

An Exact Approach for Solving Pickup-and-Delivery Traveling Salesman Problems with Neighborhoods

Cai Gao

Industrial and Systems Engineering Department, University at Buffalo

Ningji Wei

Heinz College of Information Systems and Public Policy, Carnegie Mellon University

Jose L. Walteros *

Industrial and Systems Engineering Department, University at Buffalo

Abstract

This paper studies a variant of the traveling salesman problem called the pickup-and-delivery traveling salesman problem with neighborhoods that combines traditional pickup and delivery requirements with the flexibility of visiting the customers at locations within compact neighborhoods of arbitrary shape. We derive two optimality conditions for the problem, a local condition that verifies whether a given tour is locally optimal at the visiting points and a global condition that can be used to cut off sub-optimal regions of the neighborhoods. We model the problem as a mixed-integer nonlinear program and propose a generalized Benders decomposition to solve instances of the problem with convex neighborhoods. Finally, we conduct an extensive set of computational experiments to demonstrate the efficacy of the proposed solution framework.

Keywords: pickup-and-delivery, traveling salesman problem, mixed-integer nonlinear program, close-enough traveling salesman problem, generalized Benders decomposition.

1 Introduction

Pickup-and-delivery problems refer to a class of vehicle routing problems in which vehicles are used to transport commodities or passengers from a set of origins to a set of destinations on a given transportation network (Toth and Vigo, 2002, Battarra et al., 2014). Several variations of these problems have been extensively studied over the years, addressing applications in areas such as transportation, disaster relief, and communications, among others (Berbeglia et al., 2007, 2010, Kalantari et al., 1985, Parragh et al., 2008, Ropke and Pisinger, 2006, Savelsbergh and Sol, 1995, Xu et al., 2003).

When only a single vehicle is available to conduct the transportation tasks, the given problem is commonly referred to as the *pickup-and-delivery traveling salesman problem* (PDTSP) (Kalantari et al., 1985), and, contrary to the classic TSP where the decision-maker only has to minimize the distance the vehicle must

*Corresponding author. Phone: 716-645-8876; Fax: 716-645-3302; Address: 413 Bell Hall, Buffalo, NY 14260; Email: josewalt@buffalo.edu

travel, the PDTSP also requires considering additional constraints posed by the relationship between the pickup and delivery orders (Gendreau et al., 1999, Hernández-Pérez and Salazar-González, 2004a, Kalantari et al., 1985, Renaud et al., 2000).

In this study, we focus on a generalization of PDTSP called the *pickup-and-delivery traveling salesman problem with neighborhoods* (PDTSPN), where each customer is willing to meet with the vehicle at a point located within a certain neighborhood (i.e., a compact region embedded in \mathbb{R}^2). As such, instead of visiting each customer at some predefined location, the vehicle requires determining a shortest closed tour that intersects with each customer’s neighborhood at least once, while satisfying their corresponding pickup or delivery requirements. Considering that the neighborhoods may overlap, a vehicle following an optimal tour may then meet with multiple customers at the same location.

In practice, PDTSPN can be used to model routing problems where there is some degree of flexibility in the choice of meet-up locations. For example, one notable application arises when solving intelligence, surveillance, and reconnaissance problems in remote or denied areas (Wei et al., 2021). In these problems, a collection of sensors—often mounted on unmanned aerial vehicles (UAVs)—are sent to gather information from multiple targets. Without the support of a communication network to directly transfer the collected data or receive instructions from their base, the sensors depend on other UAVs (a.k.a, relays) that are routinely sent to follow collection routes to interact with them. Since both the sensors and relays can transfer information remotely at any point in the proximity of the sensors, the relay’s routing problem can be modeled as a PDTSPN.

Other applications in the context of UAV routing involve the transportation of critical medical supplies such as emergency medicines (Johnson et al., 2021), blood samples (Yakushiji et al., 2020), and defibrillators (Rosamond et al., 2020) to zones of difficult access. Moreover, similar applications not involving UAVs arise from problems related to the operation of wireless communication networks (Ciullo et al., 2010) and as part of optimization frameworks used by RFID remote meter-reading systems (Gulczynski et al., 2006).

1.1 Literature review

In this section, we review the relevant literature associated with PDTSPN. Because of the lack of papers on this problem, we focus our review on studies addressing its two main individual components. We begin our discussion with an overview of general developments on the PDTSP, followed by a survey of the literature that addresses the TSP with neighborhoods (TSPN).

1.1.1 PDTSP literature.

In general, most pickup-and-delivery problems partition the customers into two sets: the *demand customers*, which are the ones requesting commodities to be delivered, and the *supply customers*, which are the ones requiring commodities to be picked up. Furthermore, depending on the precedence relationship between the demand and supply customers, there are three main categories often used to classify most variations of PDTSP: the *one-to-many-to-one (1-M-1)*, the *many-to-many (M-M)*, and the *one-to-one (1-1)* (Toth and Vigo, 2002) problems. In the 1-M-1 version, all the commodities delivered to the demand customers must come directly from the depot, and all the commodities collected from the supply customers must be transported back to the depot. This problem is also called the TSP with pickups and deliveries in Mosheiov (1994) and the TSP with delivery and backhauls in Anily and Mosheiov (1994). In the M-M

version, there is only one type of commodity that is collected from the supply customers and delivered to the demand customers. This version is also known as the one-commodity PDTSP (Hernández-Pérez and Salazar-González, 2004a). Finally, in the 1-1 version, there is a one-to-one correspondence between the supply and the demand customers. i.e., the commodities collected from each supply customer are designated to be delivered to a specific demand customer. When the number of commodities to be transported between each pair of customers is a single unit, this problem is also called the *single vehicle many-to-many dial-a-ride problem* (Psaraftis, 1983). We now address specific developments related to these three variations.

To the best of our knowledge, Mosheiov (1994) was the first to study the 1-M-1 PDTSP. In this paper, the author proposed an approximation algorithm with a worst-case performance ratio of 2.5 that works under the assumption that the distance matrix is symmetric. Subsequently, Anily and Mosheiov (1994) improved this bound to 2 by using a minimum-spanning-tree-based approximation algorithm derived from the well-known Christofides heuristic for TSP (Christofides, 1976). It is important to note that, under some considerations, if the problem allows customers to have both types of orders (supply and demand), the optimal tour may visit some customers more than once. While this case is out of scope for this paper, we refer the interested reader to the work by Gribkovskaia et al. (2007), where each customer is allowed to be visited once for a combined pickup-delivery operation or twice for performing the two operations separately.

For the M-M PDTSP case, Hernández-Pérez and Salazar-González (2004a) were the first to tackle this problem, proposing a branch-and-cut algorithm able to solve instances with up to 40 customers to optimality. To solve larger instances within a reasonable time, Hernández-Pérez and Salazar-González (2004b) proposed two heuristics, one based on 2-*opt* and 3-*opt* local moves, and the other based on a branch-and-cut procedure. With these heuristics, the authors were able to solve instances with up to 500 customers. Since then, other heuristic and metaheuristic methods have been successfully applied to solve the problem in recent years (Mladenović et al., 2012, Hernández-Pérez et al., 2009, Zhao et al., 2009).

As for the 1-1 PDTSP version, Kalantari et al. (1985) were the first to solve this problem under the name *TSP with pickup and delivery* (TSPPD). The authors proposed an exact branch-and-bound algorithm able to solve instances with up to 31 customers. In recent years, Dumitrescu et al. (2010) analyzed the polyhedral structure of the problem and proposed a branch-and-cut algorithm with multiple cut separation subroutines.

In addition to the standard version of 1-1 PDTSP, there are other interesting variants worth mentioning: the PDTSP with last-in-first-out (LIFO) or first-in-first-out (FIFO) loading policies (Cordeau et al., 2010b,a), the PDTSP with handling costs (Veenstra et al., 2017, Qu and Bard, 2015), and the TSP with dynamic pickups and deliveries (O’Neil and Hoffman, 2017). Also, while our previous discussion was focused on the single-vehicle version of the problem (i.e., the TSP version), we note that other developments exist for solving the problem with multiple vehicles. We refer the interested reader to the surveys found in Toth and Vigo (2002).

1.1.2 TSPN literature.

The TSPN is a generalization of the classic TSP, where the nodes representing the customers are replaced with compact sets called neighborhoods or regions. To the best knowledge Arkin and Hassin (1994) were the first to introduce this problem with the name *geometric covering salesman problem*. In this study, the authors provide several $O(1)$ -approximation algorithms for cases where the neighborhoods have specific shapes, such as parallel line segments of equal lengths, or regions that are generated from the same type

of convex polygon. More recently, Dumitrescu and Mitchell (2003) extended and improved these results for neighborhoods with similar diameters, unit disks, and straight lines. In particular, the authors proposed a polynomial-time approximation scheme (PTAS) for the problem with neighborhoods that are disjoint disks with similar diameters. For TSPN with arbitrary compact shapes, Mata and Mitchell (1995) provided a framework that gives an $O(\log k)$ -approximation algorithm with time complexity $O(n^5)$, where k stands for the number of polygonal neighborhoods and n is the total number of vertices of all the polygons. More recently, Gudmundsson and Levcopoulos (1999) proposed an algorithm that improves such complexity to $O(n^2 \log n)$.

In some literature, TSPN is alternatively called the *close-enough traveling salesman problem* (CETSP) (Behdani and Smith, 2014, Gulczynski et al., 2006). In CETSP, the neighborhoods are often defined as disks in a two-dimensional space or closed balls in a three-dimensional space (Coutinho et al., 2016, Carrabs et al., 2017, Wang et al., 2019). This problem has been generalized for compact neighborhoods of arbitrary shapes in (Behdani and Smith, 2014), where the authors proposed two different discretization schemes to approximate any compact neighborhoods and developed three mixed-integer programming formulations based on these discretization schemes to provide lower bounds for the problem. Carrabs et al. (2017) further improved the lower and upper bounds by presenting a new discretization scheme, which is capable of solving instances with up to 30 neighborhoods. Coutinho et al. (2016) proposed an efficient exact approach based on a combinatorial branch-and-bound algorithm that uses a second-order conic program as a subroutine. In addition to exact approaches, several efficient heuristics have been developed to solve the problem as well (Gulczynski et al., 2006, Yuan et al., 2007, Mennell, 2009, Wang et al., 2019).

1.2 Contributions

Despite the broad interest regarding both the PDTSP and TSPN, little work has been devoted to studying PDTSPN, which considers the two fundamental components of the other two problems simultaneously. In this paper, we are interested in addressing this gap by studying exact methods for solving the PDTSPN. The main contributions of this paper follow.

1. We derive two optimality conditions for PDTSPN with compact neighborhoods that have arbitrary shapes. The first condition characterizes whether a given tour is locally optimal at the visiting point of each neighborhood. For the second condition, we identify a certain type of half-plane we called the boundary-projection-closed separator which contains all the “non-redundant” optimal tours. Then, we show that the intersection of all these separators, defined as the domination hull, can be used to cut off sub-optimal regions of the neighborhoods. We also develop a binary-search-based algorithm to approximate the domination hull efficiently.
2. We propose a mixed-integer nonlinear programming (MINLP) formulation to solve PDTSPN instances with convex neighborhoods and introduce a cut-generation implementation that produces exact optimal solutions. Our approach is based on a generalized Benders decomposition that iterates between two models, a master problem that identifies the feasible visiting order of the tour and a subproblem that computes the optimal visiting locations of the tour given the visiting order provided by the master and generates optimality Benders cuts. We show how to adapt our approach to tackle instances of PDTSPN with several supply-demand precedence relationships including the 1-1, 1-M-1, and M-M

versions.

3. We conduct an extensive set of computational experiments to test the proposed solution approach on a wide variety of instances. We analyze the performance of our approach and discuss some observations and possible adaptations.

The rest of the paper is organized as follows. In Section 2, we give an exact account of the problem setting and assumptions. In Section 3, we study two optimality conditions for PDTSPN, which can also be implemented to solve other variants of TSPNs with general shapes. In Section 4, we propose a mixed-integer nonlinear programming formulation to solve PDTSPN. In Section 5, we conduct a battery of computational experiments to test our solution approach. Finally, in Section 6, we conclude the paper and offer further insights.

2 Problem Description

For the sake of simplicity, we limit the following exposition to the M-M version of PDTSPN. However, in Section 4, we will describe how to apply the proposed framework to solve instances with other types of precedence relationships.

