CTU
CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering
Department of Computer Science

Master's thesis

Generalized Routing Problems with Continuous Neighborhoods

Bc. Jindřiška Deckerová

May 2021
Supervisor: prof. Ing. Jan Faigl, Ph.D.

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Deckerová Jindřiška**	Personal ID number:	**420796**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Computer Science**		
Study program:	**Open Informatics**		
Specialisation:	**Artificial Intelligence**		

II. Master’s thesis details

Master’s thesis title in English:

Generalized Routing Problems with Continuous Neighborhoods

Master’s thesis title in Czech:

Zobecněné směrovací problémy se spojitým okolím

Guidelines:

1. Familiarize yourself with routing problems continuous neighborhoods such as the Generalized Traveling Salesman Problem with Neighborhoods (GTSPN) [1] and computation of lower bound estimates [2],[3],[4],[5],[6].
2. Implement and evaluate a branch-and-bound algorithm for the Close Enough TSP (CETSP) [7].
3. Propose generalization of the existing solvers to the high-dimensional instances of the GTSPN, such as unsupervised learning based heuristic [8] and branch-and-bound algorithm [7].
4. Evaluate performance of the developed solvers in regular and high-dimensional instances.

Bibliography / sources:

[1] K. Vicencio, B. Davis and I. Gentilini, "Multi-goal path planning based on the generalized Traveling Salesman Problem with neighborhoods," 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, 2014, pp. 2985-2990, doi: https://doi.org/10.1109/IROS.2014.6942974.
[2] F. Carrabs, C. Cerrone, R. Cerulli, and C. D'Ambrosio, ”Improved Upper and Lower Bounds for the Close Enough Traveling Salesman Problem,” International Conference on Green, Pervasive, and Cloud Computing, Springer, Cham, 2017, pp. 165-177, doi: https://doi.org/10.1007/978-3-319-57186-7_14.
[3] F. Carrabs, C. Cerrone, R. Cerulli, and M. Gaudioso, “A novel discretization scheme for the close enough traveling salesman problem,” Computers & Operations Research, 78, 2017, pp. 163-171, doi: https://doi.org/10.1016/j.cor.2016.09.003.
[4] B. Behdani, and J. C. Smith, “An integer-programming-based approach to the close-enough traveling salesman problem,” INFORMS Journal on Computing, 26(3), 2014, pp. 415-432, doi: https://doi.org/10.1287/ijoc.2013.0574.
[5] M. Karpinski, and R. Schmied, “On approximation lower bounds for TSP with bounded metrics”, Electron. Colloquium Comput. Complex 19, 2012, pp. 8, http://arxiv.org/abs/1201.5821.
[6] I. O., Bercea, ”Improved Bounds for the Traveling Salesman Problem with Neighborhoods on Uniform Disks”, CCCG, 2018, http://arxiv.org/abs/1809.07159.
[7] W. P. Coutinho, R. Q. do Nascimento, A. A. Pessoa, and A. Subramanian, ”A Branch-and-Bound Algorithm for the Close-Enough Traveling Salesman Problem,” INFORMS Journal on Computing 2016, 28, issue 4, pp. 752-765, doi: https://doi.org/10.1287/ijoc.2016.0711
[8] J. Faigl, “GSOA: growing self-organizing array-unsupervised learning for the close-enough traveling salesman problem and other routing problems,” Neurocomputing, 2018, 312, pp. 120-134, doi: https://doi.org/10.1016/j.neucom.2018.05.079

Name and workplace of master’s thesis supervisor:

prof. Ing. Jan Faigl, Ph.D., Artificial Intelligence Center, FEE

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **12.02.2021** Deadline for master's thesis submission: **21.05.2021**

Assignment valid until: **30.09.2022**

prof. Ing. Jan Faigl, Ph.D. Supervisor’s signature	Head of department’s signature	prof. Mgr. Petr Páta, Ph.D. Dean’s signature

III. Assignment receipt

The student acknowledges that the master’s thesis is an individual work. The student must produce her thesis without the assistance of others, with the exception of provided consultations. Within the master’s thesis, the author must state the names of consultants and include a list of references.

. Date of assignment receipt	Student’s signature

"Started making it. Had a breakdown. Bon appetite."

-JAMES ACASTER, THE GREAT CELEBRITY
BAKE OFF FOR SU2C (SEASON 2, EP. 2)

Declaration

I declare that the presented work was developed independently and that I have listed all sources of the information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of the university thesis.

Prague, May 21, 2021

. . . .

Bc. Jindřiška Deckerová

Acknowledgement

I would like to thank prof. Ing. Jan Faigl, Ph.D. for guidance during writing this thesis, Ing. Petr Váňa for his help and cooperation in the research presented in this thesis and related publications. Also, I would like to thank my friends and family for their support during these challenging times of my thesis writing.

Abstrakt

V této práci studujeme složité zobecněné směrovací problémy motivované robotickými úlohami jako je plánování pro robotické manipulátory, sběr dat nebo osvětlení objektů autonomními vzdušnými prostředky. Studované směrovací problémy, ve kterých je cílem nalézt cestu navštěvující množinu daných regionů, můžeme považovat za variantu úlohy obchodního cestující s okolími (TSPN). Řešením TSPN je nejkratší uzavřená cesta spojující všechny daná okolí (regiony). Okolí mohou mít různé tvary - konvexní, nekonvexní, překrývající se, nepřekrývající se, a přesná řešení TSPN jsou známa pouze pro speciální typy okolí. Z tohoto důvodu je obtížné porovnat metody řešení zobecněných problémů a ohodnotit kvalitu řešení vzhledem k optimu. Proto v práci využíváme výpočtu dolní a horní meze hodnoty optimálního řešení a stanovení relativní odchylky od optima. V této práci jsme navrhli zobecněnou metodu větví a mezí, abychom získali dolní a horní meze různých obecných problémů, kontrétně problému obchodního cestujícího s dostatečně blízkým okolím (CETSP), problému obchodního cestujícího s okolími na sféře (TSPNS), a zobecněného problému obchodního cestujícího s okolími (GTSPN). Empirické vyhodnocení naznačuje, že dolní meze pro CETSP vypočtené prezentovanou metodou dosahují podobné kvality jako stávající meze. Pro TSPNS a GTSPN neexistují žadné dolní meze, proto předkládáme první dolní meze na optimálním řešení těchto složitých problémů. Dále problémy řešíme metodou *učení bez učitele* rostoucího samo-organizujícího se pole (GSOA). GSOA poskytuje kvalitní řešení v krátkém výpočetním čase a v několika případech GSOA překonává stávající metody.

Klíčova slova: Dolní mez, Metoda větví a mezí, Učení bez učitele, Problém obchodního cestujícího s dostatečně blízkými okolími, Problém obchodního cestujícího s okolími na sféře, Zobecněný problém obchodního cestujícího s okolími

Abstract

This thesis focuses on studying complex generalized routing problems arising from practical applications in robotics, such as robotic manipulator tasks, data collection missions, and object illumination by aerial vehicles. These problems can be considered tour construction problems, and thus variants of the well-known Traveling Salesman Problem with Neighborhoods (TSPN). The TSPN stands to determine the shortest tour visiting each of the given neighborhoods. The neighborhoods can be of various shapes, convex or non-convex, overlapping or non-overlapping, and exact solutions are known only for specific cases. Hence, it is difficult to compare and evaluate new approaches to general problem variants because of a lack of optimal solutions. Thefore, a lower and upper bounds values on the optimal solution value of a particular problem instance can be determined to estimate a relative optimality gap. We develop the Branch-and-Bound method to obtain the lower bound and upper bound values of various generalized routing problems, namely the Close Enough Traveling Salesman Problem (CETSP), the Traveling Salesman Problem with Neighborhoods on a Sphere (TSPNS), and the Generalized Traveling Salesman Problem with Neighborhoods (GTSPN). Regarding the reported empirical evaluation and comparison of the proposed lower bounds with the existing lower bounds for a particular problem of the CETSP, the results indicate the provided lower bounds are of similar quality. For the TSPNS and the GTSPN, such lower bound estimates have not yet been published in the literature (to the best of our knowledge); hence, we present the first lower bound values on the optimal solution of the TSPNS and GTSPN. Further, we address the studied problems by the proposed modifications of the unsupervised learning Growing Self-Organizing Array (GSOA) to address all of the studied problems. The computationally efficient GSOA is evaluated using the relative optimality gap and provides solutions of competitive quality, and in several cases, the GSOA outperforms the existing methods.

Keywords: Lower bounds, Branch and Bound, Unsupervised learning, Close Enough Traveling Salesman Problem, Traveling Salesman Problem with Neighborhoods on a Sphere, Generalized Traveling Salesman Problem with Neighborhoods

Contents

List of Figures

List of Tables

List of Algorithms

List of Models

Abbreviations

BeFS	Best-First Search
BFS	Breadth-Firsth Search
BNB	Branch-and-Bound
CETSP	Close Enough Traveling Salesman Problem
DFS	Depth-First Search
GSOA	Growing Self-Organizing Array
GTSP	Generalized Traveling Salesman Problem
GTSPN	Generalized Traveling Salesman Problem with Neighborhoods
HRGKA	Hybrid Random-Key Genetic Algorithm
LP	Linear Programming
MILP	Mixed Integer Linear Programming
MIP	Mixed Integer Programming
MIQCP	Mixed Integer Quadratically Constrained Programming
MTP	Multi-goal Path Planning
NLP	Non-Linear Programming
$(lb/ub)Alg$	Adaptive algorithm
PTAS	Polynomial-Time Approximation Schema
SOCP	Second-Order Cone Programming
SZ2	Steiner-Zone based algorithm
TSP	Traveling Salesman Problem
TSPN	Traveling Salesman Problem with Neighborhoods
TSPN-LKH	Lin-Kernighan heuristic for the TSPN
TSPN-GLKH	Generalized Lin-Kernighan for the TSPN
TSPNS	Traveling Salesman Problem with Neighborhoods on a Sphere

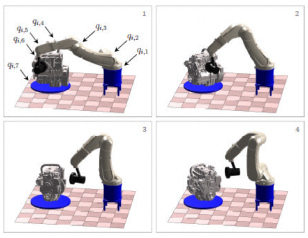
Used Symbols

S_i	i-th target region
$\mathcal{S}$	Set of n target regions S_i
d	Dimension of input space
C_i	Center of region S_i
δ_i	Sensing radius associated with target region S_i
$\mathcal{Q}$	Sphere
Q	Center of sphere $\mathcal{Q}$
δ	Radius of sphere $\mathcal{Q}$
$\boldsymbol{c}_i$	Lighting direction vector associated with target region S_i
Ω_i	Solid angle associated with target region S_i
$\mathcal{R}_{i,k}$	k-th regions of neighborhood set S_i
$\boldsymbol{A}, \boldsymbol{b}$	Matrix and vector defining polyhedron
$\boldsymbol{P}$	Symmetric positive definite matrix defining ellipsoid
$\\|S_i - S_j\\|$	Euclidean distance between target locations S_i and S_j
$\mathcal{L}_g(\boldsymbol{x}_i, \boldsymbol{x}_j)$	Length between vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ on sphere
P_i	Waypoint location associated with target S_i
$\mathcal{P}$	Set of waypoint locations/vectors
Σ	Sequence of visits to target regions
σ_i	i-th visit to target regions
$\mathcal{L}(\Sigma, \mathcal{P})$	Value of solution $(\Sigma, \mathcal{P})$
$\mathcal{LB}(\Sigma, \mathcal{P})$	Lower bound value on solution $(\Sigma, \mathcal{P})$
$\mathcal{UB}(\Sigma, \mathcal{P})$	Upper bound value on solution $(\Sigma, \mathcal{P})$
$\mathcal{N}$	Array of nodes in GSOA
σ, α, μ	Learning parameters of GSOA
M	Number of nodes of $\mathcal{N}$
ν	Node of $\mathcal{N}$
ν^*	Winner node
$\nu.N$	Location of node ν
$\nu.P$	Waypoint location associated with node ν
$\nu.\boldsymbol{n}$	Location vector of node ν
$\mathcal{M}$	Solution space of BNB
$t_{\max}$	Timeout of BNB
μ	Node of solution space $\mathcal{M}$
$\mu.\mathcal{LB}, \mu.\mathcal{UB}$	Lower and upper bounds associated with μ
$\mu.\text{exact}$	Indication a solution associated with μ is exact
$\mathcal{O}$	Open list
B	Set of not covered regions
$\boldsymbol{f}$	Variable representing length of path between two consecutive points
$\boldsymbol{x}$	Continuous variable representing waypoints
$\boldsymbol{w}, \boldsymbol{v}$	Auxilary variable representing difference in coordinates and vectors
$\boldsymbol{B}$	Binary variable for *big M*
M	*Big M* variable
$\mathcal{L}_{ref}$	Best solution value among all solvers
$\mathcal{L}$	Best solution value among performed trials by a particular solver
$\mathcal{L}_{mean}$	Mean solution value among performed trials by a particular solver

Chapter 1

Introduction

Routing problems are combinatorial problems that can be classified into two categories: the pathfinding problems and tour construction problems [1]. The first category denotes planning problems to determine single-source shortest paths. The second category of the routing problems includes a set of problems to determine a tour that visits multiple targets within a given network. The herein studied generalizations of the well-known combinatorial routing problem, the Traveling Salesman Problem (TSP), fall into the second category of the tour construction problems. Having a set of locations, the TSP stands to determine the cost-efficient closed path, such that it visits each of the locations. Thus, the combinatorial optimization of the TSP is to determine the optimal sequence of visits to the given locations. Consequently, we can study the TSP and its variants in robotics as the robot sequencing problems [2], see Fig. 1.1(a), or with aerial vehicles [3] such as shown in Fig. 1.1(b). The robotic variant of the TSP is referred to as the Multi-goal Path Planning (MTP) [2] because it inherently includes a solution of the path planning problem for a robot that needs to satisfy the robot motion constraints.


(a) Robotic manipulator for vision inspection [4].

(b) An UAV in a data collection mission.

(c) Object illumination by a team of UAVs [5].

Figure 1.1: Examples of applications of routing problems in robotics.

Furthermore, in robotic routing, it might not be required to visit the defined target locations precisely. Instead of that, it might be beneficial to exploit the possibility to visit the targets remotely, such as a customer visit by a utility company to read data distantly using wireless radio-based transmission [6] or to photograph a distant object with a constant illumination [5] shown in Fig. 1.1(c). Thus, we can consider locations as regions and exploit a degree of freedom to save the total travel cost. Such a variant of the TSP is called the TSP with Neighborhoods (TSPN), where the neighborhoods can be represented as continuous regions [6–10] or as complex clustered regions motivated by tasks for robotic manipulators and data collection missions [11–13].

The problems with neighborhoods can be addressed by various existing heuristics and exact methods [11, 14–23]. The heuristics vary from approximation heuristic and soft computing techniques to sampling-based approaches using discretization of the continuous neighborhoods. Heuristics are designed to provide solutions to particular types of problem instances of sufficient quality with respect to the required computational time that is to be lower than time required by the exact methods because the underlying TSP is known to be NP-hard [24]. It is worth noting that even an optimal solution of problems with discretized neighborhoods is a solution of the approximate subproblem, not the optimal solution of the original continuous problem.

Optimal solutions to routing problems can be obtained by the exact methods [11, 19]. Having the optimal solution to a particular problem, we can study the problem in more detail; however, some types of robotic routing problems are so complex that it is nearly impossible to obtain the optimal solution. Therefore, we aim to determine a tight interval defined by the lower and upper bounds on the optimal solution value of the particular problem. The lower and upper bounds can be utilized in the evaluation of the solvers for routing problems. In addition, a relative optimality gap can be established for a particular solution with the known tight value of the lower bound of the same problem instance. Several methods to determine lower bounds of the TSP with the continuous neighborhoods can be found in the literature [18, 19, 25–27].

