The Hampered Traveling Salesman Problem with Neighbourhoods

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

[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 traveling salesman problems with neighbours and barriers. Each one of these two elements, neighbours and barriers, make the problems harder than their standard counterparts. Therefore, mixing both together results in a new challenging problem that, as far as we know, has not been addressed before but that has applications for inspection and surveillance activities and the delivery industry assuming uniformly distributed demand in some regions. 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 80 neighbourhoods and a range between 125-145 barriers.

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

1. Introduction

Routing problems are considered classical problems in the core of combinatorial optimisation. Among them, the Traveling Salesman Problem (TSP) is one of its more important representatives 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 the customers are willing to pick up the orders (they can be seen as uniform probability densities) in the delivery industry. Moreover, they can be also used for modelling some areas that must be inspected by the drone (whenever visiting a point of these areas is enough to consider them as inspected). On the other hand, the graphic TSP (see Möemke and Svensson (2011)), where distances are measures by geodesics (shortest paths in the respective metric space), has

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

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 analyze the mathematical implications of barriers to the geometry and computation of routes in a continuous space. Beyond the mathematical relevance by 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 in 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 location of targets to be visited are uniformly distributed in neighbourhoods giving rise to challenging problems in the continuous space. Nevertheless, the resulting problems still keep geometric elements that must be exploited to partially overcome the difficulties of the solution approaches and algorithms in the design of routes among neighbourhoods with barriers.

The well-known Traveling Salesman Problem with Neighbourhoods (TSPN) was introduced by Arkin and Hassin (1994). These authors adressed 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 nonlinear 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 (2022), the concept of spatial coverage in routing and path planning is unified and defined and a categorization scheme of related problems is introduced. Other classical combinatorial problems as the Minimum Spanning Tree (MST) or the Crossing Postman Problem (XPP) that have been extended to deal with neighbourhoods are developed in Blanco et al. (2017) and Puerto and Valverde (2022), respectively.

As far as we are concerned, the use of barriers in location problems has been widely considered (see Klamroth (2002)) but the corresponding routing problem with barriers has attracted less attention in the Operations Research field. In spite of that, one can find connections with some problems in the area of computational geometry as the shortest path problem in polygons and the touring polytopes problem (see Mitchell (2017)), in navigation games competition as for instance the Physical Traveling Salesman Problem (see Perez et al. (2014)), or in robot motion planning (see Hwang and Ahuja (1992) and Laumond et al. (1994)). Different complexity results are known and a lot of effective heuristic algorithms are 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) make the problem hard. 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. In spite that we prove that the H-SPPN is polynomially solvable, we also give a formulation for this problem which highlights the constraints that are necessary for the H-TSPN. The respective H-TSPN is NP-hard. We give exact formulations using a geodesic shortest path representation that allows to solve medium size 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 80 neighbourhoods and a range between 125-145 barriers.

The paper is organized in 8 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 along 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 hypothesis that describe the framework where we analyze the considered problems. We will deal with two problems in this paper: the Hampered Shortest Path Problem with Neighbourhoods H-SPPN and the Hampered Traveling 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 the *drone*. In both considered problems, we denote by $\mathcal{B}$ the set of line segments that model the barriers and we adopt the assumptions that are listed below.

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 two neighbourhoods without crossing an obstacle.

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

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 joining 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**.

2.1. 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 $N_T \subset \mathbb{R}^2$, that 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 minimize 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.
- $V_{\mathcal{B}} = \{P^1_B, P^2_B : B = \overline{P^1_B P^2_B} \in \mathcal{B}\}$. Set of vertices that come from the endpoints of 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 line segments that join the point selected in the source neighbourhood N_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 $\mathcal{B}$.

- $E_T = \{(P_B^i, P_T) : P_B^i \in V_{\mathcal{B}} \text{ and } \overline{P_B^i P_T} \cap B'' = \emptyset, \forall B'' \in \mathcal{B},\ i = 1, 2\}$. Set of edges formed by the line segments that join the point selected in the target neighbourhood N_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 $G_{\text{SPP}} = (V_{\text{SPP}}, E_{\text{SPP}})$ induced by the barriers and neighbourhoods, where $V_{\text{SPP}} = V_S \cup V_{\mathcal{B}} \cup V_T$ and $E_{\text{SPP}} = E_S \cup E_{\mathcal{B}} \cup E_T$.

2.2. Description of the Hampered Traveling Salesman Problem with Neighbourhoods

The H-TSPN 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-TSPN is to seek for 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-TSPN, 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 the 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.

- $E_{\mathcal{N}} = \{(P_N, P_B^i) : P_B^i \in V_{\mathcal{B}} \text{ and } \overline{P_N P_B^i} \cap B'' = \emptyset, \forall B'' \in \mathcal{B},\ i = 1, 2\}$. Set of edges formed by the line segments that join each point selected in the neighbourhoods of $\mathcal{N}$ with every endpoint in 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 the 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 graph $G_{\text{TSP}} = (V_{\text{TSP}}, E_{\text{TSP}})$ induced by the barriers and neighbourhoods, where $V_{\text{TSP}} = V_{\mathcal{N}} \cup V_{\mathcal{B}}$ and $E_{\text{TSP}} = E_{\mathcal{N}} \cup E_{\mathcal{B}}$.