2.1 Preliminaries

Let $I_0 = \{0, 1, \dots, n\}$ be an index set and $\mathcal{U} = \{U_i\}_{i \in I_0}$ be a given collection of compact sets in \mathbb{R}^2 that may overlap. In particular, U_0 is assumed to be a single point p_0 , which we call the depot. Assigned to each non-depot neighborhood U_i , there is either a positive supply of commodities s_i to be collected from or a positive demand of commodities d_i to be delivered to. We denote I_s and I_d the index sets for the supply and demand neighborhoods. For variations of PDTSPN other than the M-M, if there is a neighborhood that has both supply and demand orders, we create two copies of the same neighborhood and add one index to both I_s and I_d representing each copy. This ensures that the pair of index sets I_s and I_d induces a partition of set $I = I_0 \setminus \{0\}$. A vehicle with capacity c is assumed to be dispatched from the depot U_0 to collect or deliver the commodities to each neighborhood.

For the following presentation, we first require to adapt some common TSP concepts. In particular, since traditional TSP problems are defined over graphs where each customer is characterized by a node, every feasible tour is represented by a Hamiltonian cycle on the graph. On the other hand, since PDTSPN is defined over compact area in \mathbb{R}^2 , where feasible tours may traverse each neighborhood passing by multiple points, this concept must be adapted accordingly.

Definition 1 (Neighborhood Visit). *A visit to a neighborhood U_i is assumed to occur if the vehicle enters the neighborhood and either collects or delivers a positive amount of commodities at a certain point $p_i \in U_i$.*

Thus, a vehicle passing by a neighborhood without collecting or delivering any commodity is not considered a visit.

Definition 2 (Hamiltonian PDTSPN Tour). *Given a collection of neighborhoods $\mathcal{U} = \{U_i\}_{i \in I_0}$ with positive supplies or demands, a PDTSPN tour is assumed to be Hamiltonian if it is a closed tour that starts at the depot U_0 and visits each non-depot neighborhood exactly once.*

Under this settings, the *pickup-and-delivery traveling salesman problem with neighborhoods* (PDTSPN) seeks for a shortest Hamiltonian tour that satisfies all the supply and demand requirements (see Figure 1 for an example).

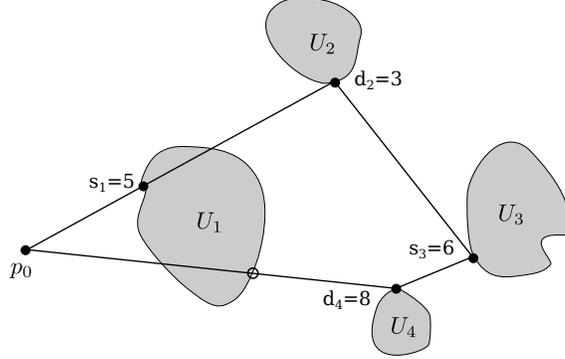


Figure 1: A M-M PDTSPN instance with $|I| = 4$ neighborhoods and one depot. A feasible tour is constructed by selecting a point (solid black dot) from each neighborhood. In this solution, the vehicle first collects 5 units at neighborhood U_1 , delivers 3 units to U_2 , continues to pick up 6 units at U_3 , then delivers all 8 units to U_4 , before returning to the depot p_0 . Notice that the hollow dot at the boundary of U_1 along with all the subsequent points that intersect that neighborhood are not counted as visits to U_i .

Throughout the paper, we have the following assumption.

Assumption 1. A PDTSPN instance satisfies the following conditions:

- (i) The total supply is equal to the total demand: $\sum_{i \in I_s} s_i = \sum_{i \in I_d} d_i$.
- (ii) The vehicle capacity c is greater or equal to the total supply $\sum_{i \in I_s} s_i$.

Condition (i) ensures that the supply and demand requirements can be fully satisfied. This can be assumed without loss of generality because the depot U_0 can always be used to adjust the deficit or surplus of commodities. Condition (ii) ensures that there exists a tour that visits each neighborhood at most once. Both conditions are sufficient to guarantee the existence of an optimal Hamiltonian tour (see Proposition 1).

2.2 Solution Characterization

Without Assumption 1, the optimal solution of PDTSPN may not be a Hamiltonian. This can occur, for instance, if capacity c is strictly less than s_i for some $i \in I$. In such a case, the vehicle has to visit neighborhood i more than once to satisfy that supply requirement. Moreover, notice that, even under the assumption that $c \geq s_i$ for all $i \in I_s$ and $c \geq d_i$ for all $i \in I_d$, it is possible to construct instances that require visiting at least one neighborhood more than once. Consider, for example, an instance with three demand customers, all with $d_i = 5$, five supply customers, all with $s_i = 3$, and a vehicle with capacity $c = 5$. In such a case, it is easy to see that the vehicle cannot satisfy the supply-demand requirements by visiting all the neighborhoods exactly once.

The following proposition shows that Assumption 1 ensures there is always a Hamiltonian solution to the problem. While the result of this proposition is quite intuitive, we provide the proof for the sake of

completeness. Furthermore, the proof is given for the M-M case; however, we note that a sufficient capacity on the vehicle always leads to a Hamiltonian optimal tours even for other versions, as visiting the same location multiple times only lengthens the total distance.

Proposition 1. *A PDTSPN instance that satisfies Assumption 1 admits an optimal tour that visits each neighborhood exactly once. Moreover, such tour is a shortest Hamiltonian tour where the visits take place at some set of points $\{p_i\}_{i \in I}$, where $p_i \in U_i$ for all $i \in I$.*

Proof. Given the problem's setting, each neighborhood except U_0 has either $d_i > 0$ or $s_i > 0$. Hence, to satisfy all the pickup and delivery requirements, the vehicle needs to visit each neighborhood at least once. For the first claim, all is left to show is that there exists an optimal tour that visits each neighborhood at most once. Notice that the problem is feasible by the first statement in Assumption 1 and the optimality is always obtainable due to the compactness of all the neighborhoods. So, we can start with an arbitrarily selected optimal tour γ that is represented by the sequence $(p_0, p_1, p_2, \dots, p_n)$, i.e., a closed tour that starts and ends at p_0 and traverses from p_1 to p_n in between, where every point p_k belongs to some neighborhood U_i that may have been visited more than once. We call the number of the visits to a neighborhood U_i in this sequence the multiplicity of U_i . If the multiplicity of every neighborhood is one in γ , we are done. Now, suppose otherwise, then the following procedure will reduce by one the multiplicity of any neighborhood that has been visited repeatedly in γ . By applying this process as many times as necessary, one can produce an optimal tour that is Hamiltonian, thus proving the claim.

The procedure is as follows. If neighborhood that has been visited multiple times, say neighborhood U_i , is a demand neighborhood, then we use p_k and $p_{k'}$ to denote the points in γ that visit U_i the first two times, and let l_k and $l_{k'}$ be the corresponding delivered commodities during each visit. Then, we split the tour γ into several subpaths as

$$\gamma = (p_0 \rightsquigarrow p_{k-1} \rightarrow p_k \rightarrow p_{k+1} \rightsquigarrow p_{k'} \rightsquigarrow p_n),$$

where “ \rightarrow ” means no other neighborhoods are visited in between and “ \rightsquigarrow ” indicates that potentially multiple neighborhoods are visited in between. We construct the following new tour

$$\gamma' = (p_0 \rightsquigarrow p_{k-1} \rightarrow p_{k+1} \rightsquigarrow p_{k'} \rightsquigarrow p_n).$$

Let v, v' be two vehicles following tours γ and γ' , respectively. Comparing with v , vehicle v' will only take two different actions in tour γ' : it skips the stop p_k and it drops $l_k + l_{k'}$ commodities at point $p_{k'}$. By this design, before the point p_{k-1} and after the point $p_{k'}$, the two vehicles are at the same state in their routes, i.e., at each fixed time, they are at the same location with the same amount of commodities. For the tour between p_{k-1} and $p_{k'}$, it is obvious that the second tour is no longer than the first tour (in fact, they have the same length since γ is assumed to be optimal). We only need to show this new construction is possible between p_{k+1} and $p_{k'}$. Notice at the time of reaching p_{k+1} , vehicle v holds strictly less commodities than vehicle v' . Along with the second statement in Assumption 1, it is clear that v' can always replicate the actions that v takes in the subpath between p_{k+1} and $p_{k'}$. Thus, we have reduced the multiplicity of the neighborhood U_i by one in this case. A similar procedure and argument can be applied to the case that the repeatedly visited neighborhood is a supply neighborhood. So, we proved the first claim.

Therefore, we can focus on all the tours that visit each neighborhood exactly once. That consists of all the closed tours that pass through some point set $\{p_i\}_{i \in I}$ with $p_i \in U_i$ for all $i \in I$. Now, because the

shortest closed tour for each such fixed point set $\{p_i\}_{i \in I}$ is a Hamiltonian tour, so is the global optimal tour among all the possible point sets. \square

According to Proposition 1, we only need to focus on tours that are Hamiltonian. Let S_n be the symmetric group that contains all the permutations of the index set I , then each $\sigma \in S_n$ indicates a unique visiting order of the neighborhoods. Thus, any Hamiltonian tour that conforms with the visiting order σ can be represented by a sequence of points (p_0, \mathbf{p}) for some $\mathbf{p} \in \prod_{i \in I} U_{\sigma(i)}$, where $U_{\sigma(i)}$ is the i th visited neighborhood under σ . This representation of tours allows us to model the problem as follows:

$$\min_{\sigma \in S_n} \min_{\mathbf{p} \in \prod_{i \in I} U_{\sigma(i)}} HT(p_0, \mathbf{p}) \quad (1a)$$

$$s.t. \quad \sum_{j \leq i} s_{\sigma(j)} \geq \sum_{j \leq i} d_{\sigma(j)}, \quad \forall i \in I, \quad (1b)$$

where $HT(\cdot)$ denotes the length of the Hamiltonian tour for the visiting order given by σ . Constraint (1b) enforces that, for each subpath $p_0 \rightsquigarrow p_i$, for all $i \in I$, the amount of supplies carried by the vehicle is sufficient to satisfy all the demands up to neighborhood i . We also note that constraints (1b) only affect the visiting order of permutation σ and do not restrict selection of visiting points \mathbf{p} . This implies that selection of such points is only handled by the inner level, independently of the supplies and demands. Hence, any new developments for solving the inner problem can be used to solve other variations of TSPN as well. The following section is devoted to studying some optimality conditions for the general TSPN.

3 Optimality Conditions for TSPN Solutions

The general TSPN described by Formulation (1a) can also be expressed as follows

$$\min_{\sigma \in S_n, \{x_i\}_{i \in I}} \sum_{i \in I \setminus \{n\}} \delta(x_{\sigma(i)}, x_{\sigma(i+1)}) + \delta(p_0, x_{\sigma(1)}) + \delta(p_0, x_{\sigma(n)}) \quad (2a)$$

$$s.t. \quad f_i(x_i) \leq 0, \quad \forall i \in I, \quad (2b)$$

where function $\delta : \mathbb{R}^2 \times \mathbb{R}^2 \rightarrow [0, \infty)$ represents the Euclidean distance, and the functions in $\{f_i\}_{i \in I}$ define the neighborhoods. There are many choices of functions $\{f_i\}_{i \in I}$ to describe compact regions in \mathbb{R}^2 . For instance, they can be used to represent (a) the intersection of a set of half-planes, (b) ellipsoids, or (c) neighborhood enclosed in line segments connecting a sequence of finite points. Given a fixed visiting order σ , different neighborhood representations induce various types of formulations: (a) produces a quadratic program with linear constraints; (b) can be formulated as a second-order cone program; (c) can be solved by the second-order conic integer program. Instead of analyzing all these variations, we are interested in deriving optimality conditions for solutions of the general TSPN, which are invariant under the specific realizations of f 's. Therefore, the results that will be introduced in this section can be used to either produce valid constraints or improve subroutines for any TSPN implementation with various realizations of f 's.

To keep the generality, we assume that all neighborhoods are compact in this section. Notice that compact neighborhoods can still have complex topological properties. For instance, Figure 2 displays several types of compact neighborhoods. From left to right, we have a disconnected neighborhood, a disconnected

neighborhood with holes, a connected but non-convex neighborhood, and a convex neighborhood. To study neighborhoods with such general properties, we need to introduce some basic geometric objects first.

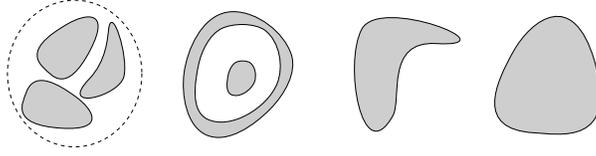


Figure 2: Neighborhoods with different topological properties.

