialization: Let *points* be the set already randomly uniformly generated in R.
Let r_{init} be one half of the initial length of the barriers.
Set $\mathcal{B} = \{\}$.

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 $\mathcal{B}$.

Neighbourhood generation

Once the set of points and barriers are generated, the neighbourhoods are created using two different sizes:

- **Randomly-sized neighbourhoods:** that do not intersect barriers and verify **A4**.
- **Fixed-sized neighbourhoods:** that can cross barriers and are not required to assume **A4**.

The first case is used to study the performance of the models H-TSPHN and H-TSPN proposed in the article when the neighbourhoods are circles or line segments. The following procedure describes the pseudocode to create circles.

Algorithm 5: Generation of randomly-sized circles

Initialization: Let *points* be the set randomly, uniformly generated in R.

Let $\mathcal{B}$ be the barriers already generated.

Set $\mathcal{N} = \{\}$.

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 $\mathcal{N}$.

The line segments instances are generated by randomly selecting two diametrically opposite points in the boundary of the balls instances obtained with Algorithm 5.

The second case focuses on the effectiveness of the exact model for the H-TSPN in terms of *overlapping ratio* of the circles. This ratio, introduced in Mennell (2009), is calculated by dividing the average radius of the neighbourhood sets by the length of the longest side of R. Mennell (2009) shows that the higher the overlaping ratio, the higher the difficulty of the instance. Algorithm 5 is slightly modified by setting a fixed value for *radii*, based on instances considered in Behdani and Smith (2014).

5.2. Configuration of the experiments

In this work, two experiments are designed to study the behaviour of the models H-TSPHN and H-TSPN. In the first experiment, we generate five instances for each number of randomly-sized neighbourhoods in $|\mathcal{N}| \in \{5, 10, 20, 30, 50, 60, 65, 70, 75, 80, 100\}$ within $R = [0, 100] \times [0, 100]$. These neighbourhoods are balls and line segments that have been created using the Algorithm 5. In addition, barriers are generated according to Algorithm 4. $|\mathcal{B}|$ reports the average number of barriers generated for each experiment. For each instance, we run the models with and without strengthening the formulations.

In the second experiment, based on Behdani and Smith (2014), we generate ten instances with a number N in $|\mathcal{N}| \in \{6, 8, 10, 12, 14, 16, 18, 20\}$ of fixed circles. The centres are drawn in $R = [0, 0] \times [16, 10]$. The fixed *radii* considered are 0.25, 0.5, and 1. Therefore, the overlap ratios are 0.015625, 0.03125 and 0.0625, respectively. We run H-TSPN with strengthening to see the performance of our methods when neighbourhoods overlap.

Formulations are coded in Python 3.9.2 (Van Rossum and Drake (2009)) and solved in Gurobi 9.1.2 (Gurobi Optimization LLC (2022)) on an AMD® Epyc 7402p 8-core processor.

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

- $\#\boldsymbol{Found}$: number of instances in which the solver finds a feasible solution.
- $\boldsymbol{Gap}$: gap between the best incumbent solution with respect to the best bound found by the solver. It is computed as $Gap = (upper\,bound - lower\,bound)/lower\,bound$.
- $\boldsymbol{Time_{model}}$: time (in seconds) spent by the solver to obtain the best solution found.
- $\boldsymbol{Time_{prepro}}$: time (in seconds) spent by Python to set up the model, including the strengthening of the formulation.

- $\boldsymbol{Time_{total}}$: overall time (in seconds) to solve the model.

For all the experiments, a time limit of 1 hour of solver time was set in the branch-and-bound procedure.

5.3. Results of the experiments

We report the results of the first experiment in Table 2. The layout is organised in 3 blocks of columns. The first block describes the parameters of the problem: number of neighbourhoods $|\mathcal{N}|$, if **A4** is assumed or not (H-TSPHN vs H-TSPN), if strengthening is considered, and the number of barriers. The second and third blocks describe the average results obtained with the solver for circles and segments, respectively.