In this thesis, we study the generalized routing problems by estimating the lower bounds using the Branch-and-Bound (BNB) method. In particular, to the best of our knowledge, we provide the first baseline methods for lower bound estimations of several studied problems, namely the Traveling Salesman Problem with Neighborhoods on a Sphere (TSPNS) and the Generalized Traveling Salesman Problem with Neighborhoods (GTSPN). Moreover, we address each of the studied problems by a relatively fast heuristic based on the unsupervised learning method of the Growing Self-Organizing Array (GSOA) [22]. Then, the proposed and existing solvers of the studied routing problems are empirically evaluated utilizing the established lower bound values and compared with the state-of-the-art approaches. The estimated lower bound values are also compared with the existing lower bounds, which are available only for a few of the herein studied problems, in particular, the CETSP. Regarding the presented results, in most cases, the proposed GSOA-based approaches provide competitive solutions to the existing solvers but with lower computational requirements. Note that most of the reported results have been already published in [10, 23, 28] during the work on the thesis.

The thesis is organized as follows. State-of-the-art approaches to the studied generalized routing problems are briefly described in Chapter 2. The notation, together with the specific problem formulations, is presented in Chapter 3. The BNB method for the lower bounds estimation of the generalized routing problems is presented in Chapter 4. The heuristic approaches based on the GSOA are in Chapter 5. Empirical evaluation results on the proposed GSOA-based approaches and the evaluation of the existing approaches for particular problems based on the lower bounds are summarized in Chapter 6. Finally, the thesis is concluded in Chapter 7.

Chapter 2
Related Work

The herein studied generalized routing problems are considered a subgroup of tour construction problems referred to as the Traveling Salesman Problem with Neighborhoods (TSPN). The TSPN stands to determine the optimal sequence of visits to the neighborhoods together with particular points of visits to the neighborhoods. Thus, the TSPN combines combinatorial optimization to determine the sequence of visits with continuous optimization in determining the points of visits. It is known the TSPN is APX-hard, unless P=NP [16]. There are many approaches to solve the TSPN such as approximation schemata [14–17], exact methods [11,18,19], decoupled approaches [20, 21, 23], sampled-based techniques [10, 20], and also neural networks based learning methods [10, 22, 23]; each approach proposed to address the TSPN with a particular type of the neighborhoods.

The approximation schemata and Polynomial-Time Approximation Schema (PTAS) [14–17] are algorithms that for a given ϵ can approximate the problem within a factor $1 + \epsilon$ in polynomial time, and in some cases even with a constant factor [29]. In [16], the authors prove that the TSPN can be approximated within the factor 391/390. One of the very first constant approximation schemata is proposed in [15], where the authors select a point to represent the neighborhood and then apply a known approximation algorithm for the TSP. The selected representative points are parallel segment units to achieve the constant factor approximation. The work [15] has been later extended in [14] to address the connected neighborhoods with a similar diameter. The neighborhoods represented as disjoint unit disks are addressed by the PTAS algorithm based on the m-guillotine structure. Further work regarding the approximation algorithms is presented in [16], where the authors introduce the constant factor approximation algorithm for disjoint convex neighborhoods. The authors of [17] address the continuous neighborhoods and the neighborhoods represented by a set of points. The approximation algorithms solve the TSPN in polynomial time; however, the existing algorithms are specifically designed for particular neighborhood types. Besides, they are parameter sensitive and also sensitive to the representation of the neighorboods [15].

The TSPN can be solved optimally by formulating the problem as Linear Programming (LP) model. In [11], the authors address polygonal neighborhood and formulate the TSPN as the Mixed Integer Linear Programming (MILP) model. Although these approaches might provide the optimal solutions, they are computationally very demanding, and solutions are provided in reasonable computational time only for relatively small instances.

Less computationally demanding and also less parameter sensitive are heuristic algorithms such as decoupled approach where a sequence of visits is determined before points of visits [21]. There are several soft computing techniques for the TSPN. The one based on unsupervised learning of the self-organizing maps [30] is of our particular interest because it has been already deployed in several robotic routing problems [9, 22, 31, 32].

There are many variants of the TSPN based on the particular type (shape) of the neighborhood. In this thesis, we focus on variants of the TSPN, where neighborhoods are represented as disks both in the Euclidean space and on a sphere, and variants with neighborhoods represented as groups of convex shapes. The variant of the TSPN with disk-shaped neighborhoods in the Euclidean space is known in the literature as the Close Enough TSP (CETSP) [6], and an example of the problem instance with its solution is depicted in Fig. 2.2. The CETSP

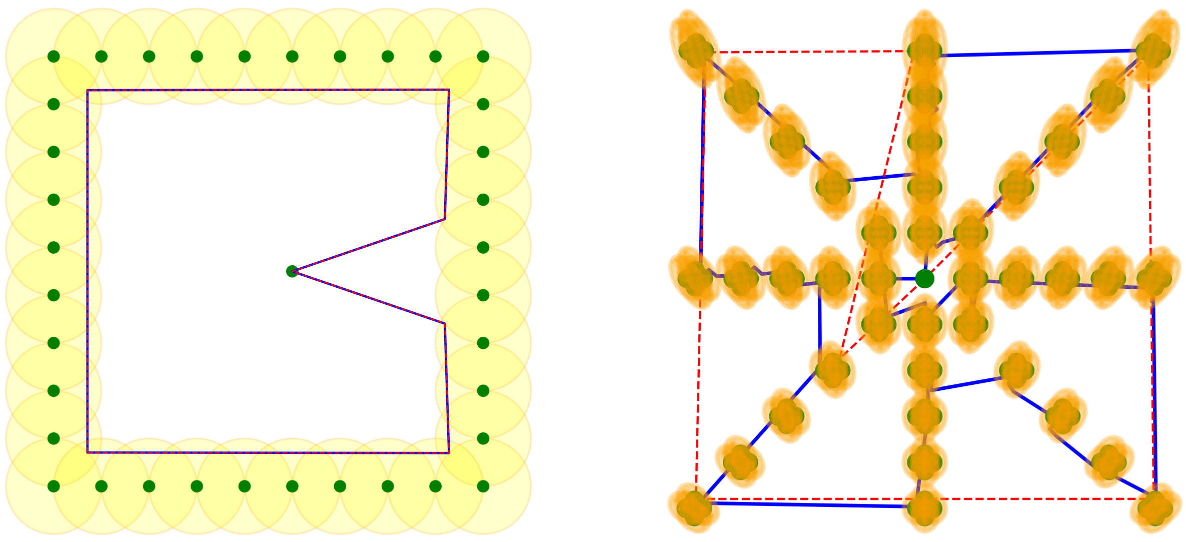

(a) Example of the CETSP instance. (b) Example of the 3D CETSP instance.

Figure 2.2: Example of the TSP with disk-shaped neigborhoods together with the solution (blue) and the lower bound (red). For the instance in Fig. 2.2(a), the lower bound coincides with the solution.

has been introduced in [6] motivated by collecting utility data via wireless communication. The authors propose a method based on supernodes determination. The CETSP is further addressed in [7, 33] and thoroughly studied in [20]. The author of [20] proposed several approaches to address the continuous neighborhoods based on the formulation of the CETSP as the Mixed Integer Programming (MIP) model, the discretization of the continuous neighborhoods, the genetic algorithm, and Steiner Zones, i.e., intersections of convex regions, that provide the best trade-off between the solution quality and computational requirements. Moreover, the author presented a set of benchmark instances for the CETSP that support empirical evaluation of further approaches.

Steiner Zones are further exploited in the Variable Neighborhood Search (VNS) based approach [34] to decrease the computational time. In [35], the authors address continuous neighborhoods by determining the point of visit based on the neighboring regions. The reported results show that the provided solutions are competitive with the Steiner Zones-based approaches. However, the approach is outperformed by the Growing Self-Organizing Array (GSOA) [22] that originates from Self-Organizing Maps to the TSP [36]. The GSOA-based approach is similar to [35]; the method simultaneously determines the sequence of visits with the exact point of visits. A GSOA-based solution of the multidimensional CETSP is presented in [28], see an instance example and its solution depicted in Fig. 2.2(b).

A variant of the CETSP, where the disk-shaped neighborhoods are on a sphere, is called the TSPN on a Sphere (TSPNS), and it is motivated by the requirement of the constant distance object illumination [10] in reflectance transformation imaging tasks. An example of a problem instance is depicted in Fig. 2.3(a). The TSPNS can also be addressed by a variant of the GSOA or by the sampling-based approach, where the neighborhoods are sampled, and the problem is transformed to the Generalized TSP (GTSP).

The GTSP can be considered a variant of the TSPN with neighborhoods strictly formed by discrete points, or the continuous neighborhoods are sampled into sets of discrete points [37]. The problem stands to determine the cost-efficient path visiting one point of each set of points. Various approaches have been proposed to address the GTSP, such as meta-heuristics [38] or genetic algorithms [39]. Besides, the GTSP can be directly transformed to the Asymmetrical

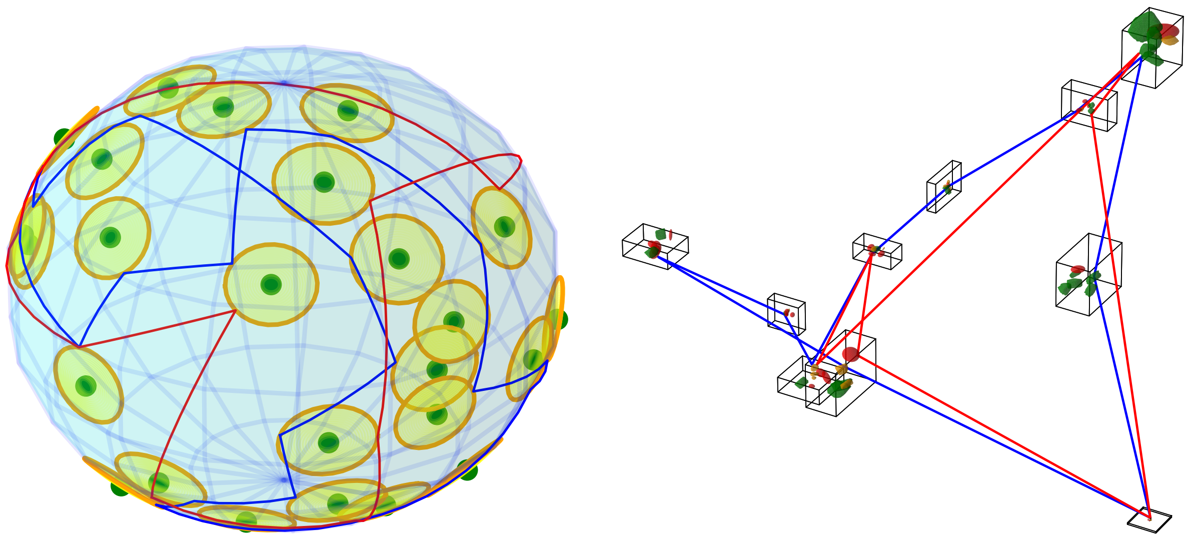

(a) Example of the TSPNS instance. **(b)** Example of the GTSPN instance.

Figure 2.3: Example of the TSP with various, more complex neigborhoods together with the solution (blue) and the lower bound (red).

TSP [40], and thus solved as a regular TSP at the cost of increased problem size. Furthermore, two heuristics solvers to the GTSP have been proposed relatively recently [41, 42].

The last variant of the TSP studied in this thesis is a generalization of the TSPN, where neighborhoods are considered sets of possibly overlapping and non-convex regions. The variant called the Generalized TSPN (GTSPN) is introduced in [12] as a combination of the GTSP and TSPN to address limitations of the discrete target locations of the GTSP and generalize single continuous sets of the TSPN. An example of the problem instance is depicted in Fig. 2.3(b). The authors of [12] proposed to adapt the Hybrid Random-Key Genetic Algorithm (HRGKA) [43] to address the complex GTSPN. The regions of the addressed GTSPN are non-overlapping sets of convex possibly overlapping regions represented as polyhedrons, ellipsoids, or a combination of both, where the particular shape of the neighborhood regions is motivated by the real-world application. In [23], the GTSPN has been addressed by the unsupervised learning of the GSOA. Besides, a decoupled approach has been proposed based on transformation to an instance of the GTSP using centroids of the regions followed by local optimization to find the particular points of visits to the sets.

2.1 Lower and Upper Bound Values

Regarding the existing approaches to estimate the lower and upper bound values on the optimal solutions of variants of the TSP and the TSPN, the lower bounds of the TSP are addressed by several approaches [44–46]. On the other hand, the TSPN is more complex and so far less studied. The most studied variant of the TSPN regarding the lower bounds is the Close Enough TSP (CETSP) [6], i.e., the TSPN with disk-shaped neighborhoods. In [20], among several approaches to the CETSP, the author presents a quick determination of the lower bound value on the optimal solution of the CETSP that is based on the simplification of the problem using distances between disk-shaped neighborhoods if neighborhoods do not overlap; otherwise, the distance is set to zero. However, the estimated lower bound values are relatively poor and do not provide a good quality indicator. The lower and upper bound values can be improved by solving the problem exactly. The authors of [18] propose to formulate the CETSP as the Mixed Integer Programming (MIP) model based on the partitioning of the neighborhoods and derive the lower and upper bound values from the obtained solution of MIP model.

The lower and upper bounds of [18] are further improved in [19], where the authors propose the exact Branch-and-Bound (BNB) method with the Second-Order Cone Programming (SOCP) formulation of the CETSP to obtain the lower and upper bound values. The BNB method provides the optimal solution for all problem instances presented in [18] and several instances presented in [20], and thus, establishes new benchmark solutions. For the remaining instances studied in [20], the method [19] finds improved lower bound values.

Heuristic to establish the lower bound values on the optimal solution of the CETSP is proposed in [25] based on the graph reduction algorithm that reduces the size of the problem instance and applies the discretization schema to partition the continuous neighborhoods. The algorithm is further improved in [26] using the SOCP model. Further, the discretization schema and the SOCP-based heuristic are improved by a new adaptive method in [27]. A mathematical approach is adapted in [47] to prove the tightness of the bounds for small disk-shaped neighborhoods.

Although several approaches to the lower bound values of the CETSP have been recently proposed, to the best of the authors' knowledge, there are not similar results available for the herein studied TSPNS and GTSPN. Therefore, we propose to address the lower bounds values on the optimal solutions of these generalized routing problems together with the solution of the problems.

Chapter 3

Problem Statement

In this thesis, we study generalized routing problems, specifically the Close Enough Traveling Salesman Problem (CETSP) with disk-shaped neighborhoods, the Traveling Salesman Problem with Neighborhoods on a Sphere (TSPNS) with disk-shaped neighborhoods, and the Generalized Traveling Salesman Problem with Neighborhoods (GTSPN) with non-overlapping sets of convex (possibly overlapping) neighborhoods, i.e., the neighborhoods are allowed to overlap within the set. Each of these problems is a variant of the Traveling Salesman Problem with Neighborhoods (TSPN) with a particular neighborhood type. Thus, we first formally introduce the TSPN in Section 3.1. The formal definitions of the particular problems are presented in Section 3.2, and used terms are summarized in Section 3.3.

3.1 Traveling Salesman Problem with Neighborhoods (TSPN)

The Traveling Salesman Problem with Neighborhoods (TSPN) is to find the most cost-efficient path to given regions. Having a set of n target regions $\mathcal{S} = \{S_1, \ldots, S_n\}$, the TSPN stands to determine the closed path visiting each of the target regions such that the cost of the path is minimal. Thus, we need to determine a sequence of visits to the target regions, denoted as $\Sigma = (\sigma_1, \ldots, \sigma_n)$, where $\sigma_i \in \{1, \ldots, n\}$ and $\sigma_i \neq \sigma_j$ for $i \neq j$, and a set of exact points of visits to the target regions denoted as waypoint locations $\mathcal{P} = \{P_1, \ldots, P_n\}$, represented as points $P_i \in \mathbb{R}^d$ for which (4) holds. The cost of travel from the waypoint P_i to the waypoint P_j is considered as the Euclidean distance between P_i and P_j denoted as $\|P_i - P_j\|$. The TSPN is then formulated as Problem 3.1.1.