3.1 Neighborhood-Related Objects

By our previous discussion, any optimal solution of TSPN must be a Hamiltonian tour (p_0, \mathbf{p}) for some $\mathbf{p} \in \prod_{i \in I} U_i$. One caveat of this representation is that it might contain repeated points, since the neighborhoods can overlap. Therefore, multiple visits to different neighborhoods may occur at the same location. To remove the symmetries that may arise from such repetitions, we instead use a reduced representation $(p_k)_{k=0}^K$ that contains only distinct points. This represents the closed tour with the shortest distance that starts at p_0 , then travels between each pair of consecutive points p_k and p_{k+1} , before returning to p_0 . We also define $\hat{U}_k := \{U_i \in \mathcal{U} \mid p_k \in U_i\}$ to be the set of *binding neighborhoods* at p_k . A sequence $(p_k)_{k=0}^K$ is feasible if $\{\hat{U}_k\}_{k=0}^K$ is a nonempty cover of \mathcal{U} , i.e., $\hat{U}_k \neq \emptyset$ for all k and their union is \mathcal{U} . A sequence is then said to be optimal if it has the shortest length among all feasible sequences.

Definition 3. Given a feasible sequence $\gamma = (p_k)_{k=0}^K$, we define the following associated objects:

- T_k is the convex combination of points $\{p_{k-1}, p_k, p_{k+1}\}$;
- $\theta_k \in [0, \pi]$ is the angle of T_k at the point p_k ;
- p_k is called a passing point if $\theta_k = \pi$, and a turning point otherwise;
- P_k is the path $p_{k-1} \rightarrow p_k \rightarrow p_{k+1}$;
- \bar{P}_k is the complement path of P_k in the closed tour γ , i.e.,

$$\bar{P}_k = p_{k+1} \rightarrow p_{k+2} \rightarrow \cdots \rightarrow p_0 \rightarrow p_1 \rightarrow \cdots \rightarrow p_{k-1}$$

- $\mathcal{U}_k := \{U_i \in \hat{U}_k \mid U_i \cap \bar{P}_k = \emptyset\}$ is called the set of active neighborhoods at p_k .

The following lemma reveals an interesting relationship between a turning point p_k and its active neighborhoods.

Lemma 1. If point p_k is a turning point in an optimal sequence γ^* , then the set of active neighborhoods \mathcal{U}_k is nonempty.

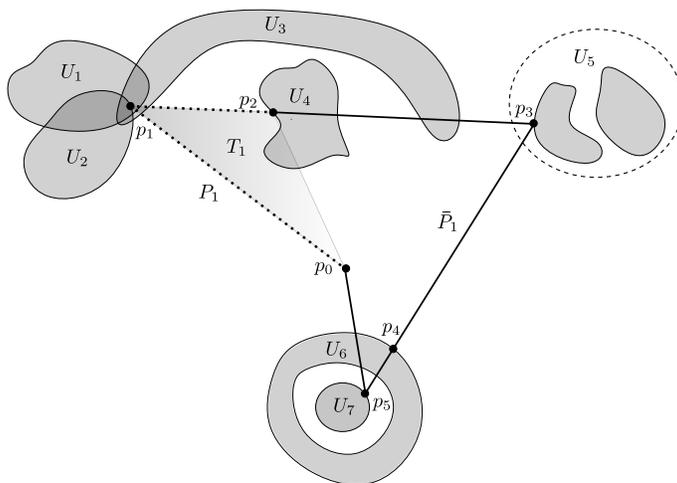


Figure 3: Neighborhood-related objects: T_1 is the shaded triangle (p_0, p_1, p_2) ; θ_1 is the angle of T_1 at p_1 ; p_2 and p_4 are passing points while the rest are turning points; the dotted path $p_0 \rightarrow p_1 \rightarrow p_2$ is P_1 ; the complement path denoted by the set of solid line segments is \bar{P}_1 ; $\mathcal{U}_1 = \{U_1, U_2\}$ is the set of active neighborhoods at p_1 . Notice that the binding neighborhood U_3 is not active since it intersects \bar{P}_1 .

Proof. $\mathcal{U}_k = \emptyset$ implies either that no neighborhood contains p_k or, if one does, say U_i , then U_i is also intersected by the complement path \bar{P}_k . Clearly, in both cases, the sequence $\gamma' = (p_0, \dots, p_{k-1}, p_{k+1}, \dots, p_n)$ also visits U_i . Moreover, p_k being a turning point implies that γ' is strictly shorter than the original optimal sequence, which leads to a contradiction. \square

Also, the following lemma will be used in the following sections. It can be trivially proven by sequentially applying the triangle inequality (see Figure 4), so we omit the proof.

Lemma 2. For any triangle $T = (a, b, c)$ where a, b, c are three points in \mathbb{R}^2 such that the angle at b is within the interval $[0, \pi)$, let b' be any point in T such that $b' \neq b$; then we have,

$$\delta(a, b) + \delta(b, c) > \delta(a, b') + \delta(b', c).$$

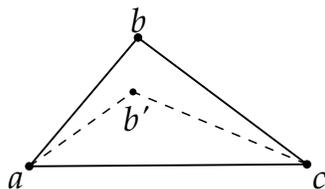


Figure 4: Illustration of Lemma 2.

3.2 Local conditions

When solving TSPN, Behdani and Smith (2014) proved that any optimal TSPN tour can be represented by an ordered set of turning points, each of which is located on the boundary of some neighborhood. In this section, we strengthen this conclusion by analyzing the relation between the objects associated with each turning point p_k . The main theorem follows.

Theorem 1. *If p_k is a turning point in an optimal sequence γ^* , then there must exist a neighborhood $U_i \in \mathcal{U}_k$ such that $U_i \cap P_k = \{p_k\}$. Moreover, if all neighborhoods in \mathcal{U}_k are also connected, then there is some $U_i \in \mathcal{U}_k$ such that $U_i \cap T_k = \{p_k\}$.*

Proof. We prove the first statement by contradiction. Let p_k be a turning point such that all neighborhoods $U_i \in \mathcal{U}_k$ intersect with P_k at more points than just p_k . Then, we will construct a strictly shorter feasible sequence γ' that still visits all the neighborhoods, which contradicts the optimality of γ^* . Let p_{k-1} and p_{k+1} be the previous and next points of p_k in γ^* , and let $[p_{k-1}, p_k)$ and $(p_k, p_{k+1}]$ represent the half open line segment between the two endpoints. For each $U_i \in \mathcal{U}_k$, we define two sets of points

$$L_i = \arg \max_{p \in U_i \cap [p_{k-1}, p_k)} \delta(p, p_k),$$

$$R_i = \arg \max_{p \in U_i \cap (p_k, p_{k+1}]} \delta(p, p_k).$$

Thus, L_i (or R_i) contains the points in the intersection of U_i and the segment $[p_{k-1}, p_k)$ (or $(p_k, p_{k+1}]$) that is the farthest to point p_k . Notice that, L_i (or R_i) is either a singleton or an empty set, but $L_i \cup R_i$ must be nonempty by the choice of U_i . Then, we define the following two sets

$$L = \arg \min_{p \in \bigcup_{i \in I_k} L_i} \delta(p, p_k),$$

$$R = \arg \min_{p \in \bigcup_{i \in I_k} R_i} \delta(p, p_k),$$

where I_k is the index set of the neighborhoods in \mathcal{U}_k . Thus, L (or R) contains the points that are closest to p_k among all the points in the union $\bigcup_{i \in I_k} L_i$ (or $\bigcup_{i \in I_k} R_i$). By the assumption on all U_i 's, $L \cup R \neq \emptyset$. Thus, we are left with three possible cases. In the first case, we have $L = \emptyset$ and $R \neq \emptyset$. This means each $U_i \in \mathcal{U}_k$ intersects some point in $(p_k, p_{k+1}]$. Take any $r \in R$, we construct a feasible sequence $\gamma' = (p_0, \dots, p_{k-1}, r, p_{k+1}, \dots, p_n)$, which replaces p_k with r . In the second case, $L \neq \emptyset$ but $R = \emptyset$. Similarly, we construct γ' by replacing p_k with l for any $l \in L$. In the last case, both L and R are nonempty. Then, we define $\gamma' = (p_0, \dots, p_{k-1}, l, r, p_{k+1}, \dots, p_n)$ for $l \in L$ and $r \in R$. In all three cases, it is easy to check that the new sequence γ' covers all the neighborhoods in \mathcal{U}_k by the construction and is strictly shorter than γ^* by Lemma 2, which contradicts that γ^* is optimal. Hence, we proved the first claim.

For the second statement, suppose every $U_i \in \mathcal{U}_k$ intersects the triangle T_k at some point $q_i \neq p_k$. Then, we can partition the neighborhoods in \mathcal{U}_k into two groups \mathcal{U}_k^1 and \mathcal{U}_k^2 . The former contains neighborhoods that have a path $q_i \rightsquigarrow p_k$ inside the inner region enclosed by the closed curve γ^* ; the latter includes all the rest neighborhoods, each of which must have a path $q_i \rightsquigarrow p_k$ intersects the path P_k , because, otherwise, they must have a path $q_i \rightsquigarrow p_k$ intersects the complement path \bar{P}_k , which makes U_i non-active by Definition 3, a contradiction. Notice that \mathcal{U}_k^1 is nonempty, otherwise, all neighborhoods in \mathcal{U}_k intersect P_k , which contradicts

the first statement we have just shown. For each $U_i \in \mathcal{U}_k^1$, let ω_i be a path connecting q_i to p_k and is inside the inner region enclosed by γ^* , and let C_ϵ be a circle of radius ϵ centered at p_k . Then, for a sufficiently small scalar $\epsilon > 0$, the circle C_ϵ intersects all the paths ω_i 's as well as the half open line segments $[p_{k-1}, p_k)$ and $(p_k, p_{k+1}]$. We use l and r to denote the intersection points of C_ϵ with $[p_{k-1}, p_k)$ and $(p_k, p_{k+1}]$, respectively. Then, the new sequence $\gamma' = (p_0, \dots, p_{k-1}, l, r, p_{k+1}, \dots, p_n)$ visits all the neighborhoods in \mathcal{U}_k^1 since the line segment $[l, r]$ intersects all the paths ω_i 's. Moreover, for a sufficiently small $\epsilon > 0$, γ' also intersects all neighborhoods in \mathcal{U}_k^2 , since the intersection points of each neighborhood $U_i \in \mathcal{U}_k^2$ and the path P_k have a fixed positive distance to p_k . Furthermore, the new curve γ' is strictly shorter than the original curve γ^* by the construction and the triangle inequality. This contradicts to the optimality of γ^* . So, we are done. \square

Notice that both $U_i \cap P_k = \{p_k\}$ and $U_i \cap T_k = \{p_k\}$ imply that $\{p_k\}$ is on the boundary of U_i . Thus, this theorem establishes a stronger optimality criterion than the one in Behdani and Smith (2014) by using the relation between some active binding neighborhood $U_i \in \mathcal{U}_k$ and the associated path P_k or triangle T_k . The extra connectedness requirement in the second statement of Theorem 1 is necessary due to the possibility that multiple neighborhoods may be binding at the same turning point. If all neighborhoods are known to be disjoint, the following corollary provides a similar result without requiring connectedness.

Corollary 1. *Let p_k be any turning point of an optimal sequence γ^* and \mathcal{U}_k be the associated active neighborhoods, if all neighborhoods in \mathcal{U}_k are disjoint, then the unique choice $U_i \in \mathcal{U}_k$ satisfies $U_i \cap T_k = \{p_k\}$ for all k .*

Proof. Because all neighborhoods are disjoint, each turning point can belong to exactly one U_i . This shows that \mathcal{U}_k contains only one neighborhood. By the first statement of Theorem 1, $U_i \cap P_k = \{p_k\}$. Suppose U_i also intersects T_k at some point $q_i \neq p_k$. By Lemma 2, replacing q_i with p_k in γ^* produces a strictly shorter feasible sequence, which is a contradiction. \square

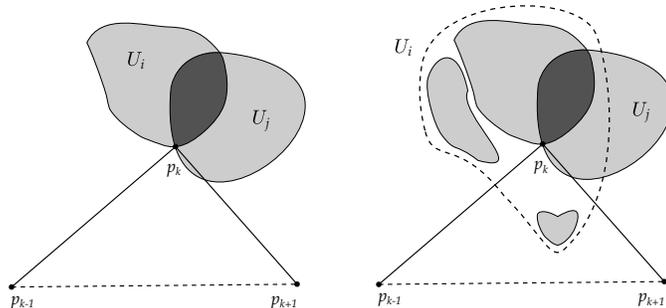


Figure 5: Illustration of the local optimality criteria for connected and disconnected neighborhoods.

Both Theorem 1 and Corollary 1 give local conditions associated with each of the turning points in an optimal sequence γ^* . Thus, we call them the local conditions. Furthermore, the proofs of both statements essentially provide constructive procedures to produce a strictly shorter feasible sequence from a given sequence that violates this criteria. Therefore, these results can be utilized to generate better feasible sequences in subroutines of various TSPN algorithms.