Note that, in this case, it may be also interesting to consider this problem without imposing assumption **A4**. This more general version is called the Hampered Traveling Salesman Problem with Visible neighbourhoods H-TSPVN. In this case, we do not require 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. The main difference lies in the description of edges for the graph induced by the neighbourhoods and endpoints of barriers.

By taking the same approach, 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.
- $V_{\text{TSPV}} = V_{\mathcal{N}} \cup V_{\mathcal{B}}$.
- $E_{\text{TSPV}} = \{(P, P') : P, P' \in V_{TSP} \text{ and } \overline{PP'} \cap B'' = \emptyset, \forall B'' \in \mathcal{B}\}$. Set of edges formed by the line segments that join every pair of points in V_{TSP} that do not cross any barrier.

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

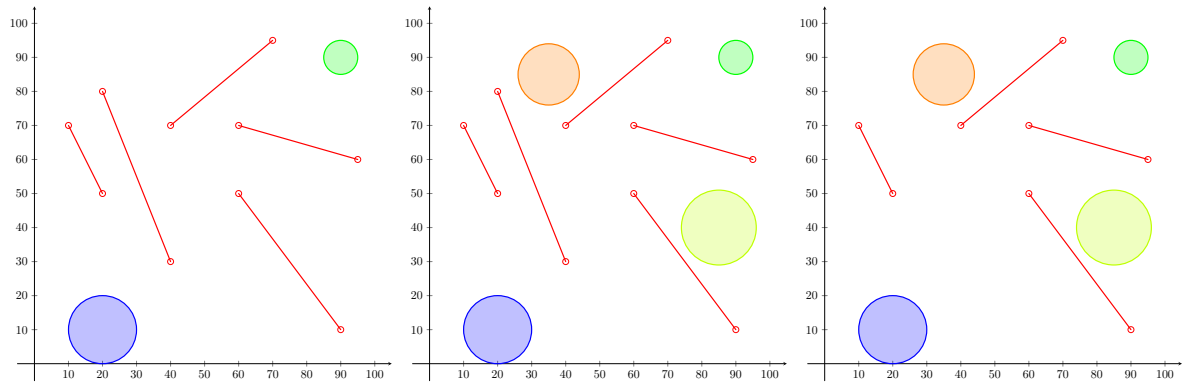


Figure 1: Problem data of the H-SPPN, H-TSPN and H-TSPVN

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 $X \in \{SPP, TSP, TSPV\}$. Finally, the formulations for the H-SPPN, H-TSPN and H-TSPVN 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 combinatorial algorithm in polynomial time. Indeed, we will need that type of representation to address some NP-hard problems that require computing shortest paths as building blocks, as for instance the location problem with barriers and neighbours and the H-TSPN. Therefore, we will start providing two different mathematical programming formulations for the H-SPPN able to be embedded in more complex problems.

First of all, we introduce the decision variables that represent the problem. These are summarized 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

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

For modelling of the distance between pair of points, we introduce the non-negative continuous variable $d(PQ)$ that represents the distance between P and Q:

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

where E_X is the set of edges E_{SPP}, E_{TSP} or E_{TSPV}.

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:

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

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

It is remarkable that 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).

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

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

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

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

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_{TSP} or V_{TSPV}. 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, that assumes the value one if the determinant is positive and zero, otherwise. Note that the case when determinants are null does not need to be considered, because the segments are located in general position.

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

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

where L and U are lower and upper bounds for the value of 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, to check whether the pairs of determinants

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

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

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

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

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 linearized by 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 that is one if 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 enforces ε to be zero because the right hand side is fractional and the first inequality is non-positive. Nonetheless, if no barrier intersects the segment $\overline{PQ}$, then ε is equals to one, because the left hand side of the first inequality is one and the right hand side of the second inequality too.

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

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

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

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

in the neighbourhoods as shown in Figure 2. The two subfigures show how the graph G_{SPP} is generated. The blue dashed line segments represent the edges of E_S, the green ones, the edges of E_T and the red dashed lines, the edges of $E_{\mathcal{B}}$. 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.