Problem 3.1.1 (Traveling Salesman Problem with Neighborhoods (TSPN))

$$\underset{\Sigma,\mathcal{P}}{\text{minimize}} \quad \mathcal{L}(\Sigma, \mathcal{P}) = \|P_{\sigma_n} - P_{\sigma_1}\| + \sum_{i=1}^{n-1} \left\|P_{\sigma_i} - P_{\sigma_{i+1}}\right\| \tag{1}$$

s.t.

$$\Sigma = (\sigma_1, \ldots, \sigma_n), \quad \sigma_i \in \{1, \ldots, n\}, \sigma_i \neq \sigma_j \text{ for } i \neq j, \forall i, j \in \{1, \ldots, n\} \tag{2}$$

$$\mathcal{P} = \{P_1, \ldots, P_n\}, \quad P_i \in \mathbb{R}^d, \forall i \in \{1, \ldots, n\} \tag{3}$$

$$P_i \in S_i, \quad \forall i \in \{1, \ldots, n\} \tag{4}$$

3.2 Studied Variants of the TSPN

The particular problem formulations of the CETSP, TSPNS, and GTSPN specify the formulation of the TSPN depicted in Problem 3.1.1. In particular, the constraint ensuring each waypoint P_i is within the particular target region S_i (4). The constraint is specified depending on the particular neighborhood type; hence we present three variants of the problem formulation.

3.2.1 Close Enough Traveling Salesman Problem (CETSP)

The Close Enough TSP (CETSP) [6] is a variant of the TSPN with disk-shaped regions. Having a set of n regions $\mathcal{S} = \{S_1, \ldots, S_n\}$, each S_i defined by its center $C_i \in \mathbb{R}^d$ ($d \in \{2, 3\}$ in our case), and the corresponding radius δ_i, the CETSP stands to determine a sequence of visits $\Sigma = (\sigma_1, \ldots, \sigma_n)$, where $\sigma_i \in \{1, \ldots, n\}$ and $\sigma_i \neq \sigma_j$ for $i \neq j$, and a set of waypoint locations $\mathcal{P} = \{P_1, \ldots, P_n\}$. Each waypoint location P_i is within the δ_i distance from the center C_i of the corresponding target region S_i (8). An instance of the CETSP with the notation is depicted in Fig. 3.4. The CETSP can be formulated as Problem 3.2.1.

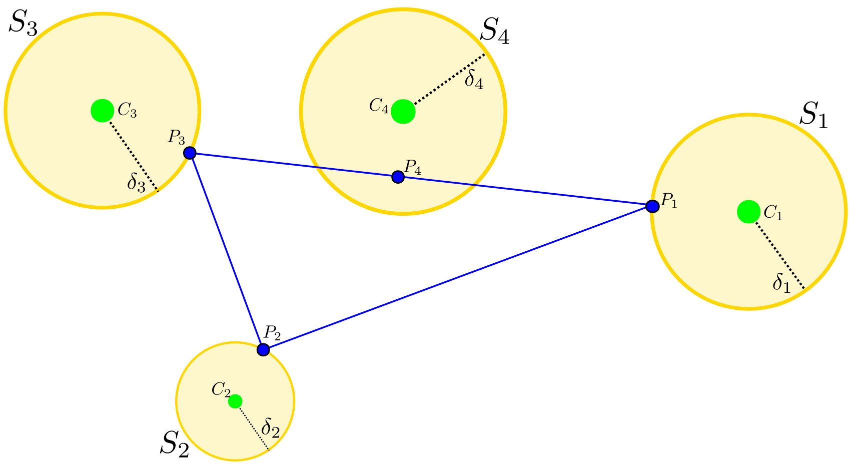


Figure 3.4: An instance of the CETSP with $n = 4$ disk-shaped target regions $\mathcal{S} = \{S_1, S_2, S_3, S_4\}$. Each region S_i (yellow) is defined by its center C_i (green) and radius δ_i. The waypoints $\mathcal{P} = \{P_1, P_2, P_3, P_4\}$ with $\Sigma = (1, 4, 3, 2)$ form a solution (blue) of the instance.

Problem 3.2.1 (Close Enough TSP (CETSP))

$$\underset{\Sigma, \mathcal{P}}{\text{minimize}} \quad \mathcal{L}(\Sigma, \mathcal{P}) = \|P_{\sigma_n} - P_{\sigma_1}\| + \sum_{i=1}^{n-1} \left\|P_{\sigma_i} - P_{\sigma_{i+1}}\right\| \tag{5}$$

s.t.

$$\Sigma = (\sigma_1, \ldots, \sigma_n), \quad \sigma_i \in \{1, \ldots, n\}, \sigma_i \neq \sigma_j \text{ for } i \neq j, \forall i, j \in \{1, \ldots, n\} \tag{6}$$

$$\mathcal{P} = \{P_1, \ldots, P_n\}, \quad P_i \in \mathbb{R}^d, \forall i \in \{1, \ldots, n\}, d \in \{2, 3\} \tag{7}$$

$$\|C_i - P_i\| \leq \delta_i, \quad \forall i \in \{1, \ldots, n\} \tag{8}$$

3.2.2 Traveling Salesman Problem with Neighborhoods on a Sphere (TSPNS)

The Traveling Salesman Problem with Neighborhoods on a Sphere (TSPNS) [10] can be considered a variant of the CETSP with the disk-shaped target regions on a sphere. The sphere $\mathcal{Q}$ is given by its center $Q \in \mathbb{R}^3$ and radius $\delta > 0$, see Fig. 3.5. Each disk-shaped target region S_i on the sphere is considered to be a spherical cap defined by the required direction vector $\boldsymbol{c}_i$[1] and the solid angle Ω_i that represents the allowed tolerance to the direction $\boldsymbol{c}_i$. The angle between two vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is defined as $\angle(\boldsymbol{x}_i, \boldsymbol{x}_j) = \arccos(\boldsymbol{x}_i \cdot \boldsymbol{x}_j)$. Having a normalized

[1]E.g., required lighting direction in reflectance transformation imaging tasks [10].

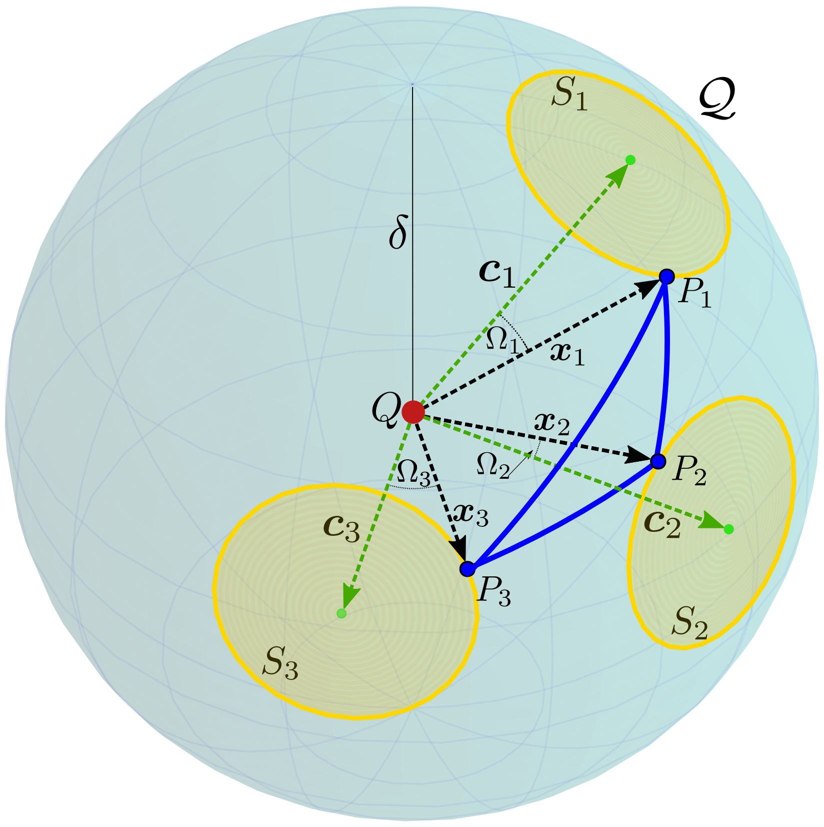


Figure 3.5: An instance of the TSPNS with $n = 3$ target regions $\mathcal{S} = \{S_1, S_2, S_3\}$ on the sphere $\mathcal{Q}$. Each region $\mathcal{S}_i$ (yellow) is defined by the vector denoting center $\boldsymbol{c}_i$ (green) and solid angle Ω_i. The waypoint vectors $\mathcal{P} = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3\}$ with $\Sigma = (1, 3, 2)$ form a solution (blue) of the instance.

vector $\boldsymbol{x}$ originating from Q, the target region S_i is defined as a set of all points X that are within δ distance from Q under the angle Ω_i:

$$S_i = \{X \,|\, X = Q + \delta\boldsymbol{x} \ \&\& \ \|\boldsymbol{x}\| = 1 \ \&\& \ \angle(\boldsymbol{x}, \boldsymbol{c}_i) \le \Omega_i\}. \tag{9}$$

Then, the TSPNS stands to determine a sequence of visits $\Sigma = (\sigma_1, \ldots, \sigma_n)$, $\sigma_i \in \{1, \ldots, n\}$, $\sigma_i \neq \sigma_j$ for $i \neq j$, together with a set of waypoint locations $\mathcal{P}$, where the locations are represented by the normalized vectors $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ from Q as $P_i = Q + \delta\boldsymbol{x}_i, \|\boldsymbol{x}_i\| = 1$, that are within the solid angle Ω_i from $\boldsymbol{c_i}$, i.e., $\angle(\boldsymbol{x}_i, \boldsymbol{c}_i) \le \Omega_i$. The distance between two points on the sphere P_i and P_j is defined by the representing normalized vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ as

$$\mathcal{L}_g(P_i, P_j) = \delta\,\angle(\boldsymbol{x}_i, \boldsymbol{x}_j) = \delta \arccos(\boldsymbol{x}_i \cdot \boldsymbol{x}_j). \tag{10}$$

The TSPNS can be formulated as Problem 3.2.2.

Problem 3.2.2 (TSP with Neighborhoods on a Sphere (TSPNS))

$$\underset{\Sigma,\mathcal{P}}{\text{minimize}} \quad \mathcal{L}(\Sigma,\mathcal{P}) = \mathcal{L}_g(P_{\sigma_n}, P_{\sigma_1}) + \sum_{i=1}^{n-1} \mathcal{L}_g(P_{\sigma_i}, P_{\sigma_{i+1}}) \tag{11}$$

s.t.

$$\Sigma = (\sigma_1, \ldots, \sigma_n), \quad \sigma_i \in \{1, \ldots, n\}, \sigma_i \neq \sigma_j \text{ for } i \neq j, \forall i, j \in \{1, \ldots, n\} \tag{12}$$

$$\mathcal{P} = \{P_1, \ldots, P_n\}, \quad \boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}, P_i \in \mathbb{R}^3, \boldsymbol{x}_i \in \mathbb{R}^3, \forall i \in \{1, \ldots, n\} \tag{13}$$

$$P_i = Q + \delta \boldsymbol{x}_i, \ \|\boldsymbol{x}_i\| = 1, \ \angle(\boldsymbol{x}_i, \boldsymbol{c}_{\sigma_i}) \leq \Omega_{\sigma_i} \quad \forall i \in \{1, \ldots, n\} \tag{14}$$

3.2.3 Generalized Traveling Salesman Problem with Neighborhoods (GTSPN)

The Generalized Traveling Salesman Problem with Neighborhoods (GTSPN) [12] is a generalization of the TSPN, where the target regions $\mathcal{S}$ are represented as neighborhood sets. Each neighborhood set S_i consists of m_i convex regions $\mathcal{R}_{i,k}$, $k \in \{1, \ldots, m_i\}$, and each region $\mathcal{R}_{i,k}$ is defined by its center $C_{i,k} \in \mathbb{R}^d$ and the particular geometrical representation of its shape. The regions might overlap within the particular set. The GTSPN is to determine a sequence of visits $\Sigma = (\sigma_1, \ldots, \sigma_n)$, $\sigma_i \in \{1, \ldots, n\}$, $\sigma_i \neq \sigma_j$ for $i \neq j$, together with a set of waypoint locations $\mathcal{P} = \{P_1, \ldots, P_n\}$, where each waypoint location $P_i \in \mathbb{R}^d$ is within at least one region $\mathcal{R}_{i,k}$ of the corresponding neighborhood set S_i (18). Note that, in this thesis, we consider $d = 3$ and $d = 7$. The GTSPN can be formally defined as Problem 3.2.3. An instance of the GTSPN together with its solution is depicted in Fig. 3.6.

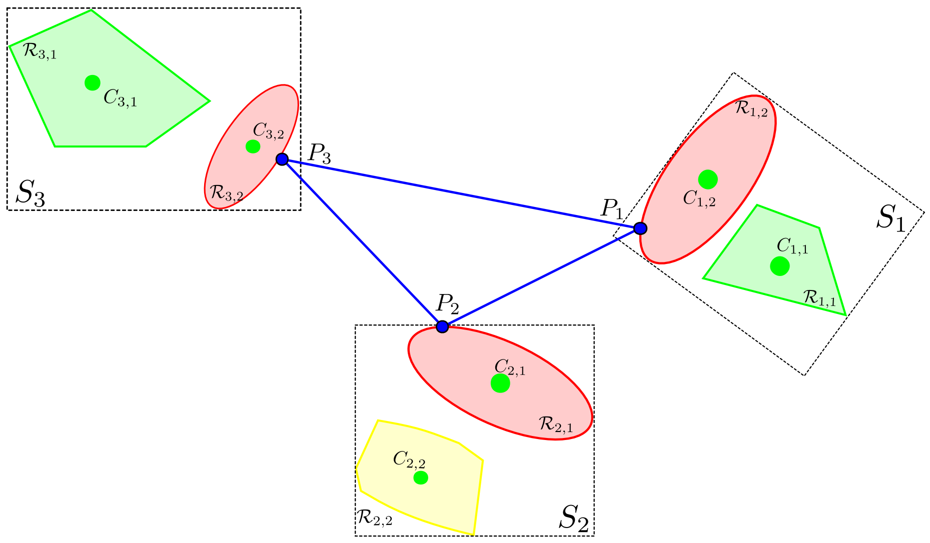


Figure 3.6: An instance of the GTSPN with $n = 3$ target regions $\mathcal{S} = \{S_1, S_2, S_3\}$, each represented as neighborhood set of $m = 2$ regions $\mathcal{R}$. Each region $\mathcal{R}_{i,k}$ - polyhedron (green), ellipsoid (red), hybrid (yellow) - is defined by its center $C_{i,k}$ (green) and all points satisfying the particular equations defining the region. The waypoint vectors $\mathcal{P} = \{P_1, P_2, P_3\}$ form a solution (blue) of the instance.

Problem 3.2.3 (Generalized TSP with Neighborhoods (GTSPN))

$$\underset{\Sigma,\mathcal{P}}{\text{minimize}} \quad \mathcal{L}(\Sigma,\mathcal{P}) = \|P_{\sigma_n} - P_{\sigma_1}\| + \sum_{i=1}^{n-1} \|P_{\sigma_i} - P_{\sigma_{i+1}}\| \tag{15}$$

s.t.

$$\Sigma = (\sigma_1, \ldots, \sigma_n), \quad \sigma_i \in \{1, \ldots, n\}, \sigma_i \neq \sigma_j \text{ for } i \neq j, \forall i, j \in \{1, \ldots, n\}, \tag{16}$$

$$\mathcal{P} = \{P_1, \ldots, P_n\}, \quad P_i \in \mathbb{R}^d, \forall i \in \{1, \ldots, n\}, d \in \{3, 7\} \tag{17}$$

$$P_i \in \mathcal{R}_{i,k}, \quad \forall i \in \{1, \ldots, n\} \, \exists k \in \{1, \ldots, m_i\} \tag{18}$$

In [12], regions of the neighborhood sets are approximated by linear and quadratic constraints and are of three shapes: polyhedron shape, ellipsoid shape, and hybrid shape, i.e., the combination of ellipsoid and polyhedron.