3.3 Global Conditions

The local conditions derived previously describe the relation between each turning point p_k of an optimal TSPN tour γ^* and the associated path P_k or triangle T_k . In contrast, the global conditions that will be presented in this section characterize the location of the entire optimal tour γ^* . Assuming that all neighborhoods are disks, Behdani and Smith (2014) have shown that the optimal tour γ^* must lie within the convex hull induced by all the centers of the disks. In this subsection, we generalize this result to compact neighborhoods with arbitrary shapes. We will focus only on neighborhoods in the two-dimensional Euclidean space, but most results can be easily extended to higher dimensions.

We start with some notation. Let ∂U be the boundary of U (i.e., the closure of U minus its interior under Euclidean topology). In particular, for some half-plane $H = \{x \in \mathbb{R}^2 \mid a^T x \leq b\}$, the boundary ∂H is the straight line $\{x \in \mathbb{R}^2 \mid a^T x = b\}$. For a half-plane $H \subseteq \mathbb{R}^2$ and a point $x \in \mathbb{R}^2$, we use $\text{proj}_H(x)$ to denote the projection of x onto the straight line ∂H . Then, we define the following,

Definition 4 (Boundary-projection-closed separators). *Given a compact neighborhood U and a half-plane H , U is boundary-projection-closed (BPC) with respect to H if for every point $x \in \partial U \setminus H$, $\text{proj}_H(x) \in U$. We call this half-plane H a BPC separator of U . Similarly, given a set of neighborhoods \mathcal{U} , a half-plane H is a BPC separator of \mathcal{U} if it is a BPC separator of every U in \mathcal{U} . We use $\mathcal{H}_{\mathcal{U}}$ to denote the set of all the BPC separators of \mathcal{U} .*

This definition uses a set of compact neighborhoods \mathcal{U} to characterize a certain type of half-plane. Figure 6(a) provides an example. By definition, both half-planes H_1 and H_3 are BPC separators, but H_2 is not because the projection of some boundary point is not contained in the same neighborhood. Notice that, for any family of compact neighborhoods \mathcal{U} , $\mathcal{H}_{\mathcal{U}}$ is nonempty as any half-plane that contains all the neighborhoods in \mathcal{U} is trivially a BPC separator of \mathcal{U} (e.g., half-plane H_3).

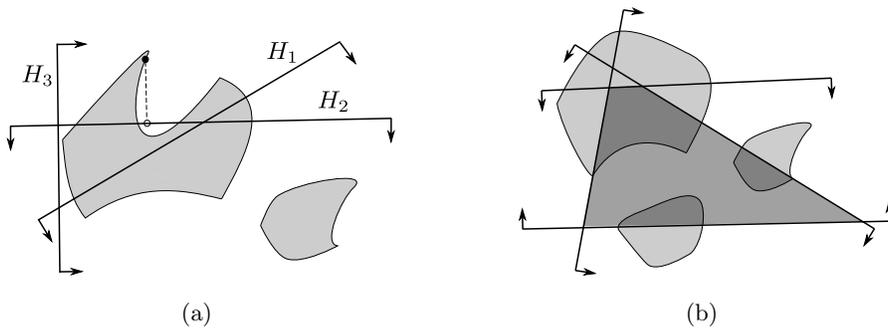


Figure 6: Illustration of projection-closed separators.

Given a fixed visiting order σ , we use γ_σ^* to denote a local optimal TSPN tour that conforms to the visiting order σ . Then, the next theorem describes the relation between a BPC separator H and any local optimal TSPN solution γ_σ^* . We first require to introduce the following definition.

Definition 5 (Redundant Tours). *Given a set of neighborhoods \mathcal{U} and a BPC separator H of \mathcal{U} , we say a feasible tour γ is redundant relative to H if $\gamma \cap H = \emptyset$ and γ lies in a straight line that is parallel to ∂H . We say γ is non-redundant if it is not redundant relative to any BPC separator H .*

A redundant tour has the property that its projection onto ∂H forms a non-redundant tour with the same length. Moreover, this non-redundant tour is still feasible as H is a BPC separator.

Theorem 2. *Given a set of neighborhoods \mathcal{U} , a BPC separator H of \mathcal{U} , and a visiting sequence σ , every non-redundant local optimal tour γ_σ^* is inside the half-plane H .*

Proof. Let $\mathbf{p} = (p_i)_{i \in I_0}$ be the local optimal curve γ_σ^* , where $p_i \in U_i$ for each $i \in I_0$ and p_i is either a turning or a passing point. Without loss of generality, we assume that the index set I_0 is sorted by the visiting order σ . We use $\hat{\mathbf{p}}$ to denote the corresponding point set $\{p_i\}_{i \in I_0}$. Clearly, if $\hat{\mathbf{p}} \subseteq H$, then the entire tour γ_σ^* also lies within the half-plane H . Suppose otherwise that $\hat{\mathbf{p}} \not\subseteq H$. Then, there are two possible cases: either all or part of the points in $\hat{\mathbf{p}}$ lie in the complement of H .

In the former case, we claim that the projected tour $\text{proj}_H(\mathbf{p}) = (\text{proj}_H(p_i))_{i \in I_0}$ visits every neighborhood U_i in the order of σ and is strictly shorter than γ_σ^* , which induces a contradiction. By assumption, γ_σ^* does not lie in a straight line that parallels to ∂H , which implies the projection of γ_σ^* onto ∂H is strictly shorter than γ_σ^* . It is left to show that $\text{proj}_H(p_i) \in U_i$ for each $i \in I_0$. If the point p_i is on the boundary of U_i , then $\text{proj}_H(p_i) \in U_i$ is true due to H is a BPC separator of U_i . Otherwise, p_i is an interior point of U_i . Then, the line extended from the line segment $[\text{proj}_H(p_i), p_i]$ intersects U_i at some point $p'_i \in \partial U_i \setminus H$. Then, we have

$$\text{proj}_H(p_i) = \text{proj}_H(p'_i) \in U_i,$$

where the equality is by the construction of p'_i and the membership relation is due to H is a BPC separator of U_i . This concludes the first case.

In the latter case, some point $p_0 \in \hat{\mathbf{p}}$ is in the half-plane H . Then, we represent γ_σ^* as a continuous function from the closed interval $[0, 1]$ to \mathbb{R}^2 such that $\gamma_\sigma^*(0) = \gamma_\sigma^*(1) = p_0$, and $\gamma_\sigma^*(t_i) = p_i$ for all $i \in I_0$. By assumption, there is also some point in $\hat{\mathbf{p}}$ that is not in H , i.e., $\gamma_\sigma^*(t_i) \notin H$ for some $i \in I$, where I is the index set $I_0 \setminus \{0\}$. The fact that γ_σ^* is a continuous function means there exists some closed neighborhood $T = [t_{\min}, t_{\max}] \subseteq [0, 1]$ of t_i such that the path $\gamma_\sigma^*(T)$ intersects H only at the two endpoints $p_{\min} = \gamma_\sigma^*(t_{\min})$ and $p_{\max} = \gamma_\sigma^*(t_{\max})$. Let $T' = (t_{\min}, t_{\max})$ be the corresponding open interval of T , and we use the set $I' = \{i \in I \mid t_i \in T'\}$ to label all the neighborhoods that are visited by the path $\gamma_\sigma^*(T')$. Then, we consider the following projected sequence

$$\text{proj}_H(\gamma_\sigma^*(T)) = (p_{\min}, (\text{proj}_H(p_i))_{i \in I'}, p_{\max}).$$

This sequence represents a subtour that visits all the neighborhoods $\{U_i\}_{i \in I'}$ with the same visiting order since H is a BPC separator of \mathcal{U} . Also, $\text{proj}_H(\gamma_\sigma^*(T))$ is strictly shorter than the original segment $\gamma_\sigma^*(T)$ since the tour is non-redundant. Moreover, $\text{proj}_H(\gamma_\sigma^*(T))$ and $\gamma_\sigma^*(T)$ shares the same endpoints. Thus, replacing the subtour $\gamma_\sigma^*(T)$ in γ_σ^* with the projected sequence produces a strictly shorter feasible solution, which yields a contradiction.

Both cases lead to contradictions, which implies the desired result. \square

A direct implication of Theorem 2 is that every non-redundant local optimal tour must be inside the intersection of any family of BPC separators of \mathcal{U} . For instance, in Figure 6(b), any local optimal tour cannot lie in a straight line. Thus, it is inside the convex hull enclosed by the four BPC separators. We define *domination hull* as the smallest intersection that contains all the non-redundant local optimal tours.

Definition 6 (Domination Hull). *Given a set of compact neighborhoods $\mathcal{U} = \{U_i\}_{i \in I_0}$, we define the following recursive construction procedure,*

1. *Step 1: initially, let $U_i^0 = U_i$ for all $i \in I_0$;*
2. *Step 2: $\mathcal{U}_k = \{U_i^k\}_{i \in I_0}$;*
3. *Step 3: $D_{\mathcal{U}_k} = \bigcap \mathcal{H}_{\mathcal{U}_k}$, where $\mathcal{H}_{\mathcal{U}_k}$ contains all the BPC separators of the neighborhoods \mathcal{U}_k ;*
4. *Step 4: $U_i^{k+1} = U_i^k \cap D_{\mathcal{U}_k}$ for all $i \in I_0$. Go to Step 2.*

Then, the domination hull $D_{\mathcal{U}}$ is defined as the limit of $D_{\mathcal{U}_k}$, i.e., $D_{\mathcal{U}} = \lim_{k \rightarrow \infty} D_{\mathcal{U}_k}$.

Essentially, the domination hull $D_{\mathcal{U}}$ is obtained by sequentially reducing all the neighborhoods using all the available BPC separators until no reduction can be further made. The reason for multiple iterations is that new BPC separators may emerge after each intersection operation (see Figure 7 for an example).

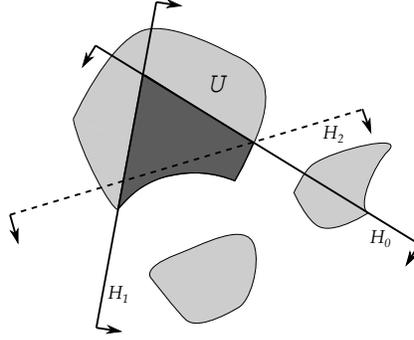


Figure 7: A new BPC separator H_2 that emerges after adding the two separators H_0 and H_1 .

The following corollary characterizes the domination hull $D_{\mathcal{U}}$ for any neighborhood set \mathcal{U} .

Corollary 2. *Given a set of compact neighborhoods \mathcal{U} , if the optimal tour γ^* does not lie in a straight line, then the domination hull $D_{\mathcal{U}}$ is a nonempty, convex, compact set.*

Proof. By Theorem 2, given that some optimal curve γ^* is non-redundant, then γ^* lies in $D_{\mathcal{U}}$, which shows $D_{\mathcal{U}}$ is nonempty. $D_{\mathcal{U}}$ is convex and closed since it is the intersection of a set of half-planes. Finally, by definition, a sufficiently large polygon that contains all the neighborhoods can be produced by the intersection of a set of BPC separators. Thus, it also contains $D_{\mathcal{U}}$, which means $D_{\mathcal{U}}$ is bounded, which entails compactness. \square

3.3.1 Domination Hull Approximation.

In general, constructing the exact domination hull is a non-trivial task, as it may require an infinite number of iterations of the procedure given in Definition 6 to fully converge. However, we can approximate it by stopping this procedure at any time we choose as long as a bounded polygon $D'_{\mathcal{U}}$ is obtained. Then, all the non-redundant local optimal tours, including the global optimal solution, are still inside $D'_{\mathcal{U}}$ according to Theorem 2.

Algorithm 1 presents a pseudocode for constructing an approximation of the domination hull. This algorithm requires some preprocessing steps. First, we compute a sufficiently small circle centered at O with a radius r that circumscribes all the neighborhoods in \mathcal{U} where O is the origin of the plane. Then, for a given angle $\theta \in [0, 2\pi)$ and a distance d , we use $H_{\theta,d}$ to denote the unique half-plane that satisfies: (1) $O \in H_{\theta,d}$, (2) the angle between the line segment $[O, \text{proj}_{H_{\theta,d}}(O)]$ and the x -axis is θ , and (3) the length $\delta(O, \text{proj}_{H_{\theta,d}}(O))$ is d .