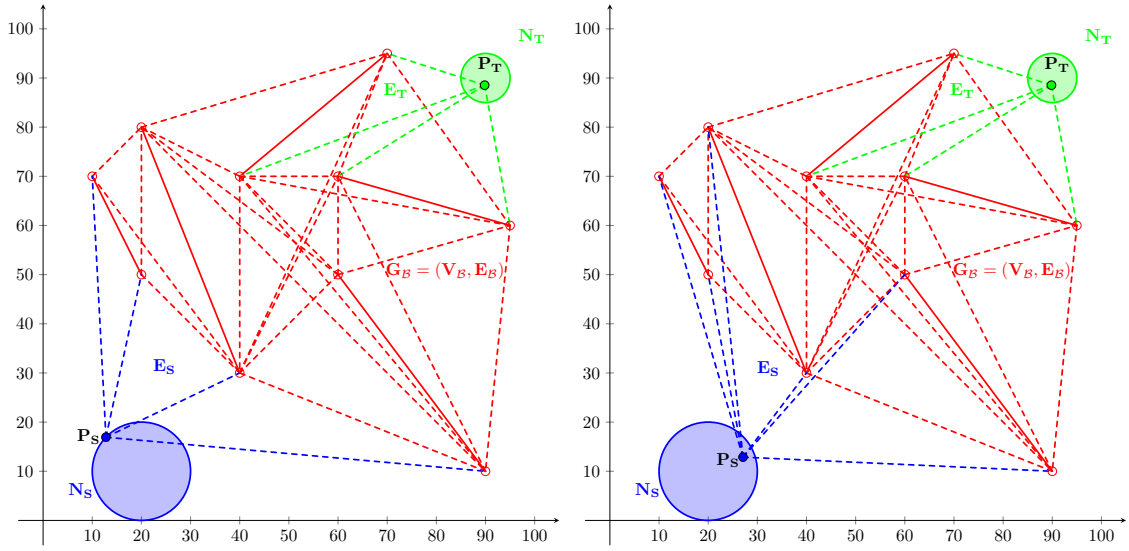


Figure 2: The construction of the graph $G = (V, E)$

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 2.1.

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.

By taking into account the constraints explained in 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 minimizes the length of the path followed by the drone on 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 points selected must be in their respective neighbourhoods.

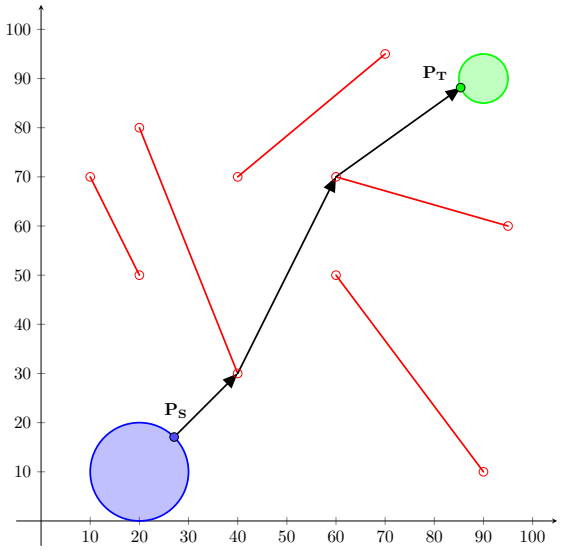


Figure 3: Optimal solution for this instance of the H-SPPN

3.3.1. Reformulating the H-SPPN

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

Proposition 1. *There exists 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 visited barriers in any optimal solution, 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_S^* and N_T^* by solving a convex problem for each endpoint of the barriers:

$$N_S^* = \{P_S(P_B^i) : P_S(P_B^i) = \arg\min_{P_S \in N_S} \|P_S - P_B^i\| \text{ and } \varepsilon(P_S P_B^i) = 1\},$$

$$N_T^* = \{P_T(P_B^i) : P_T(P_B^i) = \arg\min_{P_T \in N_T} \|P_B^i - P_T\| \text{ and } \varepsilon(P_B^i P_T) = 1\}.$$

The graph required to address the new formulation makes use of the previous graph G_{SPP} and then we augment new edges connecting 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 minimizes 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 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 joining the point selected $P_S \in N_S$ 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 joining 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 sets N_S^* and N_T^* are computed beforehand, the whole 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 4 shows how the dominating sets N_S^* and N_T^* are computed for an instance of the H-SPPN.