Polyhedron The polyhedron-shaped regions $\mathcal{R}_{i,k}$ are defined by the set of l half-spaces represented as the matrix $\boldsymbol{A}_{i,k} \in \mathbb{R}^{d,l}$, the corresponding vector $\boldsymbol{b}_{i,k} \in \mathbb{R}^l$ ($l = 12$ for $d = 3$, and $l = 20$ for $d = 7$), and all points $X \in \mathbb{R}^d$ ($d \in \{3, 7\}$) satisfying the equation

$$\boldsymbol{A}_{i,k} X - \boldsymbol{b}_{i,k} \leq 0, \quad i \in \{1, \ldots, n\}, 1 \in \{1, \ldots, m_i\}. \tag{19}$$

Ellipsoid Each ellipsoid-shaped region $\mathcal{R}_{i,k}$ is defined by the symmetric positive definite matrix $\boldsymbol{P}_{i,k} \in \mathbb{R}^d$, its center $C_{i,k} \in \mathbb{R}^d$ and all points $X \in \mathbb{R}^d$ satisfying the equation

$$(X - C_{i,k})^T \boldsymbol{P}_{i,k} (X - C_{i,k}) \leq 1, \quad i \in \{1, \ldots, n\}, 1 \in \{1, \ldots, m_i\}. \tag{20}$$

Hybrid The hybrid-shaped neighborhoods are defined similarly as the ellipsoids by the symmetric positive definite matrix $\boldsymbol{P}_{i,k} \in \mathbb{R}^d$ and the center $C_{i,k} \in \mathbb{R}^d$, but the ellipsoid is cut of by $l = 6$ half-spaces (represented as the matrix $\boldsymbol{A}_{i,k} \in \mathbb{R}^{d,l}$ and the corresponding vector $\boldsymbol{b}_{i,k} \in \mathbb{R}^l$). Thus both (19) and (20) holds for all points $X \in \mathbb{R}^d$.

3.3 Used Terms

In this section, we present a description of three terms widely used in the thesis. We define the terms regarding the lower and upper bound values. Having a solution $(\Sigma, \mathcal{P})$ of the instance of the particular routing problem consisting of the sequence of visits Σ and the set of visits to the targets $\mathcal{P}$, we can consider it a **feasible solution** or **optimal solution**. Besides, we further distinguish **lower bound**. The terms are defined as follows.

Feasible solution A solution $(\Sigma, \mathcal{P})$ is a feasible solution if the path formed by Σ and $\mathcal{P}$ visits each target region $S_i \in \mathcal{S}$.

Optimal solution A solution $(\Sigma, \mathcal{P})$ is the optimal solution if it is feasible solution with $\mathcal{L}(\Sigma, \mathcal{P})$ such that $\mathcal{L}(\Sigma, \mathcal{P}) \leq \mathcal{UB}(\Sigma, \mathcal{P})$, where $\mathcal{UB}(\Sigma, \mathcal{P})$ is the upper bound value on $(\Sigma, \mathcal{P})$, and there is no other feasible solution $(\Sigma', \mathcal{P}')$ with smaller length than $(\Sigma, \mathcal{P})$, i.e., $\mathcal{L}(\Sigma', \mathcal{P}') \geq \mathcal{L}(\Sigma, \mathcal{P})$.

Lower bound The tight lower bound value $\mathcal{LB}(\Sigma, \mathcal{P})$ on the optimal solution $(\Sigma, \mathcal{P})$ is a value such that $\mathcal{LB}(\Sigma, \mathcal{P}) \leq \mathcal{L}(\Sigma, \mathcal{P})$ and $\forall \mathcal{LB}'(\Sigma, \mathcal{P}) : \mathcal{LB}'(\Sigma, \mathcal{P}) < \mathcal{LB}(\Sigma, \mathcal{P})$.

Chapter 4

Lower Bounds on the Solutions of Generalized Routing Problems

The proposed approach in estimating the lower bound values of generalized routing problems is based on the Branch-and-Bound (BNB) algorithm. The BNB is a well-known combinatorial method to solve NP-hard combinatorial problems being adapted to particular problems. The core of the method is problem independent, and therefore, we present a brief overview of the BNB method [48] first. Then, we introduce the proposed BNB method based on [19] to address herein studied generalized routing problems, namely the Close Enough Traveling Salesman Problem (CETSP), the Traveling Salesman Problem with Neighborhoods on a Sphere (TSPNS), and the Generalized Traveling Salesman Problem with Neighborhoods (GTSPN), to estimate lower bounds on solutions of these problems.

The BNB method is used to solve a combinatorial problem represented as Linear Programming (LP) model. The model comprises of the objective function $f(\boldsymbol{x})$ and constraints $g(\boldsymbol{x})$, where $\boldsymbol{x} = (x_1, \ldots, x_d)$ denotes a set of variables. A solution of the model is obtained by assignment of values to variables $\boldsymbol{x} = (x_1, \ldots, x_d)$, such that the objective function $f(\boldsymbol{x})$ is minimized (or maximized) and all constraints $g(\boldsymbol{x})$ are satisfied. However, finding an exact solution to the problem by assigning all variables is computationally demanding. Therefore, the BNB method can be employed for problems, for which we can derive a set of subproblems of the original problem. Subproblems can be are created by assigning a value to the arbitrary variable of $\boldsymbol{x}$ and leaving the rest of the values to be assigned in further steps of the method. The subproblems and their solutions are referred to as the partial problems and partial solutions; the set of partial problems forms a solution space of the method and is represented as a tree graph. Thus, each partial problem with its solution can be represented as a tree node, where leaves contain feasible solutions to the problem. Since the solution space is represented as a tree graph, it can be searched by a graph search algorithm such as the Depth-First Search (DFS), the Breadth-First Search (BFS), or the Best-First Search (BeFS).

However, the searched solution space is large, and searching it can be computationally demanding. Moreover, searching infeasible subtrees of the solution space is undesirable. Hence two critical components, *branching* and *bounding*, are included in the method and form the BNB algorithm. The first critical part of the BNB is to create the child nodes of the most suitable nodes of the solution space, i.e., branch the nodes according to *branching rule*. The *branching rule* is a heuristic that takes as an input the partial problem and the partial solution associated with the current node and determines an arbitrary variable and a value we assign to the variable. Thus a new partial solution is created, and it is assigned to the child node of the current node. For the *bounding* part of the BNB, the solution values of each of the feasible partial problems of the solution space (partial solutions in leaves) are known, thus provide so far the best solution value. We can utilize the current best solution value to bound the other partial solutions by disregarding branches of the solution space that are not suitable for expansion.

Various heuristics and components can further modify the BNB to address the particular problems [48]. In the following section, we describe the proposed BNB to address the herein studied routing problems.

4.1 Proposed BNB-based Method

In the proposed BNB-based method, we modify the BNB approach to address herein studied routing problems, namely the CETSP, the TSPNS, and the GTSPN. We propose the general branching rule and the estimation technique to speed up the algorithm since, in general, the routing problems can be complex and consists of large instances. We utilize the Second-Order Cone Programming (SOCP) [19], Non-Linear Programming (NLP), and Mixed Integer Quadratically Constrained Programming (MIQCP) in the solution of the partial problems of the particular studied generalized routing problems. An overview of the proposed method is depicted in Algorithm 1.

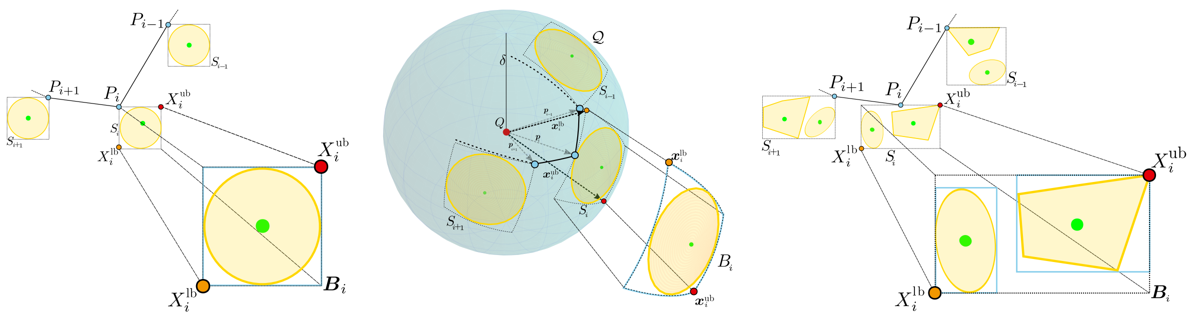
Algorithm 1: Proposed BNB-based solver for generalized routing problems

```
Input: $\mathcal{S} = \{S_1, \dots, S_n\}$ – set of neighborhood sets
Input: $t_{\max}$ – timeout
Output: $\mathcal{LB}$ – lower bound value on the solution ($\Sigma$, $\mathcal{P}$)
  $\mathcal{UB}$ – upper bound value on the solution ($\Sigma$, $\mathcal{P}$)

Function compute_bounds($\Sigma'$):
    $\mathcal{LB}' \leftarrow$ compute_lower_bound($\Sigma'$)
    $\mathcal{UB}' \leftarrow$ compute_upper_bound($\Sigma'$)
    return $\mathcal{LB}', \mathcal{UB}'$

▶ Initialization
  $\mu_{\text{root}}.\Sigma \leftarrow$ select_root($\mathcal{S}$)    // select three regions forming path of the maximal length
  $\mu_{\text{root}}.\mathcal{LB}, \mu_{\text{root}}.\mathcal{UB} \leftarrow$ compute_bounds($\mu_{\text{root}}.\Sigma$)
  $\mu_{\text{root}}$.exact $\leftarrow$ true
  $\mathcal{LB}, \mathcal{UB} \leftarrow \mu_{\text{root}}.\mathcal{LB}, \mu_{\text{root}}.\mathcal{UB}$
  $\mathcal{O} \leftarrow [\mu_{\text{root}}]$

▶ Solution Loop
while $t_{\max}$ is not reached do
    $\mu \leftarrow$ dequeue($\mathcal{O}$)
    if not $\mu$.exact then
        $\mu.\mathcal{LB}, \mu.\mathcal{UB} \leftarrow$ compute_bounds($\mu.\Sigma$)    // compute exact solution of $\mu$
        $\mathcal{UB} \leftarrow$ minimum($\mathcal{UB}, \mu.\mathcal{UB}$)
        $\mu$.exact $\leftarrow$ true
        $\mathcal{O} \leftarrow$ push($\mathcal{O}, \mu$)
        Continue
    $\mathcal{LB} \leftarrow$ maximum($\mathcal{LB}, \mu.\mathcal{LB}$)    // update lower bound of input problem
    $B \leftarrow$ compute_not_covered($\mu$)
    $S^* \leftarrow \arg\max_{S_j \in B} \|S_j - (\mu.\Sigma, \mu.\mathcal{P})\|$    // branching rule
    if is_empty($B$) then
        return $\mathcal{LB}, \mathcal{UB}$
    for $k$ in $1 : |\mu.\Sigma|$ do
        $\mu_{\text{child}}.\Sigma \leftarrow (\sigma'_1, \dots, \sigma'_k, S^*_{idx}, \sigma'_{k+1}, \dots, \sigma'_{|(\mu.\Sigma)|})$
        $\mu_{\text{child}}.\mathcal{LB} \leftarrow$ estimate_lower_bounds($\mu_{\text{child}}.\Sigma$)    // estimate solution of $\mu$
        $\mu_{\text{child}}$.exact $\leftarrow$ false
        $\mathcal{O} \leftarrow$ push($\mathcal{O}, \mu_{\text{child}}$)    // add $\mu_{\text{child}}$ with estimated solution to $\mathcal{O}$
    $\mathcal{O} \leftarrow$ filter($\mathcal{O}, \forall \mu' : \mu'.\mathcal{LB} \leq \mathcal{UB}$)  // filter open list based on lower bounds of nodes in list
return $\mathcal{LB}, \mathcal{UB}$
```



(a) Detail of B_i for the CETSP. **(b)** Detail of B_i for the TSPNS. **(c)** Detail of B_i for the GTSPN.

Figure 4.7: The bounding box B_i of the target region S_i is determined as the most lower-left point X_i^{lb} and the most upper-right point X_i^{ub}. Special case is for the GTSPN (Fig. 4.7(c)), where the points are selected from all bounding boxes of particular regions $\mathcal{R}_{i,k}$, $1 \leq k \leq m_i$, as $X_i^{\mathrm{lb}} = \min_{k=1,\ldots,m_i} X_{i,k}^{\mathrm{lb}}$ and $X_i^{\mathrm{ub}} = \min_{k=1,\ldots,m_i} X_{i,k}^{\mathrm{ub}}$.

In the proposed method, we search the solution space $\mathcal{M}$ represented as a tree, where each node μ consists of the partial problem with a limited set of regions $\mathcal{S}' \subset \mathcal{S}$ represented by the sequence of regions, the solution of the partial problem $(\mu.\Sigma, \mu.\mathcal{P})$ obtained by solving the programming model (SOCP, NLP, or MIQCP), and the lower bound $\mu.\mathcal{LB}$ and upper bound $\mu.\mathcal{UB}$ values, whereas $\mu.\mathcal{LB}$ is the exact solution value of the partial problem $(\mu.\Sigma, \mu.\mathcal{P})$, and $\mu.\mathcal{UB}$ is the solution value of a feasible solution of the original problem constructed from the partial solution. The solution space $\mathcal{M}$ is initialized by creating a root node μ_{root}. Then $\mathcal{M}$ is searched by the Best-First Search (BeFS) utilizing the open list $\mathcal{O}$, where the nodes are ordered by the $\mathcal{LB}$ value associated with the particular node.

The root node μ_{root} is created by selecting three target regions such that the length of the solution of the partial problem formed by these regions is maximal over all possible combinations. Having the partial problem and its solution, we can create the root node and insert it into the open list $\mathcal{O}$.

Once the solution space $\mathcal{M}$ is initialized, the method iteratively dequeues the open list $\mathcal{O}$. For each dequeued node μ, the method determines a set of regions B such that the regions of B are not covered by the partial solution $(\mu.\Sigma, \mu.\mathcal{P})$ (Line 19 of Algorithm 1). From the set B, a region S^* is selected to be used for the *branching* of the node μ based on the *branching rule*. The *branching rule* is a heuristic that selects the most remote region S^* from the solution, i.e., $S^* = \arg\max_{S_j \in B} \|S_j - (\mu.\Sigma, \mu.\mathcal{P})\|$. In the *branching*, the method creates child nodes of μ by inserting S^* (its index label S^*_{idx}) at each position of the sequence $\mu.\Sigma$, thus $|\mu.\Sigma|$ new child nodes μ_{child} are created. For each new sequence $\mu_{\mathrm{child}}.\Sigma$, we determine the lower bound value $\mu_{\mathrm{child}}.\mathcal{LB}$, i.e., given $\mu_{\mathrm{child}}.\Sigma$ of the partial problem, we determine $\mu_{\mathrm{child}}.\mathcal{P}$, such that the length of the path formed by $(\mu_{\mathrm{child}}.\Sigma, \mu_{\mathrm{child}}.\mathcal{P})$ is minimal, by solving the programming model. The length of the obtained path is the desired lower bound value on the solution of the original problem. Then, the child node μ_{child} is added to the open list $\mathcal{O}$. The method iteratively dequeues $\mathcal{O}$ until either the optimal solution is found or the predefined timeout $t_{\max}$ is reached.

Solving the partial problems of nodes can be potentially computationally demanding; thus, we first estimate the lower bound value using the bounding boxes of the target regions (Line 27 of Algorithm 1). The exact solution value of the partial problem is determined right after the node is dequeued from $\mathcal{O}$ (Line 13 of Algorithm 1). The bounding boxes utilized in the estimation are constructed from the most lower-left point X_i^{lb} and the most upper-right point X_i^{ub} of the target regions S_i as depicted in Fig. 4.7. Having X_i^{lb} and X_i^{ub}, we can formulate the partial problem for estimation as the model of the SOCP Model 4.1.1.