Algorithm 1: Domination Hull Approximation

input : a set of neighborhoods \mathcal{U}
output: a set of BPC separators

```

1 set the number of iterations  $N$ ;
2  $\mathcal{H} \leftarrow \emptyset$ ; // the set of BPC separators
3 for  $i \in N$  do
4    $\theta \leftarrow 2\pi(i-1)/N$ ;
5    $d \leftarrow \text{BinarySearch}(0, r, \theta, \mathcal{U})$ ; // search for  $d$  such that  $H_{\theta,d} \in \mathcal{H}_{\mathcal{U}}$ 
6    $\mathcal{H} \leftarrow \mathcal{H} \cup \{H_{\theta,d}\}$ ;
7    $\mathcal{U} \leftarrow \{U \cap H_{\theta,d} \mid U \in \mathcal{U}\}$ ; // reduce the neighborhoods by the new separator  $H_{\theta,d}$ 
8 end
9 return  $\mathcal{H}$ ;
```

Algorithm 2: Binary Search

input : $a, b, \theta, \mathcal{U}$
output: a distance scalar

```

1 set a small value  $\epsilon > 0$ ;
2 if  $b - a \leq \epsilon$  then
3   return  $b$ ;
4 end
5  $c \leftarrow (a + b)/2$ ;
6 if  $\text{BPCVerify}(H_{\theta,c}, \mathcal{U})$  is true then
7   return  $\text{BinarySearch}(a, c, \theta, \mathcal{U})$ ;
8 else
9   return  $\text{BinarySearch}(c, b, \theta, \mathcal{U})$ ;
10 end
```

At each iteration, this algorithm first enumerates an angle θ . There are many angle enumeration schemes that can be used, such as an equiangular enumeration, a random enumeration, or an angle bisection. For the following demonstration, we choose the equiangular enumeration scheme described in Algorithm 1. For every enumerated angle θ , the algorithm generates a BPC separator $H_{\theta,d}$ where the distance d is returned by the binary search subroutine provided in Algorithm 2. To execute this subroutine, a BPC separator verifier is required. The implementation of this verifier also depends on the specific neighborhood representations. For instance, if each neighborhood is represented by the intersection of half-planes or a sequence of points, then verifying the turning points on the boundary is sufficient; if the neighborhoods are ellipsoids, we can even replace the binary search subroutine by finding the two tangent lines of every ellipsoid that are perpendicular to $H_{\theta,c}$. After generating a new half-plane $H_{\theta,d}$, the last step in the for-loop is to trim each neighborhood by the newly generated $H_{\theta,d}$.

In general, the binary search subroutine only returns a locally minimized distance d . However, according to the following proposition, it converges to the global minimum when all neighborhoods are convex.

Proposition 2. *Given a set of convex neighborhoods \mathcal{U} , the binary search subroutine described in Algorithm 2 converges to the minimized distance*

$$\begin{aligned} \min d \\ \text{s.t. } H_{\theta,d} \in \mathcal{H}_{\mathcal{U}}, \\ d \in [0, r], \end{aligned}$$

when $\epsilon \rightarrow 0$.

Proof. Given the initial input $(0, r, \theta, \mathcal{U})$, it suffices to show that, for any value $c \in (0, r)$, if $H_{\theta,c}$ is a BPC separator, then so is $H_{\theta,b}$ for all $b \in (c, r]$; if $H_{\theta,c}$ is not a BPC separator, neither is $H_{\theta,a}$ for all $a \in [0, c)$. To prove the former, we suppose $H_{\theta,c}$ is a BPC separator of \mathcal{U} , i.e., it is a BPC separator for every $U \in \mathcal{U}$. Take any $U \in \mathcal{U}$ and $b \in (c, r]$, the boundary points that are not contained in the half-plane $H_{\theta,b}$ is denoted by $\partial U \setminus H_{\theta,b}$. Clearly, we have the following

$$\partial U \setminus H_{\theta,b} \subseteq \partial U \setminus H_{\theta,c}.$$

Hence, for every $p \in \partial U \setminus H_{\theta,b}$, the projection $\text{proj}_{H_{\theta,c}}(p)$ is inside U , since $H_{\theta,c}$ is BPC separator of U by assumption. Moreover, because $H_{\theta,b}$ is parallel to $H_{\theta,c}$, the line segment $[p, \text{proj}_{H_{\theta,c}}(p)]$ intersects the line $\partial H_{\theta,b}$ at the projection point $\text{proj}_{H_{\theta,b}}(p)$. Thus, $\text{proj}_{H_{\theta,b}}(p) \in U$ due to the convexity of U , which shows $H_{\theta,b}$ is a BPC separator of U by definition. For the latter statement, we take any $a \in [0, c)$ and assume $H_{\theta,c}$ is not a BPC separator of \mathcal{U} . That is, for some $U \in \mathcal{U}$, $H_{\theta,c}$ is not a BPC separator of U . This means for some point $p \in \partial U \setminus H_{\theta,c}$, the projection $\text{proj}_{H_{\theta,c}}(p)$ is not in U . Same as before, the point p belongs to $\partial U \setminus H_{\theta,a}$. Towards a contradiction, suppose $H_{\theta,a}$ is a BPC separator of \mathcal{U} , then $\text{proj}_{H_{\theta,a}}(p) \in U$. Same as before, the segment $[p, \text{proj}_{H_{\theta,a}}(p)]$ intersects the line $\partial H_{\theta,c}$ at the projection point $\text{proj}_{H_{\theta,c}}(p)$. Then, the convexity of U implies $\text{proj}_{H_{\theta,c}}(p) \in U$, which contradicts the choice of p . This concludes the proof. \square

Notice that, if all neighborhoods are convex, the trimming operation in the last step of the for-loop in Algorithm 1 preserves the convexity. Thus, intuitively, when N and ϵ converge to ∞ and 0, the intersection of the BPC separators \mathcal{H} that are returned by Algorithm 1 converges to the domination hull $D_{\mathcal{U}}$.

In the next section, we will propose a mixed-integer nonlinear programming model for solving PDTSPN with convex neighborhoods. The BPC separators can be generated as a pre-processing step to cut off some portions of the neighborhoods that contain suboptimal visiting points, which may significantly reduce the overall solution space.

4 Mathematical Formulation for PDTSPN

In the previous section, we proposed two optimality conditions for TSPN that are valid for neighborhoods with general geometric properties. As discussed before, the inner problem of Formulation (1) can be strengthened by using these results. In this section, we focus on PDTSPN, and provide a mathematical formulation along with the solution approach to solve instances where the neighborhoods are convex sets.

For this formulation, we use binary variables $e = \{e_{ij} \mid i, j \in I_0 \text{ and } i \neq j\}$ to indicate whether neighborhood U_i appears immediately before neighborhood U_j in the optimal tour, and use z_{ij} to represent the distance between the two points $(x_i, y_i) \in U_i$ and $(x_j, y_j) \in U_j$, where the visits to such neighborhoods take place.

Given any visiting sequence $\sigma \in S_n$ that violates the supply and demand requirements given by constraints (1b), let $k \in I$ be the first neighborhood in σ that violates the corresponding constraint (1b) and denote $T = \{(i-1, i)\}_{i=1}^k$ the violating subpath of σ . Furthermore, let \mathcal{T} comprise the set of violating subpaths associated with all the infeasible visiting sequences in S_n . Then, Formulation (1) admits the following nonlinear program,

$$\min \sum_{i,j \in I_0, i \neq j} e_{ij} z_{ij} \tag{3a}$$

$$\text{s.t.} \quad \sum_{j \in I_0, j \neq i} e_{ij} = 1, \quad i \in I_0, \tag{3b}$$

$$\sum_{j \in I_0, j \neq i} e_{ji} = 1, \quad i \in I_0, \tag{3c}$$

$$\sum_{i,j \in I', i \neq j} e_{ij} \leq |I'| - 1, \quad I' \subsetneq I_0, |I'| \geq 2, \tag{3d}$$

$$\sum_{(i,j) \in T} e_{ij} \leq |T| - 1, \quad T \in \mathcal{T}, \tag{3e}$$

$$f_i(x_i, y_i) \leq 0, \quad i \in I, \tag{3f}$$

$$(x_i - x_j)^2 + (y_i - y_j)^2 \leq z_{ij}^2, \quad i, j \in I_0, i \neq j, \tag{3g}$$

$$x_i, y_i \text{ free}, \quad i \in I_0, \tag{3h}$$

$$z_{ij} \geq 0, \quad i, j \in I_0, i \neq j \tag{3i}$$

$$e_{ij} \in \{0, 1\}, \quad i, j \in I_0, i \neq j. \tag{3j}$$

The objective function (3a) minimizes the total traveling distance of the tour. Constraints (3b)–(3d) and (3j) are adapted from the Danzig–Fulkerson–Johnson (DFJ) formulation for TSP that ensures every feasible solution is a Hamiltonian tour over I_0 . Constraint (3e), which is equivalent to Constraint (1b), eliminates all the visiting sequences starting at the depot that violate the supply and demand requirements. Constraint set (3f) represents the neighborhoods, which are assumed to be convex hereafter. Finally, (3g)–(3i) indicate that z_{ij} is the Euclidean distance between the points (x_i, y_i) and (x_j, y_j) .

4.1 Solution Approach

Let E_{sd} and \bar{E}_{sd} be the set of solutions that satisfy (3b)–(3e) with and without constraint set (3j), respectively. That is, \bar{E}_{sd} is a polyhedron obtained by relaxing the binary constraints of E_{sd} . Then, we define $\Phi(e)$ as the following problem,

$$\Phi(e) := \min_{x,y,z} \left\{ \sum_{i,j \in I_0} e_{ij} (z_{ij} - \bar{z}_{ij}) \mid \text{Constraints (3f)–(3i)} \right\}, \tag{4}$$

where \bar{z}_{ij} is the minimum distance between neighborhoods U_i and U_j defined as

$$\bar{z}_{ij} := \min_{x_i \in U_i, x_j \in U_j} \|x_i - x_j\|_2.$$

With these notations, Formulation (3) can be equivalently written as,

$$\min \sum_{i,j \in I_0} \bar{z}_{ij} e_{ij} + \theta \tag{5a}$$

$$\text{s.t. } e \in E_{sd}, \tag{5b}$$

$$\theta \geq \Phi(e), \tag{5c}$$

$$\theta \geq 0. \tag{5d}$$

Because \bar{z}_{ij} , by definition, is a valid lower bound of z_{ij} for all $i, j \in I_0$, this reformulation allows us to initialize the objective of any feasible solution e to be at least as large as the lower bound $\sum_{i,j \in I_0} \bar{z}_{ij} e_{ij}$. Another observation is that, when the neighborhood functions f 's are convex, the subproblem $\Phi(e)$ belongs to the class of convex optimization. For instance, if the f 's are either the intersection of half-planes or ellipsoids, $\Phi(e)$ is a second-order cone program. In this case, $\Phi(e)$ admits a dual representation that satisfies strong duality and can be solved efficiently. Moreover, the resulting problem can be solved via the generalized Benders decomposition (Geoffrion, 1972).

To this end, we decompose Formulation (3) into a master problem, obtained directly from Formulation (5) by replacing (5c) with a set of *generalized Benders optimality cuts* (GBCs), which are iteratively generated by the subproblem $\Phi(e)$. The outline of the algorithm is as follows. At each iteration, we solve the master problem under the current set of GBCs to obtain a candidate visiting order $e^* \in E_{sd}$ and an estimated value of θ^* . Then, we solve subproblem $\Phi(e^*)$, parameterized with such an order to find the optimal visiting points of the corresponding tour. Then, if the optimal value of the subproblem is larger than θ^* , we generate a new GBC and add it to the current constraint pool of the master problem.

By this design, at each iteration, the value of the master problem always serves as a lower bound, and the value that results by replacing θ^* with the optimal subproblem's objective is an upper bound as it finds the optimal visiting locations of the given feasible order. This iterative procedure terminates when a desired gap between the lower and upper bounds is achieved. In the next subsection, we will discuss the implementation details of the master problem and the subproblem.

4.2 Implementation Details

As with most Benders decomposition approaches, constraint (5c) in the master problem is replaced by a set of GBCs that are produced iteratively from the subproblem. Also, constraint (5b) is composed of constraints from (3b)–(3e) and (3j). In particular, since both the subtour elimination constraint set (3d) and the supply-demand constraint set (3e) are potentially quite large in practice, even for instances of moderate sizes, we employ an iterative separation method commonly used to handle such constraints (Behdani and Smith, 2014).

At each iteration, when solving the master problem, we use the following procedure to iteratively produce a feasible visiting order that satisfies the supply and demand constraints:

1. Initialize the master problem with a subset (possibly empty) of constraint sets (3d) and (3e);
2. Solve the master problem to obtain a candidate solution e^* ;
3. Following the visiting order represented by e^* , test whether the supply-demand constraints (1b) are violated or whether e^* contains a subtour. This may result in the following three possible cases:
 - (a) at a certain neighborhood $i \in I_0$, the corresponding subpath T associated with the visiting order violates the supply-demand requirements. In that case, we add the corresponding violated constraint (3e) into the master problem and then go back to Step 2;
 - (b) the candidate solution contains a subtour, in which case we add the corresponding subtour elimination constraint (3d) into the master and then go back to Step 2;
 - (c) otherwise, the subroutine returns a feasible visiting order.