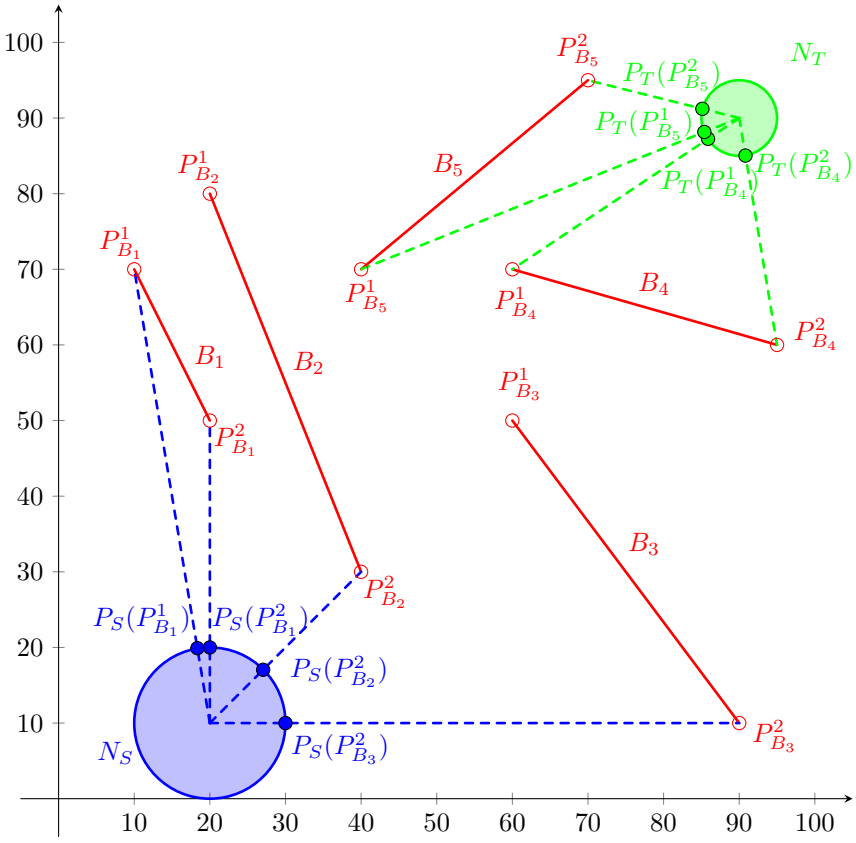


Figure 4: Illustration of the dominating sets associated with an instance of the H-SPPN.

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:

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

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

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

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

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 of all, 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 linear size

shortest map from any given point can be computed in $O(n \log n)$ time. Once one has 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 $P_S(P_B^i) \in N_S^*$ we compute its linear size shortest path map and solve the shortest path problem with respect to all the $O(n)$ points in N_T^*. Overall, this operation takes $O(n \log n + n^2)$. The same operation has to be repeated for all the $O(n)$ points in N_S^*. Hence, the overall complexity for solving the H-SPPN is $O(n^3)$.

□

3.4. A formulation for the H-TSPN

The rationale of the formulation for the H-TSPN 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, this simplification cannot be applied to obtain an optimal solution for the H-TSPN, although it may produce an approximation to generate lower bounds for the H-TSPN.

Our approach relies on adjusting a single-commodity flow formulation to ensure connectivity. We can assume wlog that the neighbourhood N_1 is required and the drone departs from that depot (assuming to be 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 $(P, Q) \in E_{\text{TSP}}$, we define the following variables:

- $y(PQ)$, binary variable that is 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_{TSP} as follows:

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

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

The first group of constraints impose that the drone departs from each neighbourhood. The second block of constraints are the flow conservation constraints. The third inequalities ensure that one unit of commodity is delivered to each of the required neighbourhood. The fourth ones ensure that the ficticious 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. Inequalities (α-C), (β-C), (γ-C), (δ-C), (ε-C), (y-C), (d-C), (N-C) enforce the variables of the problem to be well-defined.

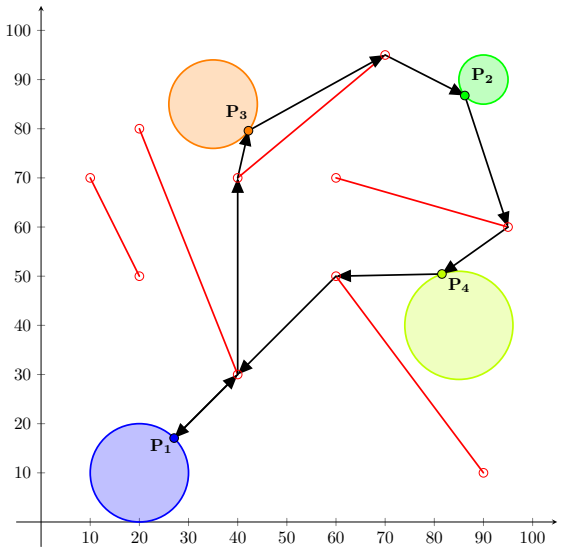


Figure 5: Solution for the instance of the H-TSPN

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

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

3.4.1. Reformulating the H-TSPN

The idea of computing a dominating set that represents each of the neighbourhoods in the H-SPPN can be adopted to reformulate the H-TSPN 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. Hence, the points chosen in an optimal solution for H-TSPN must be those that produce the minimum distances to the points of the previously and next visited barriers in the optimal solution. Therefore, N^* must be composed, at most, by 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 neighbourhoods.

We define the graph $G_{\text{TSP}}^* = (V_{\text{TSP}}^*, E_{\text{TSP}}^*)$, where $V_{\text{TSP}}^* = V_{\text{TSP}} \cup V_{\mathcal{N}}^*$ and $E_{\text{TSP}}^* = E_{\mathcal{N}}^* \cup E_{\mathcal{B}}$.