Model 4.1.1 (Second-Order Cone Programming (SOCP) for Solution Estimation)

$$\underset{\boldsymbol{x}}{\text{minimize}} \sum_{i=1}^{\bar{n}} f_i \tag{21}$$

s.t.

$$f_i^2 \geq \boldsymbol{w}_i^T \boldsymbol{w}_i \qquad \forall i \in \{1, \dots, \bar{n}\} \tag{22}$$

$$\boldsymbol{w}_i = \boldsymbol{x}_{i+1} - \boldsymbol{x}_i \qquad \forall i \in \{1, \dots, \bar{n}-1\} \tag{23}$$

$$\boldsymbol{x}_i^{\text{lb}} \leq \boldsymbol{x}_i \leq \boldsymbol{x}_i^{\text{ub}} \qquad \forall i \in \{1, \dots, \bar{n}\} \tag{24}$$

Model 4.1.1 is to determine a set of $\bar{n} \leq n$ waypoint variables $\boldsymbol{x}_i$ such that, the path formed by $\boldsymbol{x}_i$ is of the minimal length. The objective function (21) is thus minimized, while the constraints (22) to (24) are satisfied. The variables $f \in \mathbb{R}^{\bar{n}}, f_i \in \mathbb{R}$, and $\boldsymbol{x} \in \mathbb{R}^{\bar{n},d}$ are continuous; the variable f represents the length between two successive waypoint variables, and the variable $\boldsymbol{x}$ denotes the set of the waypoint locations. The variable $\boldsymbol{w} \in \mathbb{R}^{\bar{n},d}$ is auxiliary and denotes the difference in coordinates of the two following waypoint variables (23). The partial solution is then estimated as the set of waypoints, where each waypoint variable $\boldsymbol{x}_i$ is within the bounding box B_i represented by the variables $\boldsymbol{x}_i^{\text{lb}}$ and $\boldsymbol{x}_i^{\text{ub}}$ corresponding to points X_i^{lb} and X_i^{ub} of the particular S_i (24). The value of $\boldsymbol{x}_i$ represents the waypoint location P_i.

The exact solution value of the partial problem of a node is determined after the node is dequeued (Line 13 of Algorithm 1). The exact solution value of the partial problem is considered as a lower bound value of the original problem (Line 18 of Algorithm 1), and it is computed by solving the SOCP, NLP, and MIQCP model formulated to represent the partial problem, for the solution of the CETSP, TSPNS, and GTSPN, respectively. Together with the lower bound value, the upper bound value is computed as a value of a feasible solution created by iteratively adding regions not covered by the exact solution of the node to the solution at the most suitable position of $(\mu.\Sigma, \mu.\mathcal{P})$ (Line 3 of Algorithm 1). The description of the particular models is presented in Sections 4.1.1 to 4.1.3.

4.1.1 Second-Order Cone Programming Model

The Second-Order Cone Programming (SOCP) model utilized for the CETSP is based on the SOCP model presented in [19]. The SOCP is a convex optimization model with linear objective function and affine constraints. It is thus suitable for the formulation of the CETSP as Problem 3.2.1. The utilized model is formulated in Model 4.1.2.

Model 4.1.2 (Second-Order Cone Program (SOCP))

$$\underset{\boldsymbol{x}}{\text{minimize}} \sum_{i=1}^{\bar{n}} f_i \tag{25}$$

s.t.

$$f_i^2 \geq \boldsymbol{w}_i^T \boldsymbol{w}_i \qquad \forall i \in \{1, \dots, \bar{n}\} \tag{26}$$

$$\boldsymbol{w}_i = \boldsymbol{x}_{i+1} - \boldsymbol{x}_i \qquad \forall i \in \{1, \dots, \bar{n}-1\} \tag{27}$$

$$\boldsymbol{v}_i \cdot \boldsymbol{v}_i \leq \delta_i^2 \qquad \forall i \in \{1, \dots, \bar{n}\} \tag{28}$$

$$\boldsymbol{v}_i = \boldsymbol{c}_i - \boldsymbol{x}_i \qquad \forall i \in \{1, \dots, \bar{n}\} \tag{29}$$

Model 4.1.2 is to determine a set of $\bar{n} \leq n$ waypoint variables $\boldsymbol{x}_i$ that form the shortest path visiting $\bar{n}$ disk-shape regions $\mathcal{S}'$. Hence, the model aims to minimize the objective function (25), while the constraints (26) to (29) are satisfied. The model consists of the continuous variable $f \in \mathbb{R}^{\bar{n}}$, $f_i \in \mathbb{R}$, that represents the length of the path between two successive waypoint variables, and the continuous variable $\boldsymbol{x} \in \mathbb{R}^{\bar{n},d}$ representing the waypoint locations. Further, the model contains four constraints that need to be satisfied to obtain a feasible solution for the model. The constraints (26) and (27) denote the length of the path between two consecutive waypoint locations as the difference in coordinates using the auxilary variable $\boldsymbol{w} \in \mathbb{R}^{\bar{n},d}$; the constraints (28) and (29) with the auxilary variable $\boldsymbol{v} \in \mathbb{R}^{\bar{n},d}$ ensure the value of the waypoint variable $\boldsymbol{x}_i$ is within the δ_i distance from the center C_i of the corresponding region S_i, where C_i is represented as variable $\boldsymbol{c}_i$.

4.1.2 Non-Linear Programming Model

The Non-Linear Programming (NLP) model is utilized in solution the TSPNS, where the disk-shaped regions are on the sphere $\mathcal{Q}$. The NLP is an optimization model with non-linear objective function or constraints. We utilize Model 4.1.3 in the solution of the TSPNS as Problem 3.2.2.

Model 4.1.3 (Non-Linear Program (NLP))

$$\underset{\boldsymbol{x}}{\text{minimize}} \sum_{i=1}^{\bar{n}} f_i \tag{30}$$

$$\text{s.t.}$$

$$f_i \geq \arccos(w_i) \qquad \forall i \in \{1,\dots,\bar{n}\} \tag{31}$$

$$-1 \leq w_i \leq 1 \qquad \forall i \in \{1,\dots,\bar{n}\} \tag{32}$$

$$w_i = \boldsymbol{x}_i \cdot \boldsymbol{x}_{i+1} \qquad \forall i \in \{1,\dots,\bar{n}-1\} \tag{33}$$

$$v_i = \boldsymbol{c}_i \cdot \boldsymbol{x}_i \qquad \forall i \in \{1,\dots,\bar{n}\} \tag{34}$$

$$\arccos(v_i) \leq \Omega_i \qquad \forall i \in \{1,\dots,\bar{n}\} \tag{35}$$

$$-1 \leq v_i \leq 1 \qquad \forall i \in \{1,\dots,\bar{n}\} \tag{36}$$

$$\|\boldsymbol{x}_i\| = 1 \qquad \forall i \in \{1,\dots,\bar{n}\} \tag{37}$$

Model 4.1.3 is to determine a set of $\bar{n} \leq n$ waypoint variables $\boldsymbol{x}_i$, such that the path connecting these points is the shortest. Thus the objective function (30) is minimized while the constraints (31) to (37) are met. The model consists of the continuous variables $f \in \mathbb{R}^{\bar{n}}$, $f_i \in \mathbb{R}$, that denotes the length of the arc between two consecutive variables, the auxiliary variables $w \in \mathbb{R}^{\bar{n}}$, $w_i \in \mathbb{R}$, and $v \in \mathbb{R}^{\bar{n}}$, $v_i \in \mathbb{R}$, and the continuous variable $\boldsymbol{x} \in \mathbb{R}^{\bar{x},d}$ corresponding to the waypoint locations $\mathcal{P}$. The model also contains seven constraints; the constraints (31) to (33) represent the minimization of the length of the arc between two consecutive waypoint variables. The constraints (34) to (36) express that the waypoint vector is within Ω_i angle from the center $\boldsymbol{c}_i$ of the region S_i. Finally, (37) expresses the constraints on the variable $\boldsymbol{x}$.

4.1.3 Mixed Integer Quadratically Constrained Programming Model

The Mixed Integer Quadratically Constrained Program (MIQCP) model is utilized for the formulation of the GTSPN. The MIQCP is an optimization model with a linear objective function and at least one quadratic constraint. The utilized Model 4.1.4 follows the formulation of Problem 3.2.3.

Model 4.1.4 (Mixed Integer Quadratically Constrained Program (MIQCP))

$$\underset{\boldsymbol{x}}{\text{minimize}} \sum_{i=1}^{\bar{n}} f_i \tag{38}$$

s.t.

$$f_i^2 \geq \boldsymbol{w}_i^T \boldsymbol{w}_i \qquad \forall i \in \{1, \ldots, \bar{n}\} \tag{39}$$

$$\boldsymbol{w}_i = \boldsymbol{x}_{i+1} - \boldsymbol{x}_i \qquad \forall i \in \{1, \ldots, \bar{n}-1\} \tag{40}$$

$$(\boldsymbol{x}_i - \boldsymbol{c}_{i,k})^T \, \mathbf{P}_{i,k} \, (\boldsymbol{x}_i - \boldsymbol{c}_{i,k}) \leq 1 + M_{i,k} \, \boldsymbol{B}_{i,k} \qquad \forall i \in \{1, \ldots, \bar{n}\}, \forall k \in \{1, \ldots, m_i\} \tag{41}$$

$$\mathbf{A}_{i,k} \, \boldsymbol{x}_i - \mathbf{b}_{i,k} \leq M_{i,k} \, \boldsymbol{B}_{i,k} \qquad \forall i \in \{1, \ldots, \bar{n}\}, \forall k \in \{1, \ldots, m_i\} \tag{42}$$

$$\sum_{k=1}^{m_i} \boldsymbol{B}_{i,k} = 1 \qquad \forall i \in \{1, \ldots, \bar{n}\} \tag{43}$$

$$\boldsymbol{B}_{i,k} \in \{0, 1\} \qquad i \in \{1, \ldots, \bar{n}\}, \forall k \in \{1, \ldots, m_i\} \tag{44}$$

The MIQCP is to determine a set of $\bar{n} \leq n$ waypoint variables $\boldsymbol{x}_i$, such that the path formed by the value of waypoint variables has the minimal length. Thus, the model is to minimize function (38), while the constraints (39) to (44) are satisfied. Model 4.1.4 consists of two continuous variables $f \in \mathbb{R}^{\bar{n}}$, $f_i \in \mathbb{R}$, and $\boldsymbol{x} \in \mathbb{R}^{\bar{n},d}$ corresponding to the waypoint location $\mathcal{P}$, the binary variable $\mathbf{B} \in \mathbb{R}^{\bar{n},m}$, the auxilary variable $\boldsymbol{w} \in \mathbb{R}^{\bar{n},d}$, and several constraints. The constraint (39) denotes the length of the path between two successive neighborhood sets by the difference in coordinates of waypoint locations is minimized using variable $\boldsymbol{w}$ (40). We need to determine which region $\mathcal{R}_{i,k} \in S_i$ of m_i regions is visited, and therefore, we model it as a selection of one of the constraints that represent the particular region, i.e., either (41) or (42) is selected. We introduce the binary variable $\boldsymbol{B}$ together with the *large enough* variable $M \in \mathbb{R}^{\bar{n},m}$ to ensure only one of these constraints (41) and (42) that each corresponds to the particular region is in effect. Then, the variables $\boldsymbol{B}$ and M allow for turning off and on constraints so that only one of m_i constraints is “activated” per each set S_i.

The variable M needs to be *large enough* (the so called *big* M) so that all infeasible solutions of Model 4.1.4 are discarded. Hence, $M_{i,k}$ is determined for each region $\mathcal{R}_{i,k}$ of each set $S_i \in \mathcal{S}$ as the maximal distance between the region $\mathcal{R}_{i,k}$ and its bounding box $B_{i,k}$, i.e., $M_{i,k} = \max_{\mathcal{R}_{i,k} \in S_i} \|\mathcal{R}_{i,k} - B_{i,k}\|$, where $\|\mathcal{R}_{i,k} - B_{i,k}\|$ is the distance between the region and its bounding box. The bounding box B is determined using the MIQCP model with (19) and (20) as the particular constraints and minimization (or maximization) of the particular bounding box coordinate x_j, $j \in \{1, \ldots, d\}$, as the objective function. The bounding box B is determined from the lower-left point X^{lb} and the upper-right point X^{ub}.

Chapter 5

GSOA-based Heuristics to Generalized Routing Problems

The herein studied heuristics solutions of the generalized routing problems are based on the unsupervised learning heuristic approach of the Growing Self-Organizing Array (GSOA) [22]. The GSOA originating from the Self-Organizing Map [30] is adapted to address various routing problems such as the three-dimensional Close Enough Traveling Salesman Problem (CETSP) [28], the Traveling Salesman Problem with Neighborhoods on a Sphere (TSPNS) [10], and the Generalized Traveling Salesman Problem with Neighborhoods (GTSPN) [23]. To make the text of this thesis self-contained, we first describe the GSOA approach and then present particular modifications addressing the studied problems.

Algorithm 2: GSOA-based solver for generalized routing problems

```
Input: $\mathcal{S} = \{S_1, \dots, S_n\}$ – set of target regions
Parameters : $c_{max}$ – the number of learning epochs, $c_{max} = 150$
             $\sigma$ – learning gate, $\sigma = 10$
             $\alpha$ – gain decreasing rate, $\alpha = 0.0005$
             $\mu$ – learning rate, $\mu = 0.6$
Output: $(\Sigma, \mathcal{P})$ – solution represented as sequence of visits $\Sigma$ and correspoding waypoints $P$
► Initialization
1  $c \leftarrow 0$                        // set the learning epoch counter
2  $\mathcal{N} \leftarrow \{\nu_1\}$      // $\nu_1.N$ is randomly selected location within bounding box of $\mathcal{S}$
3  $c_{max} \leftarrow \min(c_{max}, 1/\alpha)$      // avoid negative $\sigma$
► Learning Loop
4  while $c \le c_{max}$ and termination cond. is not satisfied do
5      foreach $S_i$ in a random permutation of $\mathcal{S}$ do
6          $(\nu^*.N, \nu^*.P, \nu_j, \nu_{j+1}) \leftarrow$ select_winner_node$(\mathcal{N}, S_i)$
7          $\mathcal{N} \leftarrow$ insert_node$(\mathcal{N}, \nu^*.N, \nu_j, \nu_{j+1})$
8          $\mathcal{N} \leftarrow$ adapt$(\mathcal{N}, \mu, \sigma, \nu^*, \nu^*.P)$
9      $c \leftarrow c + 1$                // update the epoch counter
10     $\mathcal{N} \leftarrow$ Remove all nodes from the previous epoch $c - 1$
11     $\sigma \leftarrow (1 - c\,\alpha)\,\sigma$       // decrease the learning gain
12     $(\Sigma', \mathcal{P}') \leftarrow$ Traverse $\mathcal{N}$, determine $\Sigma'$, and $\mathcal{P}'$ using waypoints associated with nodes
13     if $c = 1$ or $\mathcal{L}(\Sigma', \mathcal{P}') \le \mathcal{L}(\Sigma, \mathcal{P})$ then
14         $(\Sigma, \mathcal{P}) \leftarrow (\Sigma', \mathcal{P}')$    // update the best solution found so far
15 return $(\Sigma, \mathcal{P})$
```

The GSOA is a growing array of nodes $\mathcal{N} = \{\nu_1, \dots, \nu_M\}$ that is iteratively adapted towards the target regions $\mathcal{S}$ in a finite number of learning epochs. During each epoch, the number of nodes M is varying since new nodes are added to the array, and the nodes added in the previous epoch are removed. Each node $\nu_j \in M$ is associated with its location $\nu_j.N \in \mathbb{R}^d$ in the input space, the region $\nu_j.S_i \in \mathcal{S}$ it covers, and the waypoint location $\nu_j.P \in \nu_j.S_i$ defining the point of visit to the region. Thus, the array $\mathcal{N}$ defines a sequence