For subproblem $\Phi(e^*)$, as long as functions f 's are convex, $\Phi(e^*)$ is a convex optimization problem. Therefore, any off-the-shelf solver able to handle such type of problems can be used to obtain an optimal solution (x^*, y^*, z^*) .

Let π^* and μ^* be the optimal value of the dual variables associated with constraints (3f) and (3g). Then, the GBCs, as given by Geoffrion (1972), are as follows:

$$\theta \geq \sum_{i,j \in I_0, i \neq j} e_{ij}(z_{ij}^* - \bar{z}_{ij}) - \sum_{i \in I_0} \pi_i^* f_i(x_i^*, y_i^*) - \sum_{i,j \in I_0, i \neq j} \mu_{ij}^* ((x_i^* - x_j^*)^2 + (y_i^* - y_j^*)^2 - z_{ij}^*). \quad (6)$$

Furthermore, notice that since e only appears in the subproblem's objective and not in constraints (3f) and (3g), the last two terms in (6) can be omitted, as they are always equal to zero by complementary slackness. The GBCs can then be expressed as

$$\theta \geq \sum_{i,j \in I_0, i \neq j} e_{ij}(z_{ij}^* - \bar{z}_{ij}). \quad (7)$$

This turns out to be quite convenient in practice, as constraints (7) can be generated directly by solving the (primal) subproblem $\Phi(e^*)$ without the need of obtaining the optimal dual solution (π^*, μ^*) .

Remark 1. Subproblem $\Phi(e^*)$ is feasible for any given visiting sequence produced by the master problem (3). Therefore, our implementation does not require any Benders feasibility cuts.

4.3 Solution Approaches for Modeling Other Supply-Demand Precedences

Up to this point, our discussion has been focused on the M-M version of the problem. Therefore, to solve any of the other variants described in Section 1, Formulation (3) must be adapted to represent the corresponding demand-supply precedence requirements. In general, under some minor considerations, this can often be done by simply identifying the corresponding set \mathcal{T} of violating subpaths in constraints (3e). In this subsection, we illustrate how to do this for the 1-M-1 and the 1-1 versions.

4.3.1 1-M-1 PDTSPN.

For this case, given any visiting sequence $\sigma \in S_n$, one can check whether a vehicle following σ has sufficient capacity when visiting each supply neighborhood to satisfy the given pick up requirements. That is, for

every node $i \in I_s$, one can test whether

$$s_0 + \sum_{j \leq i: \sigma(j) \in I_s} s_j - \sum_{j \leq i: \sigma(j) \in I_d} d_j \leq c.$$

If such a condition is violated by σ at neighborhood k , then, the violating subpath $T = \{(i-1, i)\}_{i=1}^k$ exists in \mathcal{T} . As before, the resulting constraint set (3e) can also be populated iteratively via branch and cut.

4.3.2 1-1 PDTSPN.

For this case, it is possible to adapt the definition of set \mathcal{T} . Specifically, for the supply-demand neighborhood pair, denoted by (u_q, v_q) , associated with some commodity q , one simply needs to add to \mathcal{T} all the subpaths in which v_q appears before u_q . Alternatively, we now show that it is possible to impose such precedence requirements by adding instead a set of constraints of size polynomial, which has the benefit of not requiring the iterative separation process of constraints (3e). We now provide two possible ways of modeling these requirements.

In the first method, we add artificial binary variables $g = \{g_{ij} \mid i, j \in I_0 \text{ and } i \neq j\}$ into the formulation to explicitly indicate whether neighborhood i appears before j in the visiting sequence. Then, the following constraints along with (5b) enforce the desired precedence requirement.

$$e_{ij} \leq g_{ij}, \tag{8a}$$

$$e_{ik} + e_{kj} - 1 \leq g_{ij}, \quad k \in I \setminus \{i, j\} \tag{8b}$$

$$g_{ji} = 0, \quad \text{for all supply-demand pairs } (i, j). \tag{8c}$$

Another possible way to model the precedence requirement is using the flow constraints introduced in Hernández-Pérez and Salazar-González (2009). The basic idea is that under (5b), a positive flow from i to j ensures that $e_{ij} = 1$ and $e_{ji} = 0$. Specifically, for each supply-demand pair, denoted by (u_q, v_q) , we need to transfer a unit of commodity q from u_q to v_q . Let $r_{ij}^q \in [0, 1]$ be the flow of commodity q between consecutive neighborhoods i and j , then we have the following set of constraints to impose the required precedence relation. Note that, because of Assumption 1(ii), the commodity flow between each supply-demand neighborhood pair can be capped at one unit without loss of generality.

$$r_{ij}^q \leq e_{ij}, \quad i, j \in I_0, k \in K, \tag{9a}$$

$$\sum_{j \in I_0} r_{ji}^q - \sum_{j \in I_0} r_{ij}^q = \begin{cases} 1, & \text{if } i = u_q, \\ -1, & \text{if } i = v_q, \\ 0, & \text{otherwise,} \end{cases} \quad \forall i \in I, q \in Q. \tag{9b}$$

For many other PDTSPN variants, similar strategies can be adopted to modify Formulation (3) and its implementation.

5 Numerical Experiments

In this section, we conduct computational experiments to test the different developments presented in this work. In particular, we analyze the effect that the proposed domination hull algorithm (Algorithm 1) has on reducing the size of the neighborhoods. Additionally, we test the performance of our generalized Benders decomposition approach in terms of the solution times, optimality gap, and cut separation times.

We test our algorithms by solving PDTSPN instances with two supply-demand precedence types, namely 1-1 and M-M. We conduct our experiments on a computer equipped with an Intel(R) Core(TM) i7-11700F 2.50GHz processor and 16GB of RAM and running under Windows 10. All the algorithms were implemented in Python 3.6.9 and solved by the commercial optimizer Gurobi 9.1.2 with a one-hour time limit.

5.1 Test Instances

We generate 150 instances with different numbers and sizes of neighborhoods, where n , the number of neighborhoods, ranges from $\{6, 8, \dots, 30\}$. All neighborhoods of each instance are assumed to be convex polygons generated from 10 randomly chosen points within a l by l square, where the side length $l \in \{0.75, 1, 1.25\}$ controls the upper limit of the neighborhood size. We call this length l the size controller hereafter. To locate these neighborhoods, the center (x, y) of every square (as well as the depot) are randomly picked from intervals $[0, 16]$ and $[0, 10]$.

For each 1-1 PDTSPN instance, we generated the supply-demand pairs by randomly pairing two numbers from the index set $I = \{1, 2, \dots, n\}$. For each M-M PDTSPN instance, the supply or demand quantity of each neighborhood is a positive integer randomly generated from $\{1, 2, \dots, 10\}$. We also ensure that the sums of supplies and demands are equal.

Under this settings, it should be clear that the complexity of an instance mainly depends on the number of neighborhoods n and the size controller l , since all the other parameters are either identical or random. Therefore, we call every pair (n, l) an instance configuration. For both the 1-1 and M-M versions, we generate five instances for each configuration. In the following tables, we aggregate the computational results for each instance configuration and present the averages, as instances with the same configuration are expected to yield similar results.

5.2 Domination Hull Approximation

We begin our discussion by analyzing the performance of the domination hull approximation algorithm (Algorithm 1). For each configuration, we generate 10, 16, 22, and 28 separators and report the corresponding neighborhood size, the size reduction in percentage, and the runtime in seconds in Table 1.

In general, Algorithm 1 seems to be quite efficient at identifying domination hull approximations. It halts within a second on all the test instances thanks to the binary search implementation. As expected, the running times increase as the number of separators and neighborhoods increase.

The results in Table 1, show the average percentage of the area that is cut off by the BPC separators. We observe that a sizable portion of the neighborhood area, ranging from 9% to 47%, is trimmed by the separators in each instance. This phenomenon is more significant for instances with a smaller n . For instance, when $n = 6$, an average of 31.66% of the neighborhood area is removed by merely 10 separators, and this number increases to 45.96% with 28 generated separators. They are significantly larger than the

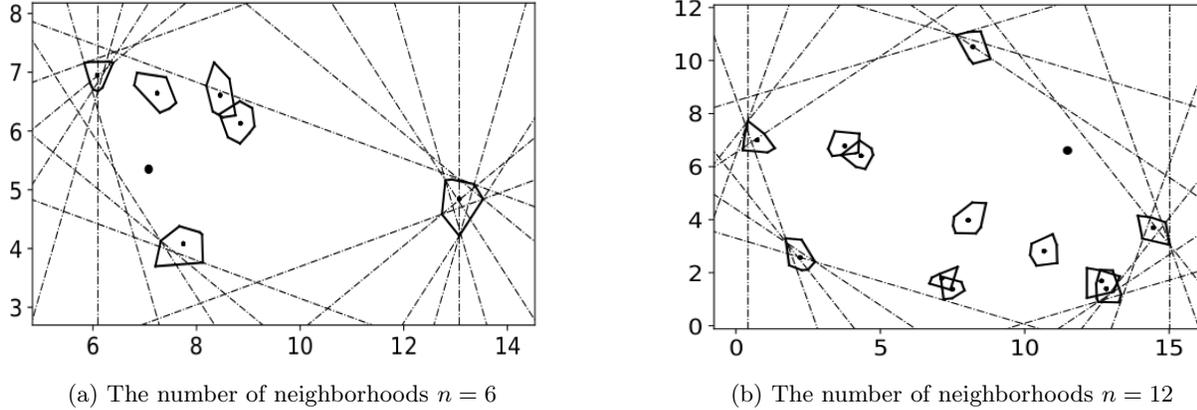


Figure 8: Illustration of separators (dashed lines) generated by Algorithm 1

corresponding numbers 8.47% and 12.74% in the case $n = 30$. The reason is that the domination hull is essentially determined by the peripheral neighborhoods (see Figure 8). When more neighborhoods are present, more of them are likely to lie in the interior of the domination hull. For instance, in Figure 8(a), only two out of six neighborhoods are not affected by the separators, while five out of twelve neighborhoods are not affected in Figure 8(b).

Another interesting observation is that the law of diminishing marginal utility seems to apply on the trimming effects of the separators. In Table 1 can be observed that, for every configuration, the compound area reduction diminishes as the number of separators increases. This phenomenon suggests that one can set the implementation to stop after a small set of separators are added to save computational resources without compromising much on the quality of the separators.

5.3 Generalized Benders Decomposition Approach

In this subsection, we analyze the computational results of the generalized Benders decomposition approach introduced in Section 4.

For each configuration, we aggregate and report the numbers of subtour elimination cuts, supply-demand cuts, and generalized Benders optimality cuts, as well as the associated separation times. We also collect the corresponding optimality gaps and the number of solved instances in percentage. All results are presented in Tables 2 and 3 for the 1-1 and M-M versions of PDTSPN, respectively.

The results on both types of instances share several similarities. First, the proposed approach can efficiently solve configurations with $n \leq 18$ to optimality, except for the case where $n = 18$ and $l = 1$. All the other instances under that size were solved within a minute on average. When the size becomes larger, some instances reach the time limit before closing the optimality gap. However, even for large instances such as $n = 20$ and 22 , many were still solved to optimality as evidenced by the ‘‘Opt. Pct’’ column in both Tables 2 and 3. Second, we observe that the separation time spent on the general Benders cuts is significantly longer than the ones for supply-demand and subtour elimination cuts, which is expected as the former requires solving subproblem (4). Third, we notice that size of the instances grow, most of the algorithm running time is spent on solving the master problem (5).