Again the sets N^* for each $N \in \mathcal{N}$ can be computed in advance so that the whole graph G_{TSP}^* will be fixed. The new formulation for the H-TSPN can be represented as the next simplified program:

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

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

$$\sum_{\{Q:(P,Q)\in E^*_{\text{TSP}}\}} y(PQ) = \sum_{\{Q:(Q,P)\in E^*_{\text{TSP}}\}} y(QP), \quad \forall P \in V^*_{\text{TSP}},$$

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

$$\sum_{\{Q:(Q,P)\in E^*_{\text{TSP}}\}} g(QP) - \sum_{\{Q:(P,Q)\in E^*_{\text{TSP}}\}} g(PQ) = 0, \qquad \forall P \in V_{\mathcal{B}} \cup V^*_{\mathcal{N}},$$

$$g(PQ) \leq (|\mathcal{N}|-1)y(PQ), \qquad \forall (P,Q) \in E^*_{\text{TSP}},$$

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

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

The reader may note that the maximum number of dominating points associated with an instance $(\mathcal{N},\mathcal{B})$ of the H-TSPN is $\frac{1}{2}|\mathcal{N}||\mathcal{B}|(|\mathcal{B}|-1)$. Therefore, to get all the 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-TSPVN

In this subsection, we analyze the differences between the H-TSPN and the H-TSPVN, where going from one neighbourhood to another one is possible 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_{TSPV} and E_{TSPV}, as described in Subsection 2.2.

The difference between the set of edges in the H-TSPVN with respect to the graph in H-TSPN is that, in the former case, the edges that join each pair of neighbourhoods must be considered. This fact leads to include product of continuous variables in the α constraints of the model that represent the determinants that determine if the segment joining the two variable points in the neighbourhoods cross any barrier or not. These products make the problem to become non-convex.

4. Strengthening the formulations

4.1. Preprocessing

In this subsection, a preprocessing result that allows one to fix some variables is proposed. It is based on analyzing the relative position between the neighbourhoods and the barriers. Specifically, we present a sufficient condition that ensures that there are some barriers whose endpoints can not be incident in 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 remark 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 6), i.e.

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

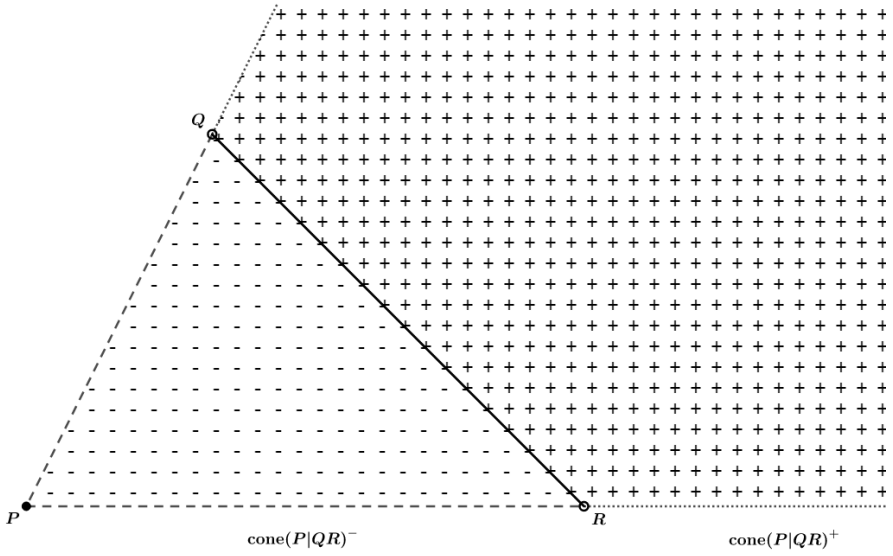


Figure 6: Representation of the cone generated by the point P and the line segment $\overline{QR}$.

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 when the neighbourhoods are segments. Let $N = \overline{P_N^1 P_N^2}$ be a line segment and r_N the straight line that contains the line segment N that is 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 in 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