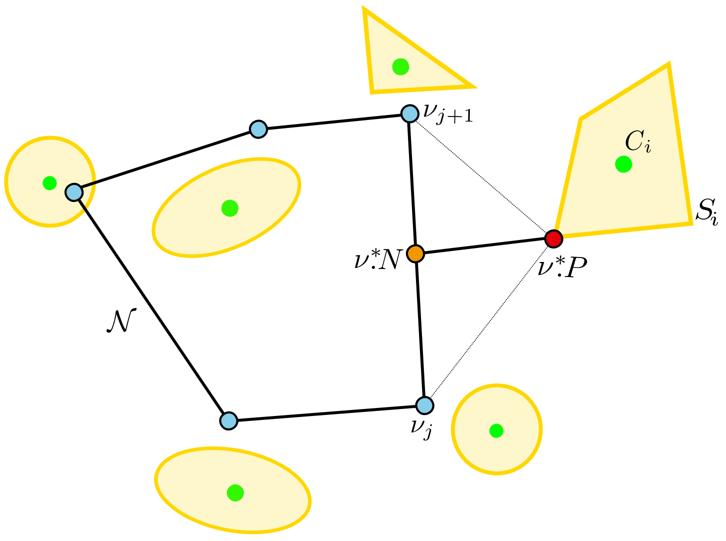


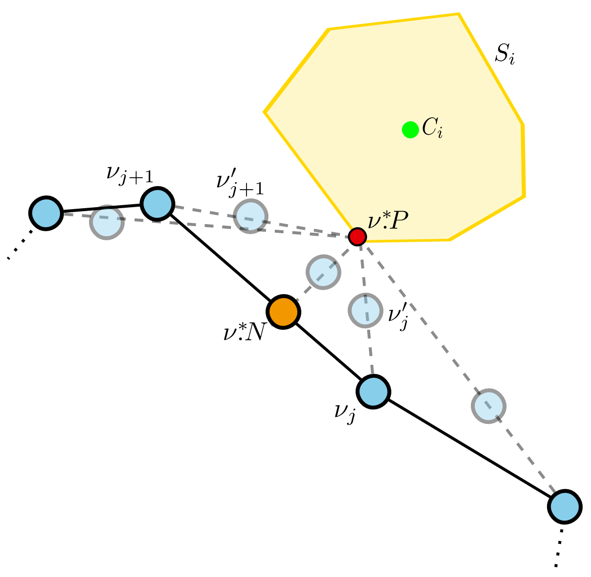
Figure 5.8: Determination of the new winner node ν^* using shortest distance of the path (formed from the array $\mathcal{N}$) to the region S_i. The new winner node ν^* is determined together with the node location $\nu^*.N$ and the waypoint location $\nu^*.P$ using the shortest distance between $\mathcal{N}$, which forms a sequence of straight line segments (ν_j, ν_{j+1}), and the particular region S_i. The shortest distance can be quickly determined in 2D, but it needs to be modified for multidimensional cases studied in this thesis.

of visits Σ to the target regions $\mathcal{S}$, and by traversing the array $\mathcal{N}$, the waypoint locations associated with the nodes form the closed path of the waypoints $\mathcal{P}$. The array $\mathcal{N}$ converges to a stable state in which locations of nodes correspond to locations of their waypoints [22] in the finite number of learning epochs. The GSOA for the generalized routing problems is summarized in Algorithm 2, and the overview of the method follows.

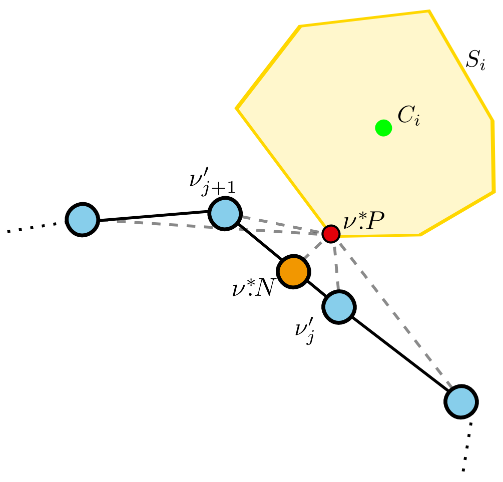
The GSOA method starts by initialization of the array $\mathcal{N}$. The array $\mathcal{N}$ is initialized with one node ν_1, where the location of the node is randomly set within the bounding box of all target regions $\mathcal{S}$. The learning parameters: the gain decreasing rate α, the learning rate μ, and the learning gain σ are set to initial values based on [22, 30], and the maximal number of epochs $c_{\max}$ is set according to the gain decreasing rate α to avoid negative learning gain σ (Line 3 of Algorithm 2).

Then, the method iteratively adapts the array $\mathcal{N}$ to the target regions $\mathcal{S}$, until the number of epochs $c_{\max}$ is reached or the solution is stable. In each epoch, the method iterates over target regions (randomly to avoid local extremes) and for each target region $S_i \in \mathcal{S}$ determines a new winner node ν^* in the function `select_winner_node` (Line 6 of Algorithm 2). The winner node ν^* is determined together with the corresponding waypoint location $\nu^*.P$ as a new node of the path represented by the array $\mathcal{N}$ with the closest distance to the target regions S_i. The node of the array $\nu^*.N$ (whereas the array is defined as the path of consecutively connected node locations) and the waypoint location $\nu^*.P$ are determined during the examination of the closest distance between the path and region. The waypoint is either on the boundary of the target region S_i or is identical to $\nu^*.N$ if the path intersects the region S_i. The determination of $\nu^*.N$ and $\nu^*.P$ depends on the particular type of the region; the description of the proposed winner node selections is described in Sections 5.1 to 5.3 for regions defined as disks (CETSP), disk-shaped regions on a sphere (TSPNS), and the regions defined as neighborhood sets (GTSPN), respectively. The new winner node ν^* is inserted into the array $\mathcal{N}$ at position $\nu^*.N$ between nodes ν_j. and ν_{j+1}, since the array is considered as a sequence of consecutively connected nodes (ν_j, ν_{j+1}), $1 \leq j \leq M$ and $\nu_{M+1} = \nu_1$, see Fig. 5.8.

After the winner node is determined, the array $\mathcal{N}$ with M nodes is adapted towards



(a) Locations of winner node $\nu^*.N$ and its neighboring nodes before adaptation towards the waypoint location $\nu^*.P$.



(b) Locations of winner node $\nu^*.N$ and its neighboring nodes adapted towards the waypoint location $\nu^*.P$.

Figure 5.9: Visualization of the GSOA adaptation. The location of the winner node $\nu^*.N$ and its neighboring nodes are adapted towards the determined waypoint location $\nu^*.P$.

the regions S_i by adapting the new winner node ν^* and its neighboring nodes towards $\nu^*.P$. The adapted neighboring nodes are within $d' < 0.2M$ distance from ν^* in the array, and their locations are updated according to

$$\nu.N \leftarrow \nu.N + \mu f(\sigma, d')(\nu^*.P - \nu.N), \tag{45}$$

where the neighboring function is defined as

$$f(\sigma, d') = \exp\frac{-d'^2}{\sigma^2}. \tag{46}$$

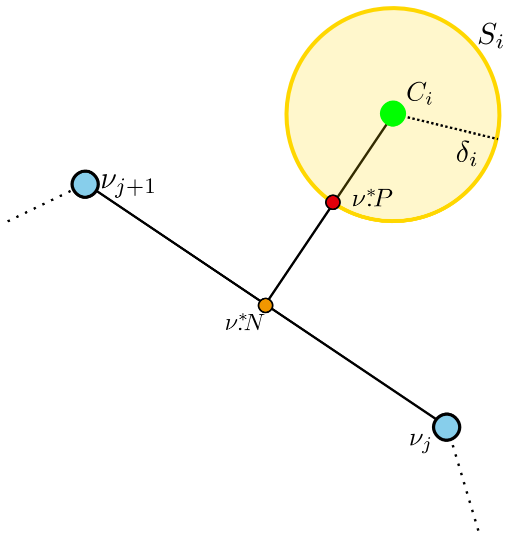
The adaptation is visualized in Fig. 5.9.

After the adaptation to all target regions, all nodes from the previous epoch are removed, and only nodes added in the current epoch are preserved. Hence the array $\mathcal{N}$ contains $M = n$ nodes, each associated with a target region and waypoint location, and thus a solution can be extracted from $\mathcal{N}$ after each learning epoch. The selection of the winner nodes and adaptation are repeated until $c_{\max}$ is reached or the solution is stable, i.e, nodes are negligibly close to their waypoint locations, $\|(\nu.N, \nu.P)\| \leq 10^{-4}$ for all $\nu \in \mathcal{N}$ [22]. The final solution is found as the best solution from the performed epochs.

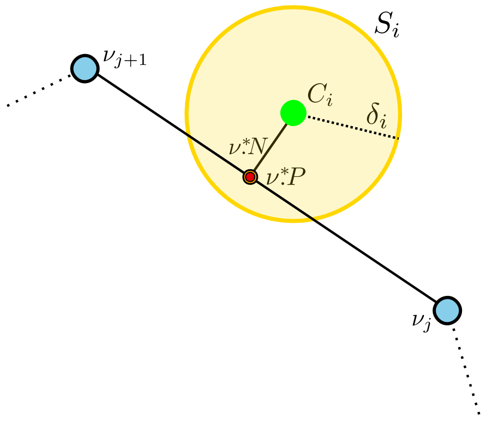
5.1 Winner Selection for the 3D CETSP

The deployment of the GSOA in the solution of the 2D CETSP ($d = 2$) is presented in [22] and the extension to the 3D ($d = 3$) is relatively straightforward. Therefore, we present only an overview of `select_winner_node` method from [28].

The winner node ν^* is determined using the array $\mathcal{N}$ with M nodes defined as path of straight line segments between two consecutive nodes (ν_j, ν_{j+1}), $1 \leq j \leq M$ and $\nu_{M+1} = \nu_1$.



(a) Determination of the waypoint $\nu^*.P$ in case when $\nu^*.N$ is outside region S_i.



(b) Determination of the waypoint $\nu^*.P$ in case $\nu^*.N$ is within region S_i, hence $\nu^*.P = \nu^*.N$.

Figure 5.10: Visualization of the proposed determination of the winner node location $\nu^*.N$ and the associated waypoint $\nu^*.P$ at which the disk-shaped region S_i is visited. The node location $\nu^*.N$ corresponding to the closest point of the line segment $(\nu_j,\ \nu_{j+1})$ to S_i is determined as the intersection of the perpendicular segment consisting of C_i and the determined $\nu^*.N$ with $(\nu_j,\ \nu_{j+1})$. The waypoint location is determined as intersection of the region border and $(\nu^*.P, C_i)$ (Fig. 5.10(a)); however, if $(\nu_j,\ \nu_{j+1})$ intersects S_i, the waypoint locations is the same as the node location $\nu^*.N$ within S_i (Fig. 5.10(b)).

We iterate over the segments and determine the node location $\nu^*.N$ as the closest point of the j-th segment (ν_j, ν_{j+1}) to the disk-shaped region S_i defined by its center C_i and radius δ_i, and we select the segment with the minimal distance to C_i (47).

$$\nu^*.N = \operatorname*{arg\,min}_{N' \in (\nu_j, \nu_{j+1})} \left\| N' - C_j \right\| \tag{47}$$

With the node location $\nu^*.N$, we determine the corresponding location $\nu^*.P$, where we distinguish two cases, see Fig. 5.10. In the first case shown in Fig. 5.10(a), the node location $\nu^*.N$ is outside the δ_i-neighborhood of C_i. The corresponding waypoint location $\nu^*.P$ is determined on the border of the disk-shaped region with distance δ_i (and some small constant ϵ) from C_i as

$$\nu^*.P = C_i + (\nu^*.N - C_i)\frac{\delta_i - \epsilon}{\|\nu^*.N - C_i\|}. \tag{48}$$

In the second case, the node location $\nu^*.N$ is within the δ_i-neighborhood from C_i, and thus, $\nu^*.P = \nu^*.N$, see Fig. 5.10(b).

5.2 Winner Selection for the TSPNS

The deployment of the GSOA in the solution of the TSPNS is presented in [10]. Since the disk-shaped regions are on the sphere, we need a different approach to determine the winner node than the one used in [22, 28]. We consider the array $\mathcal{N}$ as a path of arcs between

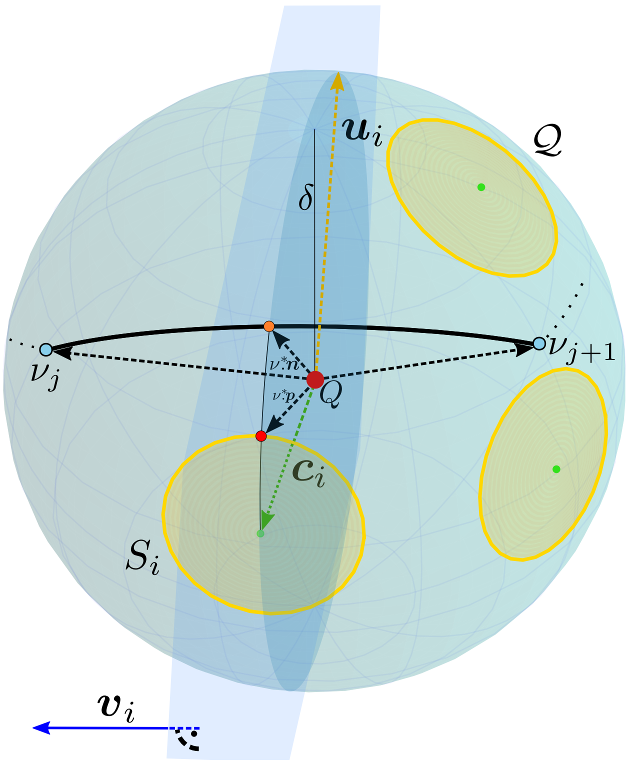


Figure 5.11: Proposed determination of winner node location vector $\nu^*.\boldsymbol{n}$ and the associated waypoint vector $\nu^*.\boldsymbol{p}$ at which the region S_i is visited. The node location $\nu^*.\boldsymbol{n}$ corresponds to the closest vector of the arc $(\nu_j,\ \nu_{j+1})$ to S_i and is determined as the cross product of the vectors $\boldsymbol{v}_i$ and $\boldsymbol{u}_i$. The vector $\nu^*.\boldsymbol{p}$ is derived from the rotation of $\nu^*.\boldsymbol{n}$ to the close neighborhood of $\boldsymbol{c}_i$.

two consecutive nodes (ν_j, ν_{j+1}), $1 \le j \le M$ and $\nu_{M+1} = \nu_1$, the locations represented as vectors, and the measured distance as distance on the sphere. Detailed visualization of the determination of the winner node ν^* for the target region S_i is depicted in Fig. 5.11 and it as follows.