In our implementation, the 1-1 PDTSPN instances are solved with the precedence constraints (9) added

Table 1: Results for the domination hull approximation algorithm (Algorithm 1). For each configuration, we generate 10, 16, 22, and 28 separators and report the corresponding neighborhood size (Area Size), the size reduction in percentage (Cutoff Pct (%)), and the runtime in seconds (Time)

n	l	10 Separators		16 Separators		22 Separators		28 Separators	
		Cutoff Pct (%)	Time						
6	0.75	36.30	0.08	39.63	0.11	42.15	0.16	45.20	0.20
	1	30.40	0.08	36.01	0.14	41.96	0.18	46.58	0.23
	1.25	28.29	0.08	34.97	0.12	40.92	0.17	46.09	0.22
	Avg	31.66	0.08	36.87	0.13	41.68	0.17	45.96	0.22
8	0.75	20.97	0.10	29.56	0.16	32.48	0.22	35.50	0.28
	1	23.20	0.11	27.39	0.17	29.54	0.24	32.74	0.30
	1.25	26.89	0.11	28.91	0.17	33.43	0.24	34.96	0.30
	Avg	23.69	0.11	28.62	0.17	31.82	0.23	34.40	0.29
10	0.75	19.44	0.12	21.38	0.19	24.81	0.26	25.05	0.34
	1	18.84	0.12	19.36	0.19	22.66	0.26	23.94	0.33
	1.25	24.08	0.14	28.03	0.21	31.69	0.30	33.38	0.37
	Avg	20.79	0.13	22.92	0.20	26.39	0.27	27.46	0.35
12	0.75	17.75	0.16	20.01	0.24	24.16	0.33	26.50	0.42
	1	18.51	0.16	20.06	0.26	22.93	0.35	24.41	0.44
	1.25	18.55	0.13	21.13	0.21	23.36	0.29	26.29	0.37
	Avg	18.27	0.15	20.40	0.24	23.48	0.32	25.73	0.41
14	0.75	14.40	0.18	17.58	0.28	19.15	0.38	20.40	0.49
	1	17.06	0.19	20.43	0.30	24.10	0.41	24.35	0.52
	1.25	17.72	0.19	19.71	0.30	24.48	0.41	25.31	0.53
	Avg	16.40	0.19	19.24	0.29	22.58	0.40	23.35	0.51
16	0.75	13.09	0.19	17.20	0.31	18.45	0.43	19.31	0.55
	1	12.54	0.19	15.00	0.31	17.23	0.42	18.37	0.54
	1.25	12.08	0.19	14.28	0.31	17.12	0.42	18.05	0.54
	Avg	12.57	0.19	15.49	0.31	17.60	0.43	18.58	0.54
18	0.75	10.17	0.21	16.55	0.34	16.75	0.46	18.52	0.58
	1	11.37	0.21	13.50	0.35	16.30	0.47	16.13	0.60
	1.25	11.78	0.22	15.69	0.35	15.37	0.48	17.73	0.61
	Avg	11.11	0.21	15.25	0.34	16.14	0.47	17.46	0.60
20	0.75	10.58	0.23	11.82	0.35	14.11	0.49	14.33	0.62
	1	10.60	0.23	13.46	0.37	14.98	0.50	16.85	0.65
	1.25	11.09	0.22	13.72	0.36	15.77	0.50	16.56	0.65
	Avg	10.76	0.23	13.00	0.36	14.95	0.50	15.92	0.64
22	0.75	12.27	0.26	13.27	0.42	15.72	0.59	15.78	0.74
	1	10.70	0.28	13.05	0.44	14.71	0.60	15.98	0.77
	1.25	11.84	0.27	14.83	0.44	16.95	0.61	19.09	0.77
	Avg	11.60	0.27	13.72	0.44	15.79	0.60	16.95	0.76
24	0.75	9.58	0.30	11.49	0.47	12.52	0.65	12.85	0.83
	1	11.09	0.31	12.84	0.50	14.96	0.69	15.82	0.86
	1.25	9.12	0.27	12.17	0.44	13.90	0.62	14.21	0.77
	Avg	9.93	0.29	12.17	0.47	13.79	0.65	14.29	0.82
26	0.75	9.15	0.33	10.50	0.53	11.97	0.71	12.08	0.89
	1	8.09	0.28	11.43	0.45	12.30	0.64	13.44	0.80
	1.25	7.25	0.30	11.05	0.49	11.09	0.67	11.45	0.83
	Avg	8.17	0.30	10.99	0.49	11.79	0.67	12.32	0.84
28	0.75	8.28	0.32	10.97	0.52	12.24	0.71	13.60	0.90
	1	8.75	0.32	9.62	0.52	11.15	0.70	12.28	0.90
	1.25	8.65	0.31	10.86	0.50	11.63	0.67	12.26	0.86
	Avg	8.56	0.32	10.48	0.51	11.67	0.70	12.71	0.89
30	0.75	7.76	0.36	9.06	0.59	11.83	0.80	11.97	1.03
	1	8.84	0.38	10.63	0.60	11.60	0.83	12.67	1.04
	1.25	8.82	0.37	10.97	0.57	12.22	0.81	13.58	1.02
	Avg	8.47	0.37	10.22	0.59	11.88	0.81	12.74	1.03

Table 2: Results for the generalized Benders decomposition approach on 1-1 PDTSPN instances. For each configuration, we report the average number and separation time of subtour elimination cuts (Num. SECs and SEC Time), the average number and separation time of generalized Benders optimality cuts (Num. GBCs and GBC Time), the average total runtime for solving the instance (Model Time), the average optimality gap (MIP Gap), and the average percentage of instances solved to optimality (Opt. Pct).

n	l	Num. SECs	Num. GBCs	SEC Time	GBC Time	Model Time	MIP Gap (%)	Opt. Pct (%)
6	0.75	4.8	5.8	<0.001	0.058	0.078	0.00	100
	1	6.4	12.0	<0.001	0.116	0.147	0.00	100
	1.25	6.4	9.0	<0.001	0.085	0.102	0.00	100
	Avg	5.9	8.9	<0.001	0.086	0.109	0.00	100
8	0.75	15.2	14.8	<0.001	0.233	0.301	0.00	100
	1	8.0	13.6	<0.001	0.213	0.254	0.00	100
	1.25	10.8	11.0	<0.001	0.162	0.215	0.00	100
	Avg	11.3	13.1	<0.001	0.203	0.257	0.00	100
10	0.75	22.4	27.2	<0.001	0.622	0.737	0.00	100
	1	18.4	20.6	<0.001	0.457	0.559	0.00	100
	1.25	16.4	12.6	<0.001	0.281	0.404	0.00	100
	Avg	19.1	20.1	<0.001	0.454	0.567	0.00	100
12	0.75	34.0	27.8	<0.001	0.948	1.367	0.00	100
	1	13.2	19.6	<0.001	0.572	0.763	0.00	100
	1.25	52.8	45.8	0.001	1.384	1.716	0.00	100
	Avg	33.3	31.1	0.000	0.968	1.282	0.00	100
14	0.75	85.6	38.2	0.001	1.778	2.581	0.00	100
	1	90.0	41.0	0.001	1.788	2.639	0.00	100
	1.25	54.0	91.4	0.001	4.274	5.910	0.00	100
	Avg	76.5	56.9	0.001	2.613	3.710	0.00	100
16	0.75	72.4	66.6	0.001	3.457	5.969	0.00	100
	1	116.8	113.4	0.002	6.452	12.932	0.00	100
	1.25	95.6	363.8	0.002	20.200	27.446	0.00	100
	Avg	94.9	181.3	0.002	10.036	15.449	0.00	100
18	0.75	152.4	131.2	0.004	11.300	33.563	0.00	100
	1	423.6	124.4	0.017	10.698	390.422	0.00	100
	1.25	382.0	247.0	0.009	20.361	44.748	0.00	100
	Avg	319.3	167.5	0.010	14.119	156.245	0.00	100
20	0.75	190.4	164.6	0.005	18.292	116.984	0.00	100
	1	401.6	508.6	0.013	57.358	381.838	0.00	100
	1.25	402.4	542.6	0.014	61.908	637.254	0.00	100
	Avg	331.5	405.3	0.011	45.853	378.692	0.00	100
22	0.75	278.8	230.2	0.010	28.158	830.674	0.57	80
	1	830.0	471.2	0.044	58.693	2577.876	4.62	40
	1.25	592.0	492.0	0.030	62.129	1531.215	1.93	80
	Avg	566.9	397.8	0.028	49.660	1646.588	2.37	67
24	0.75	567.6	349.0	0.061	53.804	2305.244	2.96	60
	1	683.6	386.8	0.062	60.529	3118.071	10.16	20
	1.25	1096.4	880.2	0.137	138.192	3600.000	14.83	0
	Avg	782.5	538.7	0.087	84.175	3007.835	9.31	27
26	0.75	243.6	273.8	0.029	50.868	1576.985	1.87	80
	1	443.6	943.0	0.051	166.494	2853.801	1.21	80
	1.25	654.0	584.2	0.074	109.588	3600.000	14.47	0
	Avg	447.1	600.3	0.051	108.984	2676.984	5.85	53
28	0.75	322.4	358.2	0.023	73.885	3600.000	9.36	0
	1	588.4	377.2	0.072	80.385	3600.000	15.33	0
	1.25	1091.2	468.0	0.164	102.690	3600.000	22.59	0
	Avg	667.3	401.1	0.086	85.654	3600.000	15.76	0
30	0.75	790.8	334.4	0.130	85.099	3600.000	23.92	0
	1	474.4	513.0	0.041	129.715	3600.000	19.75	0
	1.25	430.8	521.0	0.058	155.943	3600.000	27.81	0
	Avg	565.3	456.1	0.077	123.586	3600.000	23.83	0

Table 3: Results for the generalized Benders decomposition approach on M-M PDTSPN instances. For each configuration, we report the average number and separation time of subtour elimination cuts (Num. SECs), supply-demand cuts (Num. SDCs), and generalized Benders optimality cuts (Num. GBCs). We also report the average model runtime (Model Time), the average optimality gap (MIP Gap), and the average percentage of instances solved to optimality (Opt. Pct).

n	l	Num. SECs	Num. SDCs	Num. GBCs	SEC Time	SDC Time	GBC Time	Model Time	MIP Gap (%)	Opt Pct (%)
6	0.75	10.4	4.0	3.6	0.000	0.000	0.038	0.054	0.00	100
	1	23.6	8.8	8.8	0.000	0.000	0.098	0.127	0.00	100
	1.25	27.6	16.2	13.4	0.000	0.000	0.146	0.187	0.00	100
	Avg.	20.5	9.7	8.6	0.000	0.000	0.094	0.123	0.00	100
8	0.75	31.6	16.2	9.8	0.000	0.000	0.161	0.223	0.00	100
	1	41.6	22.4	14.2	0.000	0.001	0.249	0.311	0.00	100
	1.25	34.8	11.6	12.2	0.000	0.000	0.206	0.259	0.00	100
	Avg	36.0	16.7	12.1	0.000	0.000	0.205	0.264	0.00	100
10	0.75	127.6	89.0	32.6	0.001	0.002	0.777	0.878	0.00	100
	1	87.6	39.6	26.6	0.001	0.001	0.714	0.787	0.00	100
	1.25	79.2	30.8	22.4	0.001	0.001	0.565	0.615	0.00	100
	Avg	98.1	53.1	27.2	0.001	0.001	0.685	0.760	0.00	100
12	0.75	133.2	44.6	23.8	0.001	0.001	0.829	0.894	0.00	100
	1	151.6	100.4	28.0	0.001	0.003	0.900	1.006	0.00	100
	1.25	147.6	62.6	41.4	0.001	0.002	1.483	1.597	0.00	100
	Avg	144.1	69.2	31.1	0.001	0.002	1.071	1.166	0.00	100
14	0.75	262.0	84.2	38.6	0.002	0.003	1.818	1.993	0.00	100
	1	187.2	70.8	22.2	0.002	0.002	1.075	1.344	0.00	100
	1.25	564.8	768.8	77.2	0.005	0.023	3.757	5.198	0.00	100
	Avg	338.0	307.9	46.0	0.003	0.010	2.217	2.845	0.00	100
16	0.75	371.6	185.6	38.4	0.004	0.006	2.042	2.535	0.00	100
	1	667.6	247.2	124.0	0.007	0.010	7.221	8.228	0.00	100
	1.25	1172.8	792.6	352.0	0.013	0.032	21.702	28.442	0.00	100
	Avg	737.3	408.5	171.5	0.008	0.016	10.321	13.068	0.00	100
18	0.75	1017.2	1149.0	124.4	0.012	0.044	10.929	16.659	0.00	100
	1	597.2	294.8	133.6	0.006	0.012	11.732	13.452	0.00	100
	1.25	1838.8	415.6	668.8	0.025	0.026	59.078	141.670	0.00	100
	Avg	1151.1	619.8	308.9	0.014	0.027	27.247	57.261	0.00	100
20	0.75	2151.6	3113.4	123.4	0.031	0.148	13.808	127.979	0.00	100
	1	1140.0	628.4	318.6	0.015	0.031	35.045	54.026	0.00	100
	1.25	9496.8	3599.8	429.8	0.184	0.187	49.115	804.967	2.49	80
	Avg	4262.8	2447.2	290.6	0.076	0.122	32.656	328.991	0.83	93
22	0.75	1194.0	565.0	168.6	0.016	0.026	23.523	35.449	0.00	100
	1	4888.4	1426.2	573.0	0.096	0.089	83.438	554.685	0.00	100
	1.25	6489.6	8880.4	1129.8	0.112	0.529	161.348	1268.469	1.97	80
	Avg	4190.7	3623.9	623.8	0.075	0.215	89.436	619.534	0.66	93
24	0.75	692.0	492.0	141.8	0.008	0.022	23.701	27.609	0.00	100
	1	6674.4	4693.6	787.2	0.130	0.254	140.985	1622.960	2.81	60
	1.25	17027.2	13257.2	945.2	0.333	0.726	171.623	2900.182	8.95	20
	Avg	8131.2	6147.6	624.7	0.157	0.334	112.103	1516.917	3.92	60
26	0.75	19348.8	7383.2	408.4	0.510	0.546	83.768	1570.052	1.69	60
	1	12798.8	11515.2	632.8	0.281	0.686	134.660	2206.446	5.31	40
	1.25	12745.6	11138.0	1064.8	0.236	0.675	227.720	3311.867	6.29	40
	Avg	14964.4	10012.1	702.0	0.342	0.636	148.710	2362.789	4.43	47
28	0.75	8915.6	9400.6	356.4	0.174	0.574	83.147	1960.745	1.65	80
	1	5844.4	1775.6	837.4	0.135	0.153	199.365	1400.179	1.90	80
	1.25	18527.6	10192.2	1474.0	0.401	0.737	375.003	3470.617	7.16	20
	Avg	11095.9	7122.8	889.3	0.237	0.488	219.172	2277.180	3.57	60
30	0.75	8775.6	9480.0	513.2	0.162	0.718	150.546	1626.341	2.84	60
	1	12975.6	13859.2	724.2	0.281	1.116	219.909	3246.540	5.22	20
	1.25	23480.4	12804.0	1072	0.489	0.883	336.829	3600.000	13.32	0
	Avg	15077.2	12047.7	769.8	0.311	0.906	235.761	2824.564	7.13	27