Analysing the results in Table 2, we first observe that solving the problem considering balls as neighbourhoods is harder than solving with segments. Next, we also observe that this approach solves to optimality all instances for circles up to $|\mathcal{N}| = 30$ with strengthening for the H-TSPHN problem. However, if we do not strengthen the formulation, the solver always reports gap for all the instances with circles. The same behaviour can be seen for the H-TSPN up to $|\mathcal{N}| = 30$. Nevertheless, for sizes $|\mathcal{N}|$ in 50-75, both formulations start to differ in the number of instances in which the solver can find a feasible solution. For the H-TSPHN, the reader may observe that strengthening does not improve the gap within the time limit. However, in the H-TSPN, strengthening does increase the number of instances in which the solver finds a solution and the fraction of gaps certified after the execution time. Anyway, whenever a feasible solution is found, the maximum average gap that is reported with strengthening is 0.35. Finally, for sizes 80 and 100, the solver cannot find any solution for any of the models, regardless of whether strengthening is considered or not. In terms of execution time, we can conclude that strengthening always improves both the time to obtain the optimal solution ($\boldsymbol{Time_{model}}$) and the time the computer takes to load all the variables and constraints of the model ($\boldsymbol{Time_{prepro}}$). This difference is more appreciable in the H-TSPHN, as the reader can notice in the aggregation of these two columns in ($\boldsymbol{Time_{total}}$). This fact can be explained in terms of the number of barriers: the higher the number of barriers, the larger the number of variables that can be fixed beforehand. On the other hand, we can observe that Gurobi can report the optimal solution for almost all segment instances generated by solving the strengthened versions of the H-TSPHN and H-TSPN. In addition, the time spent to solve all these instances is lower than the time limit. However, the time to load and strengthen the model is very similar to that for circle instances.

In Figures 12 and 13, the reader can compare, at a glance, the time that the solver spent to obtain the best solution found and the gap between this solution and the best bound found by the solver. We can conclude that strengthening improves significantly the time spent by Gurobi to get the optimal solution and instances with circles are harder to be solved than those with segments, in terms of time and final gap.

In Table 3, we detail the values obtained with Gurobi for the second experiment. The first block reports the size of the instances and the average number of barriers for each size. The other three blocks describe the features reported by the solver to compare the efficiency of the model for *radii* equals to

Table 2: Computational results obtained with (H-TSPHN) and (H-TSPN)

				Circles					*Segments*				
$\|\mathcal{N}\|$	A_4	*Strengthening*	$\|\mathcal{B}\|$	*#Found*	*Gap*	$Time_{model}$	$Time_{prepro}$	$Time_{total}$	*#Found*	*Gap*	$Time_{model}$	$Time_{prepro}$	$Time_{total}$
5	no	no	4.8	5	0	157.24	0.29	157.53	5	0	42.59	0.91	43.5
		yes	4.8	5	0	1.24	0.44	1.68	5	0	0.42	1.47	1.89
	yes	no	10.4	5	0.1	819.98	1.3	821.28	5	0	22.03	1.58	23.61
		yes	10.4	5	0	0.61	1.16	1.77	5	0	0.38	1.57	1.95
10	no	no	9.2	5	0.17	2193.53	2.09	2195.62	5	0.42	2884.22	6.6	2890.82
		yes	9.2	5	0	9.1	2.58	11.68	5	0	1.69	9.61	11.3
	yes	no	19.2	5	0.17	933.67	9.59	943.26	5	0.3	1448.73	11.8	1460.53
		yes	19.2	5	0	2.56	6.36	8.92	5	0	1.53	9.24	10.77
20	no	no	17.6	5	0.17	3600	16.15	3616.15	5	0.2	3600	50.93	3650.93
		yes	17.6	5	0	68.67	17.43	86.1	5	0	11.66	74.89	86.55
	yes	no	35.8	5	0.21	2349.26	87.7	2436.96	5	0	145.06	111.34	256.4
		yes	35.8	5	0	42.45	43.49	85.94	5	0	8.05	64.01	72.06
30	no	no	28	4	0.5	3600	75.15	3675.15	5	0.4	3600	201.05	3801.05
		yes	28	5	0	2034.53	65.23	2099.76	5	0	54.35	246.96	301.31
	yes	no	56.4	5	0.23	3027.02	512.03	3539.05	5	0.18	1039.08	672.46	1711.54
		yes	56.4	5	0	147.98	179.95	327.93	5	0	96.72	270.56	367.28
50	no	no	44.2	3	0.87	3600	364.75	3964.75	4	0.3	3600	960.53	4560.53
		yes	44.2	3	0	2485.76	297.99	2783.75	5	0	311.49	1043.15	1354.64
	yes	no	89	5	0.37	3600	4650.3	8250.3	5	0	1445.39	4654.95	6100.34
		yes	89	5	0.1	3600	1213.85	4813.85	5	0	353.17	1292.12	1645.29
60	no	no	50.2	0	-	-	-	-	5	0.53	3600	1213.08	4813.08
		yes	50.2	3	0.22	3600	538.11	4138.11	5	0	1384.92	1103.43	2488.35
	yes	no	100.8	5	0.15	3600	8688.65	12288.65	5	0	1671.85	8726.63	10398.48
		yes	100.8	5	0.13	3600	2179.26	5779.26	5	0.01	2903.27	2339.58	5242.85
65	no	no	52.8	0	-	-	-	-	4	0.75	3600	1561.46	5161.46
		yes	52.8	2	0.35	3600	671.07	4271.07	5	0.02	3249.12	1343.12	4592.24
	yes	no	106	5	0.13	3600	11250.62	14850.62	5	0	1381.66	11269.25	12650.91
		yes	106	5	0.17	3600	2843.48	6443.48	5	0.01	2877.66	3003.47	5881.13
70	no	no	57.6	0	-	-	-	-	4	0.85	3600	1977.3	5577.3
		yes	57.6	3	0.12	3600	898.99	4498.99	5	0.04	3211.2	1754.51	4965.71
	yes	no	115.6	5	0.29	3600	17366.28	20966.28	5	0.04	2853.58	17433.28	20286.86
		yes	115.6	5	0.32	3600	4106.95	7706.95	5	0.02	3203.33	4311.14	7514.47
75	no	no	63.2	0	-	-	-	-	3	0.74	3600	2976.6	6576.6
		yes	63.2	0	-	-	-	-	5	0.24	3283.37	242407	5707,44
	yes	no	126.8	4	0.24	3600	26363.48	29963.48	5	0.01	1903.99	26153.87	28057.86
		yes	126.8	3	0.23	3600	5382.14	8982.14	5	0.02	2458.75	6140.02	8598.77
80	no	no	64	0	-	-	-	-	1	0.83	3600	4205.41	7805.41
		yes	64	0	-	-	-	-	5	0	1775.16	2858.51	4633.67
	yes	no	128.6	0	-	-	-	-	5	0.11	3471.06	29073.69	32544.75
		yes	128.6	0	-	-	-	-	5	0	2701.21	7031.87	9733.08
100	no	no	81.6	0	-	-	-	-	0	-	-	-	-
		yes	81.6	0	-	-	-	-	4	0	1761.08	6620.09	8381.17
	yes	no	163.2	0	-	-	-	-	5	0.48	3600	91153.55	94753.55
		yes	163.2	0	-	-	-	-	5	0	2720.76	19429.1	22149.86