First, we determine the vector $\boldsymbol{u}_i$ that is perpendicular to the plane formed by the arc between two vectors corresponding to the node locations $(\nu_j.\boldsymbol{n}, \nu_{j+1}.\boldsymbol{n})$ as the cross product of two vectors

$$\boldsymbol{u}_i = \nu_j.\boldsymbol{n} \times \nu_{j+1}.\boldsymbol{n}. \tag{49}$$

After that, we determine the vector $\boldsymbol{v}_i$ as perpendicular vector to the plane formed by the vector $\boldsymbol{u}_i$ and $\boldsymbol{c}_i$ (the vector defining the target region S_i). The vector $\boldsymbol{v}_i$ is determined as cross product of two vectors

$$\boldsymbol{v}_i = \boldsymbol{c}_i \times \boldsymbol{u}_i. \tag{50}$$

Then, we rotate the vector $\boldsymbol{v}_i$ to the plane $(\nu_j.\boldsymbol{n}, \nu_{j+1}.\boldsymbol{n})$ represented as $\boldsymbol{u}_i$ (49) to determine the node vector $\nu^*.\boldsymbol{n}$

$$\nu^*.\boldsymbol{n} = \boldsymbol{v}_i \times \boldsymbol{u}_i. \tag{51}$$

Now, to determine the waypoint vector $\nu^*.\boldsymbol{p}$, we need to rotate $\nu^*.\boldsymbol{n}$ so that $\angle(\boldsymbol{c}_i, \nu^*.\boldsymbol{p}) \le \Omega_i$ holds, i.e., the $\nu^*.\boldsymbol{p}$ is determined as

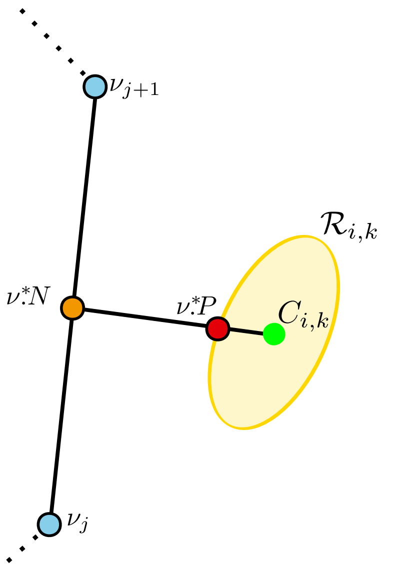
$$\nu^*.\boldsymbol{p} = \frac{\boldsymbol{p}'}{\|\boldsymbol{p}'\|}, \tag{52}$$

where

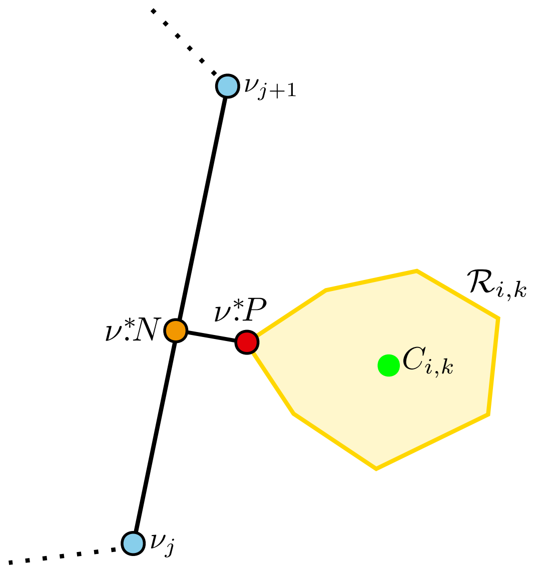
$$\boldsymbol{p}' = \begin{cases} \boldsymbol{c}_i + \tan\Omega_i \, (\nu^*.\boldsymbol{n} \times \boldsymbol{c}_i) \times \boldsymbol{c}_i, & \text{if } \angle(\boldsymbol{c}_i, \nu^*.\boldsymbol{n}) > \Omega_k, \\ \nu^*.\boldsymbol{n} & \text{otherwise.} \end{cases} \tag{53}$$

5.3 Winner Selection for the GTSPN

In the deployment of the GSOA in the solution of the GTSPN presented in [23], we consider non-overlapping neighborhood sets with 3D and 7D convex possibly overlapping regions ($d \in \{3, 7\}$) . Thus, we cannot deploy the winner node selection as presented in [22] and Section 5.1. We consider the array $\mathcal{N}$ as a sequence of straight line segments of two consecutive nodes (ν_j, ν_{j+1}), $1 \leq j \leq M$ and $\nu_{M+1} = \nu_1$. Then, we determine the winner node ν^* for the target region S_i by exploiting the convexity of regions of sets and the centroids of regions depicted in Fig. 5.12 as follows.



(a) Determination of winner node location $\nu^*.N$ and corresponding waypoint location $\nu^*.P$ for ellipsoid-shaped region $\mathcal{R}_{i,k}$.



(b) Determination of winner node location $\nu^*.N$ and corresponding waypoint location $\nu^*.P$ for polyhedron-shaped region $\mathcal{R}_{i,k}$.

Figure 5.12: Visualization of the proposed determination of the winner node location $\nu^*.N$ and the associated waypoint $\nu^*.P$ at which the neighborhood set S_i is visited. The node location $\nu^*.N$ corresponding to the closest point of the line segment (ν_j, ν_{j+1}) to S_i is determined as the intersection of the perpendicular segment consisting of C_i and the determined $\nu^*.N$ to (ν_j, ν_{j+1}). The visualization is shown in 2D, but the principle also holds for 3D and 7D regions [23].

First, we determine the shortest paths from the center of each particular region $\mathcal{R}_{i,k} \in S_i$ to the array $\mathcal{N}$. From these paths, we select the shortest path with the corresponding region

$$\nu^*.N, k = \underset{1 \leq k' \leq m_i, N' \in (\nu_j, \nu_{j+1})}{\arg\min} \left\| N' - C_{i,k'} \right\| . \tag{54}$$

The waypoint locations are then determined based on the type of the region. Having the ellipsoid-shaped region, we denote the left side of equation of the ellipsoid (20) λ, i.e., $\lambda = ((X - C_{i,k})^T \boldsymbol{P}_{i,k} (X - C_{i,k}))$. If $\lambda \leq 1$, then $\nu^*.N$ is inside the ellipsoid, and thus $\nu^*.P =$

$\nu^*.N$. Otherwise, we determine the point $\nu^*.P$ as point on the segment $(\nu^*.N, C_{i,k})$ with the distance $1/\lambda$ from the ellipsoid center $C_{i,k}$. It is ensured that the point is inside the ellipsoid since in this case $\lambda > 1$, and thus, $\frac{1}{\lambda} \leq 1$. For polyhedron regions $\mathcal{R}_{i,k} \in S_i$, we utilize the definition of the polyhedra as a set of half-spaces. Thus, we determine the waypoint as the closest intersection point to the segment (ν_j, ν_{j+1}). The intersection points are intersection of $(\nu^*.N, C_{i,k})$ and half-spaces of the polyhedron. For hybrid regions, both the methods are combined, and the waypoint $\nu^*.P$ is the closest point to $\nu^*.N$ that satisfies (19) and (20).

Chapter 6

Results

In this chapter, we present and discuss empirical evaluation and results of the proposed BNB method and the GSOA-based methods for the generalized routing problems. Since we study several routing problems with various neighborhoods: the Close Enough Traveling Salesman Problem (CETSP), the Traveling Salesman Problem with Neighborhoods on a Sphere (TSPNS), and the Generalized Traveling Salesman Problem with Neighborhoods (GTSPN), the problems are evaluated in particular sections.

The obtained lower bound values are compared with the existing lower bound values; however, in the literature, the lower bounds are reported only for CETSP in 2D and 3D, and thus, we compare the proposed lower bounds only for CETSP with lower bounds obtained by the Branch-and-Bound [19], the simplification method by Mennell [20], and the adaptive algorithm $(lb/ub)Alg$ [27]. We report the first-ever obtained lower bounds values for the TSPNS and GTSPN.

The proposed GSOA-based methods are compared with the existing approaches to each particular problem; the solutions of the CETSP are compared with solutions obtained by the Steiner-Zone-based method SZ2 [20] and the adaptive algorithm $(lb/ub)Alg$ [27]. For the TSPNS, the baseline methods are the TSPN-LKH and TSPN-GLKH [10]. In the case of the GTSPN, we consider the GTSPN solvers HRGKA [12] and Centroid-GTSP$^+$ [23].

The proposed BNB-based method is implemented in `Julia 1.5.3` [49] and further referred to as the BNB. The BNB solver is deterministic and executed only once with the predefined timeout $t_{\max}$. The solver utilizes `JuMP v0.21.6` [50], the domain-specific modeling language for modeling the proposed mathematical models listed in Model 4.1.2, Model 4.1.3, and Model 4.1.4. The models are solved using two mathematical solvers, the `CPLEX v0.6.6` solver [51] for the SOCP and MIQCP models, and the `Ipopt v.0.6.5` solver [52] for the NLP model. The GSOA-based methods are implemented in `C++` and denoted as GSOA. Since the GSOA is randomized, each examined instance is solved for multiple trials. The computational environment consists of the `Intel Xeon Scalable Gold 6146`.

Evaluation of the solvers is based on the reference solution value $\mathcal{L}_{ref}$ for each particular problem instance obtained as the best solution value among all evaluated solvers. The solution value $\mathcal{L}$ obtained by the particular solver is considered as the best solution value among the performed trials. The reported lower and upper bounds are obtained by the BNB solver and denoted as $\mathcal{LB}$ and $\mathcal{UB}$, respectively. The quality of solutions provided by a particular solver is measured as the percentage deviation of $\mathcal{L}$ to $\mathcal{L}_{ref}$ denoted %PDB and computed as

$$\%\text{PDB} = \frac{\mathcal{L} - \mathcal{L}_{ref}}{\mathcal{L}_{ref}} 100\% \,. \tag{55}$$

The solution robustness is measured as the percentage deviation of the mean solution value $\mathcal{L}_{mean}$ to $\mathcal{L}_{ref}$ denoted %PDM and determined as

$$\%\text{PDM} = \frac{\mathcal{L}_{mean} - \mathcal{L}_{ref}}{\mathcal{L}_{ref}} 100\% \,, \tag{56}$$

where the mean solution value $\mathcal{L}_{mean}$ is obtained as an average solution value among the performed trials by the particular solver. Moreover, we report solution values as the relative gap

%GAP for the particular problem instance and solver computed as

$$\%\mathrm{GAP} = \frac{\mathcal{L} - \mathcal{LB}}{\mathcal{L}} 100\%. \tag{57}$$

Besides, we utilize the five-number summary to describe the obtained results with the minimum value, the first quartile, the median value, the third quartile, and the maximum value. A summary of the evaluation results is presented in the following sections.

6.1 Lower Bounds

The presented lower bound values $\mathcal{LB}$ and the upper bound values $\mathcal{UB}$ are obtained by the proposed BNB-based solver that is executed for a single trial since the solver is deterministic and with the maximal computation time $t_{\max}$ set to 16 hours. The lower and upper bounds are established for instances of the CETSP, TSPNS, and GTSPN and are described in detail in the following Sections 6.2 to 6.4. Since no lower bounds are reported for the TSPNS and GTSPN, we only compare the obtained lower bounds for the CETSP with the lower bounds for the CETSP reported in [19, 20, 27]. The results are listed in Table A.1, Table A.2, Table A.3, Table A.4, Table A.5, and Table A.6. The obtained lower bounds for TSPNS and GTSPN are reported in Table A.7.

CETSP The results for the CETSP reported in Table A.1, Table A.2, and Table A.3 indicate that the proposed BNB solver provides lower bounds values competitive to the Branch-and-Bound [19]. In several cases, the proposed BNB provides better lower bound values than $(lb/ub)Alg$, albeit the BNB solver is significantly more computationally demanding. Regarding the varying results between the similar approaches of the Branch-and-Bound and proposed BNB solver, both solvers are based on the BNB method; however, the Branch-and-Bound [19] utilizes several *branching rules* specifically designed for the CETSP and uses feasible solution values obtained by [20] as the first $\mathcal{UB}$ estimates for particular problem instances. In the proposed BNB, we utilize only one general *branching rule*, and the first $\mathcal{UB}$ is obtained as a feasible solution to the first partial solution. Besides, the proposed BNB utilizes the bounding box based estimation of the partial solutions.

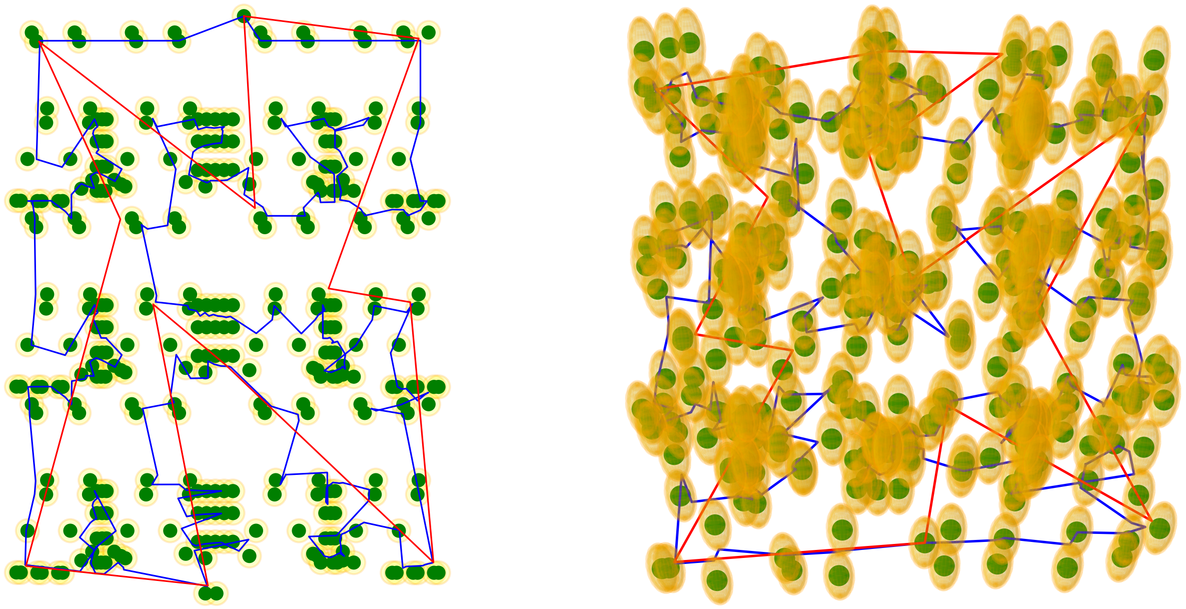
3D CETSP The lower bounds for the CETSP in the 3D are reported only in [19]; hence the comparison in Table A.4, Table A.5, and Table A.6 is between the proposed BNB solver and the Branch-and-Bound [19], whereas the most of the instances for *Varied ratio* are not reported in [19]. Similarly to the established lower bounds for the CETSP in the 2D, the lower bounds estimated by the proposed BNB are competitive to values reported in [19]. For the *Varied ratio* instances, the proposed BNB provides the first lower bounds values for the most of the problem instances.

TSPNS and GTSPN The lower bounds values of the TSPNS and GTSPN instances are depicted in Table A.7. The lower bounds for the TSPNS are tight for 13 problem instances. On the other hand, the lower bound values for the GTSPN are relatively weak since the problem is complex and computationally demanding to solve. However, no other lower bounds are reported in the literature, and therefore, the values obtained by the proposed BNB solver are used in the evaluation of heuristic solutions to these problems.

6.2 Close Enough TSP

Overall, 62 problem instances of the CETSP in 2D and 3D [20] have been examined. The instances can be divided into three groups.

- ***Overlap ratios*** - The group consists of 21 problem instances with specified overlap ratio from $\{0.02, 0.10, 0.30\}$. An example of instances is shown in Fig. 6.13.


(a) Problem `lin318-or0.02`:
$\mathcal{LB} = 1902.99$, $\mathcal{UB} = 3652.78$, %GAP = 47.90

(b) Problem `lin310-or0.02`:
$\mathcal{LB} = 1902.80$, $\mathcal{UB} = 3943.35$, %GAP = 51.72

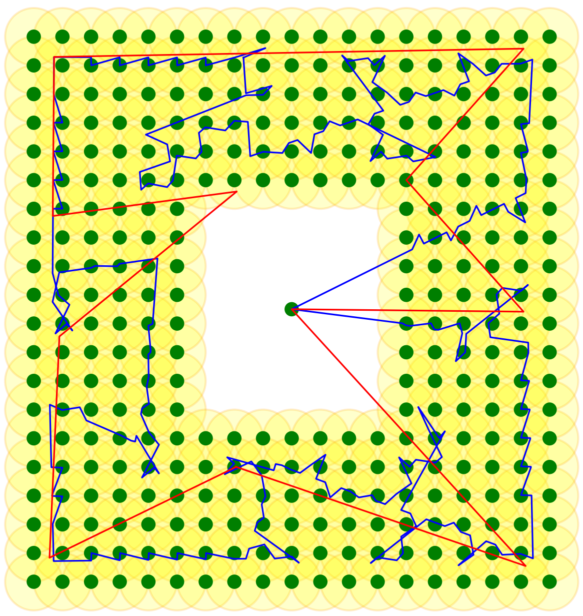
Figure 6.13: Problem instances from the category *Overlap ratios* of the CETSP in the 2D and 3D with solution and lower bound obtained by the BNB. The solution is depicted in blue and the lower bound in red.