to the master problem (5). This leads to several performance differences when compared to the M-M PDTSPN case. First, in general, as shown in Tables 3 and 2 the M-M instances are solved faster than the 1-1 counterpart. Similarly, this trend is also apparent when comparing the MIP gaps and number of instances solved of the large instances. Second, in Table 3, there is a clear pattern showing that the instances with larger l are more difficult to solve. Except for small instances where $n \leq 10$, the instances with $l = 1.25$ take much longer to solve than the others. This is mainly because larger areas potentially lead to more overlaps between the neighborhoods, which may produce many near-optimal solutions with different visiting sequences. However, this phenomenon is less apparent in Table 2. One explanation is that the one-to-one precedence constraints imposed in the master problem vastly reduce the number of feasible visiting sequences, which undermines the effects caused by the overlapped areas.

6 Concluding Remarks

In this paper we studied a variant of the traveling salesman problem called the pickup-and-delivery traveling salesman problem with neighborhoods, which combines the main characteristics of the pickup and delivery TSP and the TSP with neighborhoods.

First, we derived two optimality conditions for PDTSPN with compact neighborhoods that can have arbitrary shapes. The first condition characterizes whether a given tour is locally optimal at the visiting point of each neighborhood. The second condition states that every boundary-projection-closed separator contains all the non-redundant optimal tours. Hence, the intersection of these separators, defined as the domination hull, can be used to cut off sub-optimal regions of the neighborhoods. We also developed a binary-search-based algorithm to approximate the domination hull efficiently.

Next, we proposed a mixed-integer nonlinear programming formulation to solve PDTSPN on instances with convex neighborhoods. A cut-generation solution approach was developed based on the generalized Benders decomposition. This approach iterates between two models: a master problem that identifies a feasible visiting sequence and a subproblem that searches for an optimal visiting location under the given sequence. We further adapted this method to tackle instances of PDTSPN with different supply-demand precedence relationships, including the 1-1, 1-M-1, and M-M versions.

Finally, we conducted an extensive set of computational experiments to test the proposed solution approach on a wide variety of instances. Our experiments showed that a small number of separators is sufficient to trim a large portion of the neighborhoods. Also the experiments evidenced that our solution approach works efficiently on most of the tested instances.

Due to the generality of the proposed optimality conditions and solution approaches, we can extend them to other TSPN-related problems that may even involve non-convex neighborhoods.

References

- S. Anily and G. Mosheiov. The traveling salesman problem with delivery and backhauls. *Operations Research Letters*, 16(1):11–18, 1994.
- E. M. Arkin and R. Hassin. Approximation algorithms for the geometric covering salesman problem. *Discrete Applied Mathematics*, 55(3):197–218, 1994.
- M. Battarra, J.-F. Cordeau, and M. Iori. Chapter 6: pickup-and-delivery problems for goods transportation. In *Vehicle Routing: Problems, Methods, and Applications, Second Edition*, pages 161–191. Society for Industrial and Applied Mathematics, 2014.
- B. Behdani and J. C. Smith. An integer-programming-based approach to the close-enough traveling salesman problem. *INFORMS Journal on Computing*, 26(3):415–432, 2014.
- G. Berbeglia, J.-F. Cordeau, I. Gribkovskaia, and G. Laporte. Static pickup and delivery problems: a classification scheme and survey. *Top*, 15(1):1–31, 2007.
- G. Berbeglia, J.-F. Cordeau, and G. Laporte. Dynamic pickup and delivery problems. *European Journal of Operational Research*, 202(1):8–15, 2010.
- F. Carrabs, C. Cerrone, R. Cerulli, and M. Gaudioso. A novel discretization scheme for the close enough traveling salesman problem. *Computers & Operations Research*, 78:163–171, 2017.
- N. Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group, 1976.
- D. Ciullo, G. D. Celik, and E. Modiano. Minimizing transmission energy in sensor networks via trajectory control. In *8th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*, pages 132–141. IEEE, 2010.
- J.-F. Cordeau, M. Dell’Amico, and M. Iori. Branch-and-cut for the pickup and delivery traveling salesman problem with FIFO loading. *Computers & Operations Research*, 37(5):970–980, 2010a.
- J.-F. Cordeau, M. Iori, G. Laporte, and J. J. Salazar González. A branch-and-cut algorithm for the pickup and delivery traveling salesman problem with LIFO loading. *Networks*, 55(1):46–59, 2010b.
- W. P. Coutinho, R. Q. d. Nascimento, A. A. Pessoa, and A. Subramanian. A branch-and-bound algorithm for the close-enough traveling salesman problem. *INFORMS Journal on Computing*, 28(4):752–765, 2016.
- A. Dumitrescu and J. S. Mitchell. Approximation algorithms for tsp with neighborhoods in the plane. *Journal of Algorithms*, 48(1):135–159, 2003.
- I. Dumitrescu, S. Ropke, J.-F. Cordeau, and G. Laporte. The traveling salesman problem with pickup and delivery: polyhedral results and a branch-and-cut algorithm. *Mathematical Programming*, 121(2):269–305, 2010.
- M. Gendreau, G. Laporte, and D. Vigo. Heuristics for the traveling salesman problem with pickup and delivery. *Computers & Operations Research*, 26(7):699–714, 1999.

- A. M. Geoffrion. Generalized benders decomposition. *Journal of Optimization Theory and Applications*, 10(4):237–260, 1972.
- I. Gribkovskaia, Ø. Halskau sr, G. Laporte, and M. Vlček. General solutions to the single vehicle routing problem with pickups and deliveries. *European Journal of Operational Research*, 180(2):568–584, 2007.
- J. Gudmundsson and C. Levcopoulos. A fast approximation algorithm for tsp with neighborhoods. *Nordic Journal of Computing*, 6(4):469–488, 1999.
- D. J. Gulczynski, J. W. Heath, and C. C. Price. The close enough traveling salesman problem: A discussion of several heuristics. In *Perspectives in Operations Research*, pages 271–283. Springer, 2006.
- H. Hernández-Pérez and J.-J. Salazar-González. A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery. *Discrete Applied Mathematics*, 145(1):126–139, 2004a.
- H. Hernández-Pérez and J.-J. Salazar-González. Heuristics for the one-commodity pickup-and-delivery traveling salesman problem. *Transportation Science*, 38(2):245–255, 2004b.
- H. Hernández-Pérez, I. Rodríguez-Martín, and J. J. Salazar-González. A hybrid grasp/vnd heuristic for the one-commodity pickup-and-delivery traveling salesman problem. *Computers & Operations Research*, 36(5):1639–1645, 2009.
- H. Hernández-Pérez and J.-J. Salazar-González. The multi-commodity one-to-one pickup-and-delivery traveling salesman problem. *European Journal of Operational Research*, 196(3):987–995, 2009. ISSN 0377-2217.
- A. M. Johnson, C. J. Cunningham, E. Arnold, W. D. Rosamond, and J. K. Zègre-Hemsey. Impact of using drones in emergency medicine: What does the future hold? *Open Access Emergency Medicine: OAEM*, 13:487, 2021.
- B. Kalantari, A. V. Hill, and S. R. Arora. An algorithm for the traveling salesman problem with pickup and delivery customers. *European Journal of Operational Research*, 22(3):377–386, 1985.
- C. S. Mata and J. S. Mitchell. Approximation algorithms for geometric tour and network design problems. In *Proceedings of the Eleventh Annual Symposium on Computational Geometry*, pages 360–369, 1995.
- W. K. Mennell. *Heuristics for solving three routing problems: Close-enough traveling salesman problem, close-enough vehicle routing problem, and sequence-dependent team orienteering problem*. University of Maryland, College Park, 2009.
- N. Mladenović, D. Urošević, A. Ilić, et al. A general variable neighborhood search for the one-commodity pickup-and-delivery travelling salesman problem. *European Journal of Operational Research*, 220(1):270–285, 2012.
- G. Mosheiov. The travelling salesman problem with pick-up and delivery. *European Journal of Operational Research*, 79(2):299–310, 1994.
- R. J. O’Neil and K. Hoffman. Exact methods for solving traveling salesman problems with pickup and delivery in real time. *Technical Report. George Mason University. 4400 University Dr., Fairfax, VA, 9*, 2017.

- S. N. Parragh, K. F. Doerner, and R. F. Hartl. A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, 58(2):81–117, 2008.
- H. N. Psaraftis. Analysis of an $o(n^2)$ heuristic for the single vehicle many-to-many euclidean dial-a-ride problem. *Transportation Research Part B: Methodological*, 17(2):133–145, 1983.
- Y. Qu and J. F. Bard. A branch-and-price-and-cut algorithm for heterogeneous pickup and delivery problems with configurable vehicle capacity. *Transportation Science*, 49(2):254–270, 2015.
- J. Renaud, F. F. Boctor, and J. Ouenniche. A heuristic for the pickup and delivery traveling salesman problem. *Computers & Operations Research*, 27(9):905–916, 2000.
- S. Ropke and D. Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation science*, 40(4):455–472, 2006.
- W. D. Rosamond, A. M. Johnson, B. M. Bogle, E. Arnold, C. J. Cunningham, M. Picinich, B. M. Williams, and J. K. Zègre-Hemsey. Drone delivery of an automated external defibrillator. *New England Journal of Medicine*, 383(12):1186–1188, 2020.
- M. W. Savelsbergh and M. Sol. The general pickup and delivery problem. *Transportation science*, 29(1):17–29, 1995.
- P. Toth and D. Vigo. *The vehicle routing problem*. Society for Industrial and Applied Mathematics, 2002.
- M. Veenstra, K. J. Roodbergen, I. F. Vis, and L. C. Coelho. The pickup and delivery traveling salesman problem with handling costs. *European Journal of Operational Research*, 257(1):118–132, 2017.
- X. Wang, B. Golden, and E. Wasil. A steiner zone variable neighborhood search heuristic for the close-enough traveling salesman problem. *Computers & Operations Research*, 101:200–219, 2019.
- N. Wei, J. L. Walteros, M. R. Worden, and H. J. Ortiz-Peña. A resiliency analysis of information distribution policies over mobile ad hoc networks. *Optimization Letters*, 15(4):1081–1103, 2021.
- H. Xu, Z.-L. Chen, S. Rajagopal, and S. Arunapuram. Solving a practical pickup and delivery problem. *Transportation science*, 37(3):347–364, 2003.
- F. Yakushiji, K. Yakushiji, M. Murata, N. Hiroi, K. Takeda, and H. Fujita. The quality of blood is not affected by drone transport: An evidential study of the unmanned aerial vehicle conveyance of transfusion material in japan. *Drones*, 4(1), 2020.
- B. Yuan, M. Orłowska, and S. Sadiq. On the optimal robot routing problem in wireless sensor networks. *IEEE Transactions on Knowledge and Data Engineering*, 19(9):1252–1261, 2007.
- F. Zhao, S. Li, J. Sun, and D. Mei. Genetic algorithm for the one-commodity pickup-and-delivery traveling salesman problem. *Computers & Industrial Engineering*, 56(4):1642–1648, 2009.