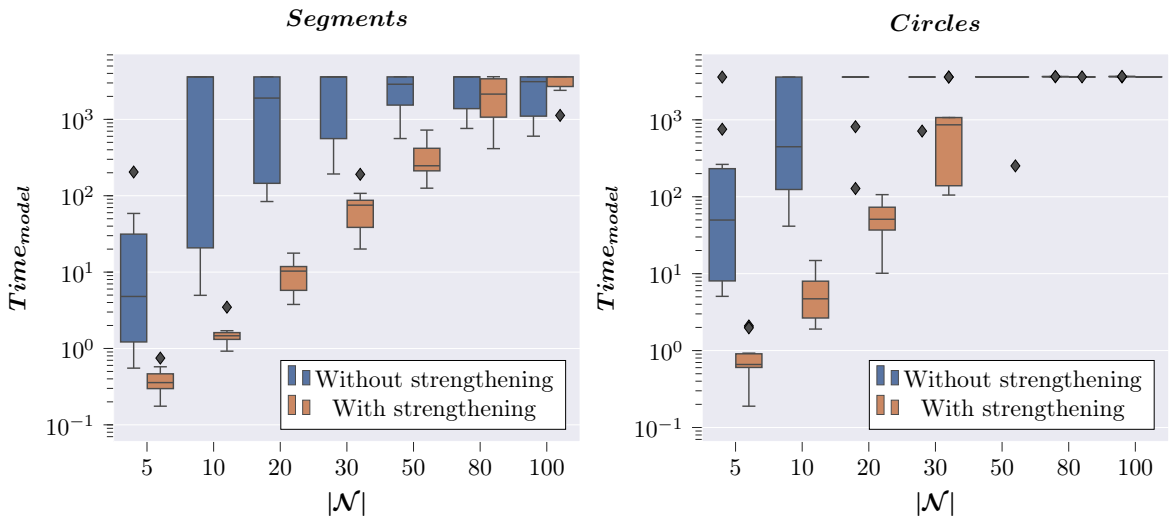


Figure 12: Runtime of the model (H-TSPN) without and with strengthening when the neighborhoods are segments and balls.