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

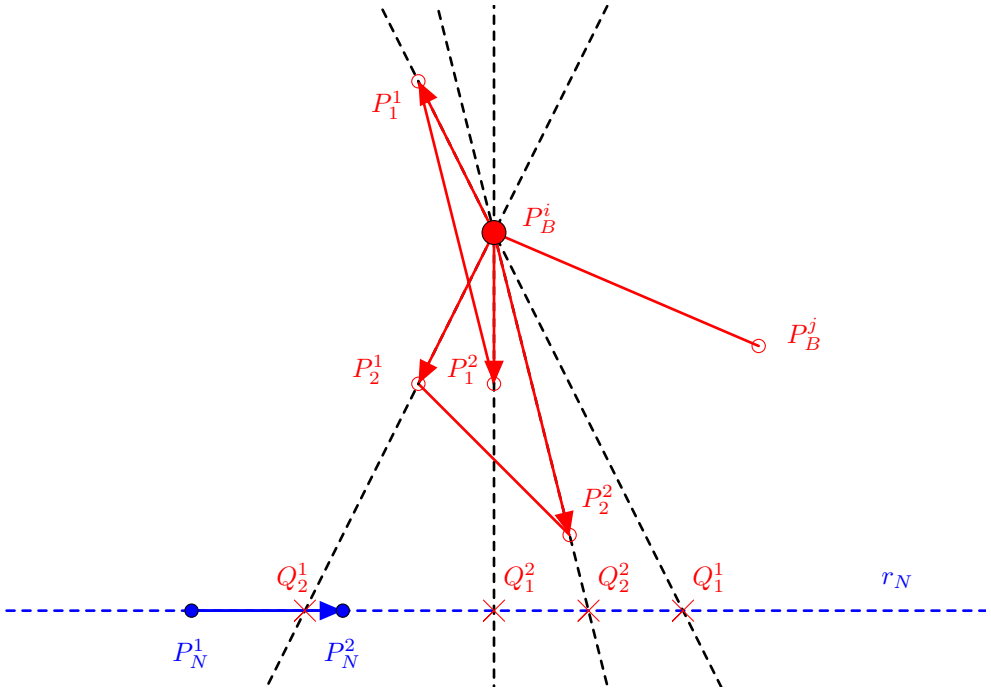
then $(P_N, P_B^i) \notin E_{\mathcal{N}}$.

In Figure 7 an example of the Algorithm 1 is described to check if there exists a rectilinear path joining 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 one 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 one of these

points has associated two 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 non-decreasing 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 $E_{\mathcal{N}}$.



	Q_1^1	P_N^1	Q_2^1	P_N^2	Q_1^2	Q_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	$\frac{9}{4}$

Figure 7: Example of the Algorithm 1

Note that this algorithm also allows us to decide whether the drone can access to a point of a barrier from any point of 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 of N being 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 modeling of the sign requires one to compute 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 its 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 an extended form, we need to find:

$$\text{minimize} \quad F(x,y) = \begin{vmatrix} P_{B'_x}^1 - x & P_{B'_x}^2 - x \\ P_{B'_y}^1 - y & P_{B'_y}^2 - y \end{vmatrix} = xP_{B'_y}^1 - xP_{B'_y}^2 + yP_{B'_x}^2 - yP_{B'_x}^1 + P_{B'_x}^1 P_{B'_y}^2 - P_{B'_y}^1 P_{B'_x}^2,$$

(L-Ellipse)

$$\text{subject to} \quad ax^2 + by^2 + cxy + dx + ey + f \le 0.$$

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:

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

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 8 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 the (H-TSPN) formulation 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.

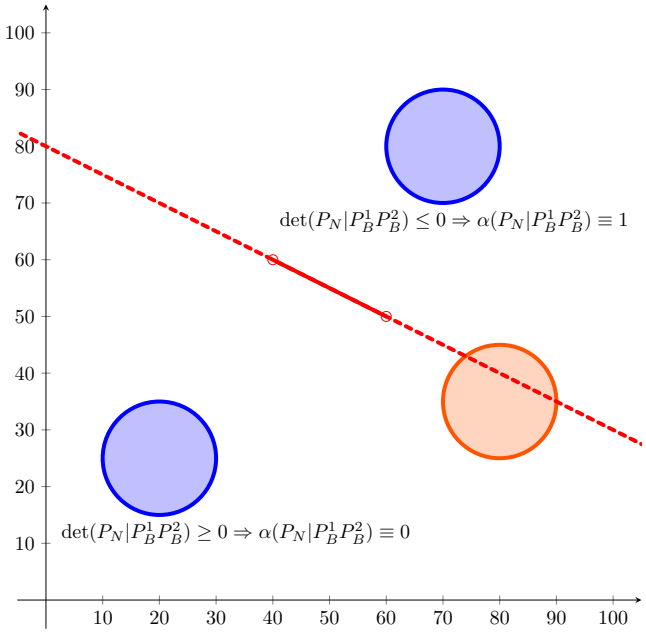


Figure 8: Fixing α variables when the whole neighborhood lies in one of the half-spaces the generated by the barrier

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-TSPN *is* $O(n^2)$.

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

The idea is to generate line segments located in general position without crossings and neighbourhoods that do not intersect with these line segments. The following procedure describes how to construct instances where the neighbourhoods are balls.

Algorithm 3: Generation of instances when the neighbourhoods are balls

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

Generate $|\mathcal{N}|$ points uniformly distributed in the square $[0,100]^2$ and include them in $points$.

for $P, P' \in points$ **do**

  if $\overline{PP'} \cap B = \emptyset$, $\forall B \in \mathcal{B}$ **then**

    Compute $\overrightarrow{d} = \overrightarrow{PP'}$.

    Compute $M = P + \frac{1}{2}\overrightarrow{d}$.