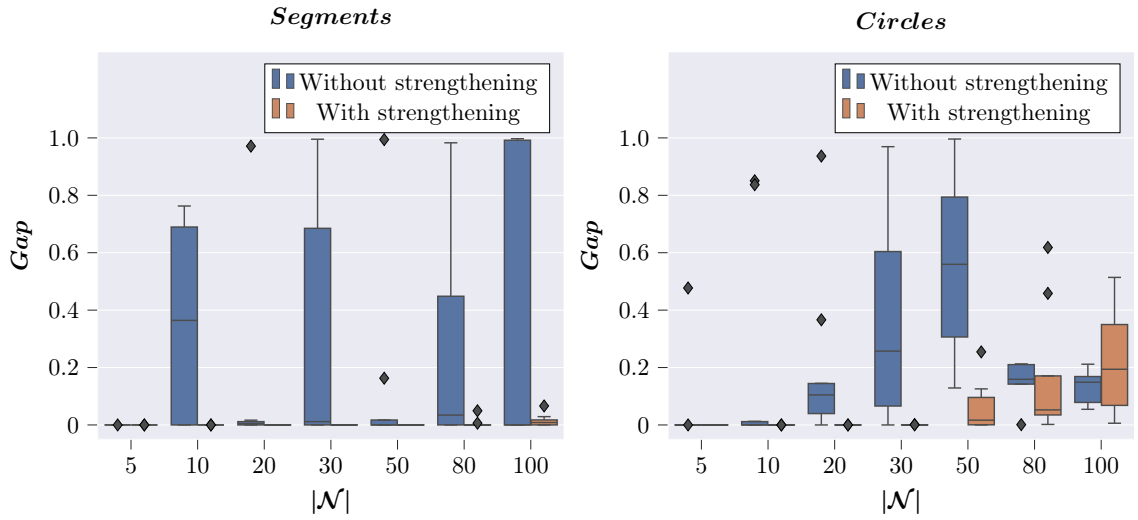


Figure 13: Gap of the model (H-TSPN) without and with strengthening when the neighborhoods are segments and balls.

0.25, 0.5 and 1, respectively. For the smallest *radii*, all the instances are almost solved to optimality reporting a maximum average gap of 0.03 for $|\mathcal{N}| = 20$. In the instances with $radii = 0.5$, the number of feasible solutions found starts to decrease from size 16. In terms of the gap, from $|\mathcal{N}| = 14$, the solver is unable to find the optimal solution because it reaches the time limit. In any case, the maximum average gap for these instances is 0.32 for the largest size. Finally, for $radii = 1$, only instances with $|\mathcal{N}| = 6$ are solved to optimality. For sizes 10, 12 and 14, the solver only finds a feasible solution for one half of the instances, approximately. However, it cannot find any solution for $|\mathcal{N}| \geq 16$. The results obtained for these instances lead us to conclude that the larger the radii of the neighbourhood, the higher the complexity to be solved. This conclusion is in line with the existing trend in the literature, as exposed in Puerto and Valverde (2022) or Blanco et al. (2017).

Table 3: Computational results obtained with (H-TSPN) for Smith instances

		$Radii = 0.25$					$Radii = 0.5$					$Radii = 1$				
$\|\mathcal{N}\|$	$\|\mathcal{B}\|$	$\#Found$	Gap	$Time_{model}$	$Time_{prepro}$	$Time_{total}$	$\#Found$	Gap	$Time_{model}$	$Time_{prepro}$	$Time_{total}$	$\#Found$	Gap	$Time_{model}$	$Time_{prepro}$	$Time_{total}$
6	8.6	10	0	2.58	1.21	3.79	10	0	16.31	1.26	17.57	10	0.03	1512.68	1.4	1514.08
8	12.7	10	0	14.12	3.07	17.19	10	0	171.44	3.16	174.6	10	0.27	3600	3.56	3603.56
10	18	10	0	199.23	7.35	206.58	10	0.03	1453.59	7.55	1461.14	5	0.35	3600	8.5	3608.5
12	19.7	10	0	149.49	10.12	159.61	10	0.09	3454.33	10.54	3464.87	6	0.46	3600	12.46	3612.46
14	24.5	10	0	374.8	23.53	398.33	10	0.11	3600	24.23	3624.23	4	0.65	3600	26.56	3626.56
16	29.4	10	0.01	2486.62	40.2	2526.82	8	0.16	3600	41.1	3641.1	0	-	-	-	-
18	32	10	0.03	3053.15	52.18	3105.33	7	0.21	3600	53.05	3653.05	0	-	-	-	-
20	37	10	0.03	3040.06	75.25	3115.31	6	0.32	3600	80.03	3680.03	0	-	-	-	-