    Compute the unitary vector $\overrightarrow{n_u}$ perpendicular to $\overrightarrow{d}$.

    Set $r = r_{\text{init}}$.

    Generate the barrier $B(r) = \overline{P_B^+ P_B^-}$ where $P_B^{\pm} = M \pm r\overrightarrow{n_u}$.

    while $B(r) \cap B' \neq \emptyset$ *for some* $B' \in \mathcal{B}$ **do**

      Set $r := r/2$.

      Generate the barrier $B(r)$.

    Include $B(r)$ in $\mathcal{B}$.

for $P \in points$ **do**

  Set $r_{\min} = \min_{\{P_B \in B : B \in \mathcal{B}\}} d(P, P_B)$.

  Generate a random $radii$ uniformly distributed in the interval $\left[\frac{1}{2}r_{\min}, r_{\min}\right]$.

  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 3.

5.2. Configuration of the experiments

Since no benchmark instances are available for this problem in the literature, we have generated ten instances with a number of neighbourhoods $|\mathcal{N}| \in \{5, 10, 20, 30, 50, 80, 100\}$ of two typologies: balls and line segments. We have distinguished the cases with and without preprocessing the variables of the formulations, as explained in Subsection 4.1, and we have reported the average results.

The 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. A time limit of 1 hour was set in the experiments.

5.3. Results of the experiments

We report the results of our experiments in Table 2. The layout is organised in 5 blocks of columns. The first one describes the number of neighbourhoods $|\mathcal{N}|$, the second (neighbourhood) describes the typology of neighbourhood used in the experiment (Balls or Segments), the third one (preprocessing) informs whether the preprocessing is being applied or not (True or False). The next three blocks include information on the number of barriers ($|\mathcal{B}|$), the gap at termination (Gap) defined as $Gap =$

$(upper\ bound - lower\ bound)/lower\ bound$, and the computing time (Runtime). The block $|\mathcal{B}|$ indicates the minimum and maximum number of barriers considered in each of the instances for each fixed number $|\mathcal{N}|$ of neighbours. The block Gap contains three columns reporting the average (mean), minimum (min), and maximum (max) gaps of the ten instances considered for each combination of parameters. Analogously, the block Runtime also includes three columns reporting the average (mean), minimum (min) and maximum (max) computing time of the ten instances considered for each combination of parameters. By rows, Table 2 reports results for instances with a number of neigbourhoods $|\mathcal{N}|$ in the set $\{5, 10, 20, 30, 50, 80, 100\}$. For each value of $|\mathcal{N}|$ we report results for two typologies of neigbourhoods, namely Balls and Segments, without preprocessing (False) and with preprocessing (True).

Analysing the results in Table 2, first of all we observe that solving the problem considering balls as neighbourhoods is harder than considering segments. Next, we also observe that our formulation solves to optimality all instances for segments up to $|\mathcal{N}| = 30$ with preprocessing. In addition, both typologies of instances, up to $|\mathcal{N}| = 50$, are almost solved by obtaining a small average gap. We also observe that the use of preprocessing helps in solving the instances since in those cases where they are not solved to optimality the gap and the runtime are reduced. The situation for $|\mathcal{N}| \in \{80, 100\}$ is slightly different. For $|\mathcal{N}| = 80$ and balls as neighbourhoods, the preprocessing time takes more than an hour so that no results can be reported whereas without preprocessing the formulations are always able to find solutions with average gaps of (≤ 0.6) within one hour of computing time. Finally, for the case of $|\mathcal{N}| = 100$ balls as neighbourhoods, none of the configurations with or without preprocessing is capable to find a feasible solution in 3600 seconds. The behaviour is different for segments. In this case, both configurations provide feasible solutions. In the last case, the use of preprocessing is rather heavy and consumes most of the 3600 seconds of computing time. This can be observed in the final gaps. The case without preprocessing reports better gaps, as leaving more time to the solver to find better solutions, and the average gap is 0.53 rather than 0.81 for the case with preprocessing.

The same behaviour is also observed in Figure 9. The left subfigure reports the boxplot diagram for the experiments with segments as neighbourhoods, whereas the right one reports the boxplot diagram for balls as neighbourhoods. We see that preprocessing always helps since the corresponding boxes are below those without preprocessing, whenever the problems are solved well. Moreover, we also observe in those cases where the model solves the instances $|\mathcal{N}| \leq 50$, that the problem with balls is harder than with segments since the boxes in the left subfigure are below the ones in the right subfigure.

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 non-convex 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-TSPVN which could help in a simplified formulation of the problem. These questions are very interesting but beyond the scope of this paper. Needless to say, that we plan to continue its analysis in a follow-up paper.