6. Concluding Remarks

This paper has dealt with two problems, the H-SPPN and the H-TSPN. In both cases, we have assumed that barriers do not allow direct movements between neighbourhoods **A4**. The more general case that does not assume **A4** gives rise to nonconvex mixed-integer problems. It is still an open problem whether there is some kind of finite dominating set with polynomial cardinality for the version of the H-TSPN which could help simplify the formulation of the problem. These questions are very interesting but beyond the scope of this paper. Needless to say, we plan to continue its analysis in a follow-up paper.

It would also be interesting to combine in the same model different typologies of barriers such as polygonals and second-order cone-representable sets. It is also interesting to consider the single- or multiple facility problem with barriers and neighbourhoods.

All the above-mentioned problems are natural extensions of the ones considered in this paper and will deserve our attention in the future.

References

Arkin, E. M. and Hassin, R. (1994). Approximation algorithms for the geometric covering salesman problem. *Discrete Applied Mathematics*, 55(3):197–218.

Behdani, B. and Smith, J. C. (2014). An Integer-Programming-Based Approach to the Close-Enough Traveling Salesman Problem. *INFORMS Journal on Computing*, 26(3):415–432.

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.

Carrabs, F., Cerrone, C., Cerulli, R., and Golden, B. (2020). An Adaptive Heuristic Approach to Compute Upper and Lower Bounds for the Close-Enough Traveling Salesman Problem. *INFORMS Journal on Computing*, 32(4):1030–1048.

Coutinho, W. P., do Nascimento, R. Q., Pessoa, A. A., and Subramanian, A. (2016). A Branch-and-Bound Algorithm for the Close-Enough Traveling Salesman Problem. *INFORMS Journal on Computing*, 28(4):752–765.

D. Perez, E. J. Powley, D. Whitehouse, P. Rohlfshagen, S. Samothrakis, P. I. Cowling, and S. M. Lucas (2014). Solving the Physical Traveling Salesman Problem: Tree Search and Macro Actions. *IEEE Transactions on Computational Intelligence and AI in Games*, 6(1):31–45.

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

Glock, K. and Meyer, A. (2023). Spatial coverage in routing and path planning problems. *European Journal of Operational Research*, 305(1):1–20.

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

J. . -P. Laumond, P. E. Jacobs, M. Taix, and R. M. Murray (1994). A motion planner for nonholonomic mobile robots. *IEEE Transactions on Robotics and Automation*, 10(5):577–593.

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

Letchford, A. N., Nasiri, S. D., and Theis, D. O. (2013). Compact formulations of the Steiner Traveling Salesman Problem and related problems. *European Journal of Operational Research*, 228(1):83–92.

Lobo, M. S., Vandenberghe, L., Boyd, S., and Lebret, H. (1998). Applications of second-order cone programming. *International Linear Algebra Society (ILAS) Symposium on Fast Algorithms for Control, Signals and Image Processing*, 284(1):193–228.

Mennell, W. K. (2009). *Heuristics for Solving Three Routing Problems: Close-enough Traveling Salesman Problem, Close-Enough Vehicle Routing Problem, and Sequence-Dependent Team Orienteering Problem*. PhD thesis, University of Maryland, College Park, United States – Maryland.

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

Nesterov, Y. and Nemirovski, A. (1994). *Interior-Point Polynomial Algorithms in Convex Programming*. Studies in Applied and Numerical Mathematics. Society for Industrial and Applied Mathematics.

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

T. Moemke and O. Svensson (22). Approximating Graphic TSP by Matchings. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 560–569.

Van Rossum, G. and Drake, F. L. (2009). *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA.

Y. K. Hwang and N. Ahuja (1992). A potential field approach to path planning. *IEEE Transactions on Robotics and Automation*, 8(1):23–32.

Yuan, B. and Zhang, T. (2017). Towards Solving TSPN with Arbitrary Neighborhoods: A Hybrid Solution. In Wagner, M., Li, X., and Hendtlass, T., editors, *Artificial Life and Computational Intelligence*, pages 204–215, Cham. Springer International Publishing.