Table 2: Table of computational results obtained with the formulation (H-TSPN)

$\|\mathcal{N}\|$	neighbourhood	preprocessing	$\|\mathcal{B}\|$		Gap			Runtime		
			min	max	mean	min	max	mean	min	max
5	Ball	False	8	11	0.05	0	0.55	371.84	2.96	3600
		True			0	0	0	3.46	1.4	13
	Segment	False			0.08	0	0.49	1085.48	0.68	3600
		True			0	0	0	1.44	0.6	2.46
10	Ball	False	15	21	0.17	0	0.86	998.71	59.09	3600
		True			0	0	0	36.66	10.72	87.29
	Segment	False			0.33	0	0.86	1460.31	16.78	3600
		True			0	0	0	8.53	2.66	12.25
20	Ball	False	35	40	0.06	0.02	0.1	3600	3600	3600
		True			0	0	0.01	1424.85	140.54	3600
	Segment	False			0	0	0.02	1430.13	166.89	3600
		True			0	0	0	198.9	30.75	426.55
30	Ball	False	49	60	0.12	0	0.93	3566.2	3246.53	3600
		True			0	0	0.02	2146.17	314.65	3600
	Segment	False			0	0	0.01	2119.06	315.56	3600
		True			0	0	0	1140.02	353.73	3209.94
50	Ball	False	79	91	0.09	0.02	0.19	3600	3600	3600
		True			0.08	0.01	0.21	3600	3600	3600
	Segment	False			0.02	0.01	0.09	3600	3600	3600
		True			0.02	0	0.1	2956.24	2053.62	3600
80	Ball	False	125	145	0.6	0.42	0.79	3600	3600	3600
		True			-	-	-	-	-	-
	Segment	False			0.16	0.03	0.62	3600	3600	3600
		True			0.61	0.01	1	3600	3600	3600
100	Ball	False	156	178	-	-	-	-	-	-
		True			-	-	-	-	-	-
	Segment	False			0.53	0.02	0.99	3600	3600	3600
		True			0.81	0.5	1	3600	3600	3600

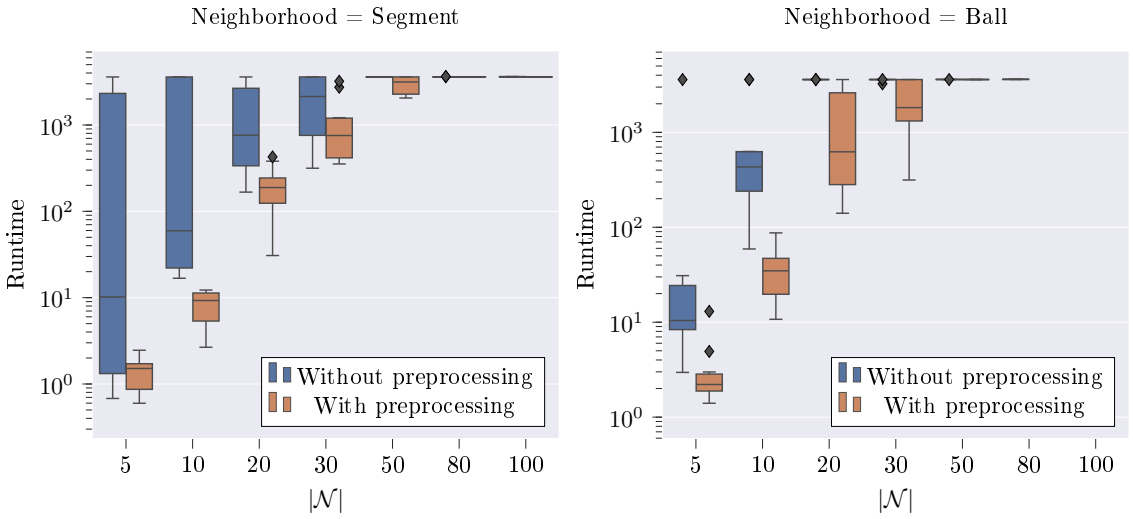


Figure 9: Runtime of the model (H-TSPN) without and with preprocessing when the neighborhoods are segments and balls.

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 location 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.

Acknowledgements

This research has been partially supported by the Agencia Estatal de Investigación (AEI) and the European Regional Development Fund (ERDF): PID2020-114594GB-C21; Regional Government of Andalusia: projects FEDER-US-1256951, P18-FR-1422; and Fundación BBVA: project NetmeetData (Ayudas Fundación BBVA a equipos de investigación científica 2019)

References

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

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. Publisher: Elsevier BV.

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

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

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

Glock, K. and Meyer, A. (2022). Spatial coverage in routing and path planning problems. *European Journal of Operational Research*.

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

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

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

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

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. *Linear Algebra and its Applications*, 284(1):193–228.

Möemke, T. and Svensson, O. (2011). Approximating Graphic TSP by Matchings. pages 560–569. IEEE Computer Society. ISSN: 0272-5428.

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

Perez, D., Powley, E. J., Whitehouse, D., Rohlfshagen, P., Samothrakis, S., Cowling, P. I., and Lucas, S. M. (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. Conference Name: IEEE Transactions on Computational Intelligence and AI in Games.

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

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