- ***Varied overlap ratios*** - The group of 27 problem instances with varied overlap ratio, where all locations of particular instance are associated with the same radius δ_i. An example of instances is depicted in Fig. 6.14.

- ***Arbitrary radius*** - The group of 14 problem instances, where every location of the particular instance is associated with different arbitrary radius δ_i. An example of instances is depicted in Fig. 6.15.

Performance in 2D instances of the CETSP The performance results of the examined solvers in the solution of 2D instances of the CETSP are reported in Table A.8, Table A.9, and Table A.10. In particular, we examined solutions obtained by the Steiner-Zone-based method SZ2 [20], the adaptive algorithm $(lb/ub)Alg$ [27], and the unsupervised learning method of the GSOA [22]. The implementation available for [22] has been utilized to obtained the results for the 2D CETSP instead of using the results reported in [22]. It is because we were unable to replicate therein reported results, and we discovered that few instances were not as originally proposed in [20].[2] The solution optimality is measured by %GAP (57) utilizing $\mathcal{LB}$ and $\mathcal{UB}$ obtained by the proposed BNB solver executed for a single trial, since it is deterministic, with

[2]Here, we would like to acknowledge a discussion with our collegue Miroslav Kulich, who pointed out that there might be some mistakes in few instances reported in [22].


(a) Problem `bubbles6`:
$\mathcal{LB} = 945.30$, $\mathcal{UB} = 1817.29$, %GAP = 47.98

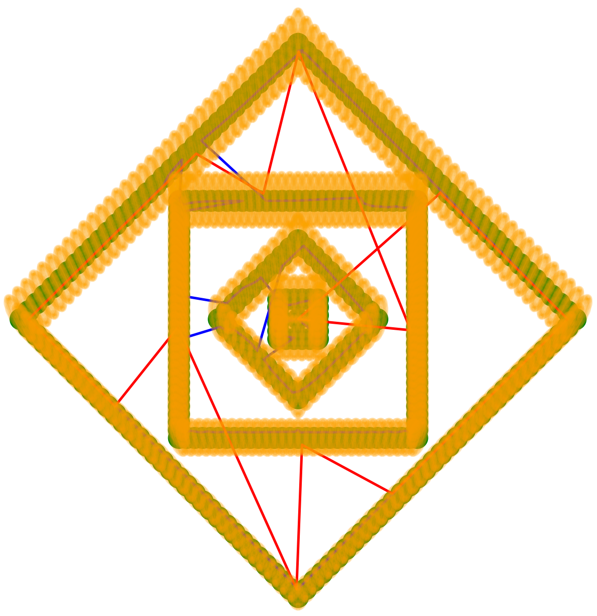
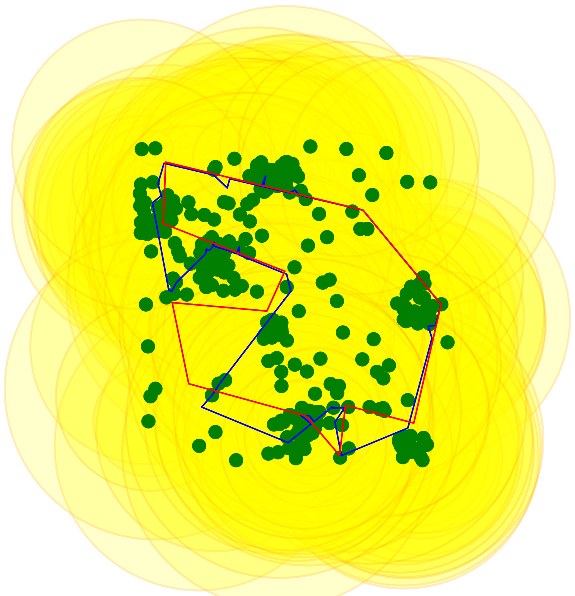

(b) Problem `rotatingDiamonds4`:
$\mathcal{LB} = 568.48$, $\mathcal{UB} = 842.19$, %GAP = 32.50

Figure 6.14: Problem instances from the category *Varied ratios* of the CETSP in the 2D and 3D with solution and lower bound obtained by the BNB. The solution is depicted in blue and the lower bound in red.


(a) Problem `team3_300rdmRad`:
$\mathcal{LB} = 367.11$, $\mathcal{UB} = 404.77$, %GAP = 9.3

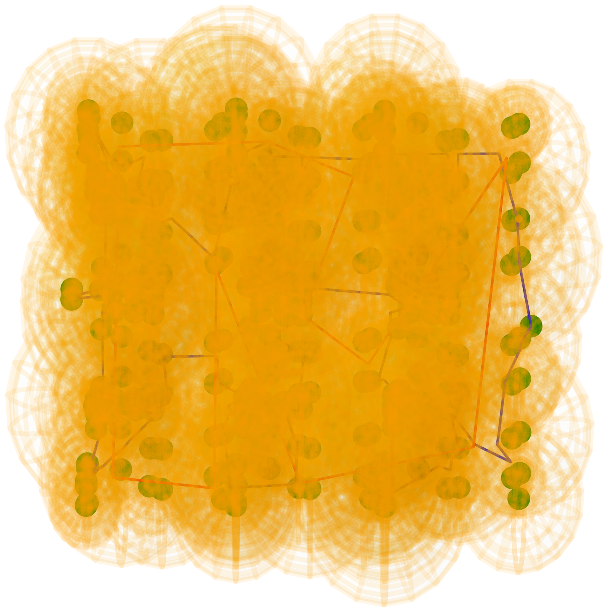
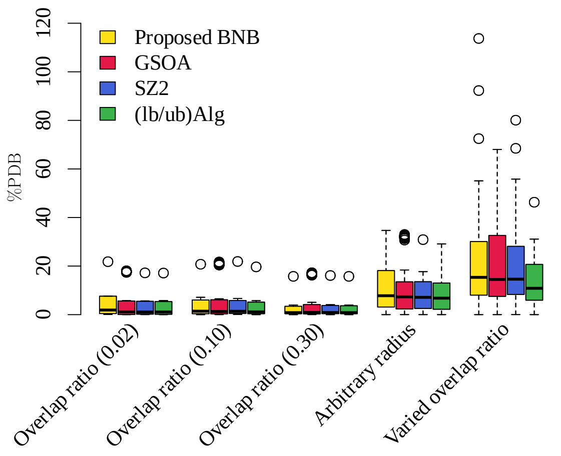
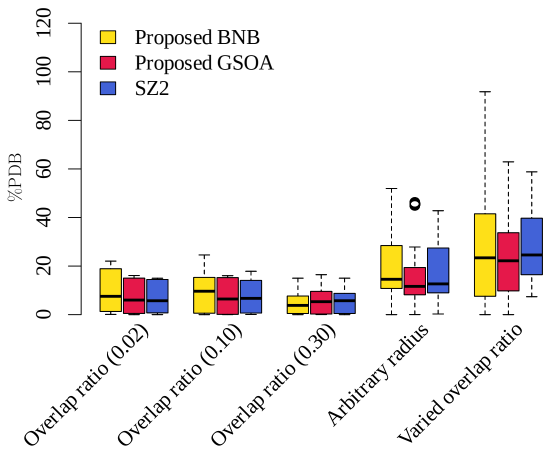

(b) Problem `lin318rdmRad`:
$\mathcal{LB} = 1670.27$, $\mathcal{UB} = 2557.20$, %GAP = 34.68

Figure 6.15: Problem instances from the category *Arbitrary radius* of the CETSP in the 2D and 3D with solution and lower bound obtained by the BNB. The solution is depicted in blue and the lower bound in red.

the timeout $t_{\max}$ is set to 16 hours. Comparing the reported results of the examined methods, the $(lb/ub)Alg$ provides solutions of better quality than the other two solvers. Besides, it provides 7 optimal solutions and 11 close to the optimal solutions. However, $(lb/ub)Alg$ is computationally demanding, and both SZ2 and GSOA have significantly low computational requirements. An overview of the aggregated results as the five-number summary is depicted in Fig. 6.16(a).



(a) 2D CETSP instances



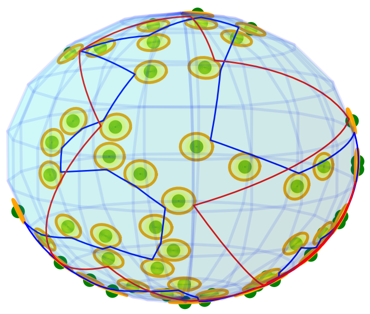
(b) 3D CETSP instances

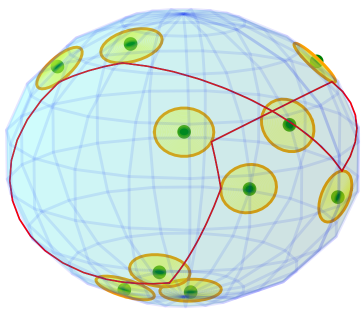
Figure 6.16: Aggregated quality of solutions %PDB depicted as five-number summary obtained by the CETSP solvers in the groups based on the overlap ratio.

Performance in the 3D instance of the CETSP In the case of 3D instances, we present $\mathcal{LB}$ and $\mathcal{UB}$ provided by the proposed BNB-based solver and report results of the existing solver SZ2 [20] and the proposed GSOA solver. The BNB solver is executed for a single trial since it is deterministic, and the proposed GSOA solver is executed for 20 trials. Results depicted in Table A.11, Table A.12, and Table A.13 indicate the proposed GSOA-based solver outperforms the SZ2 method for most of the instances. However, it fails in highly overlapping instances, as it can be seen in Fig. 6.16(b).

6.3 TSP with Neighborhoods on a Sphere

The instances of the TSPNS [10] that vary in the number of regions, the volume of the sphere surface covered by regions, and the distribution of regions over the sphere's surface, can be divided into several categories. Hence each instance is denoted by the type of the distribution, if the regions overlap, and the number of regions. The distribution of regions is either random or regular even distribution. The instances with the random distribution are denoted with `rand`, and the instances with even distribution `reg`, where the distribution is created based on the Fibonacci Lattice method. Moreover, the instances with the even distribution can be distinguished based on the locations of regions. Instances with regions placed in the middle part of the sphere are denoted with `band`, instances with regions located on the top of the sphere are denoted either as `cap` if they are in a spherical pattern, and `rect_cap` if the regions are in a rectangular pattern. If at least one region of the particular instance overlaps with at least one other region, the instance is denoted as `ol`, `nol` otherwise. Finally, each instance is denoted by the number of regions n selected from the range $n \in \{10, 25, 50, 100, 500\}$. The instances are created with a sphere $\mathcal{Q}$ centered in $Q = (0, 0, 0)$ with $\delta = 1$. Examples of selected instances are depicted in Fig. 6.17.


(a) Problem `sphere_cap_rand_nol_50x1`: $\mathcal{LB} = 10.46$, $\mathcal{UB} = 10.88$, %GAP $= 3.85$


(b) Problem `sphere_rand_ol_10x1`: $\mathcal{LB} = 7.56$, $\mathcal{UB} = 7.56$, %GAP $= 0.00$

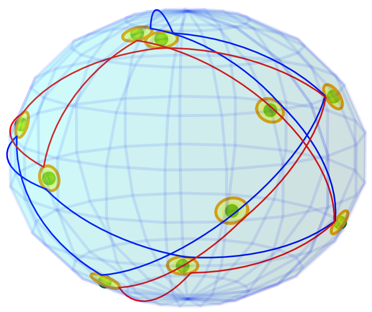
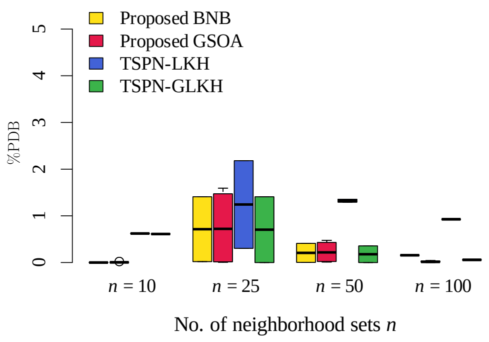
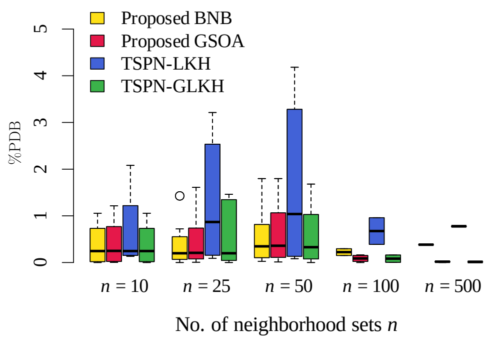

(c) Problem `sphere_reg_nol_10x1`: $\mathcal{LB} = 10.11$, $\mathcal{UB} = 10.11$, %GAP $= 0.00$

Figure 6.17: Problem instance of the TSPNS with solution and lower bound obtained by the BNB. The solution is depicted in blue and the lower bound in red.

The performance indicators of examined methods are reported in Table A.14. In particular, the instances are solved by the TSPN-LKH and TSPN-GLKH, both introduced in [10], and the herein proposed GSOA solver for the TSPNS. The lower bound $\mathcal{LB}$ and upper bound $\mathcal{UB}$ values are obtained by the proposed BNB solver. The BNB solver is executed for a single trial since it is deterministic, and the GSOA solver is executed for 20 trials.



(a) Overlapping TSPNS instances



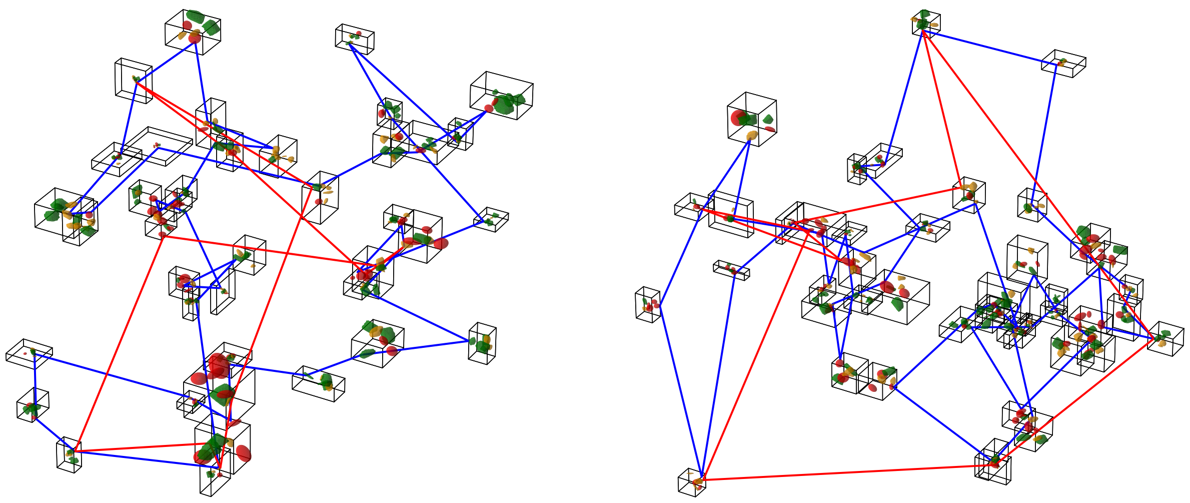
(b) Non-overlapping TSPNS instances

Figure 6.18: Aggregated quality of solutions %PDB depicted as five-number summary obtained by the TSPNS solvers in groups of problems based on the number of regions.

Based on the reported results that are further summarized in Fig. 6.18, the proposed GSOA-based solver computationally outperforms both the TSPN-LKH and TSPN-GLKH methods. In comparison to the TSPN-GLKH, the results reported for the TSPNS-based method with 10 samples indicate that the TSPN-GLKH provides optimal solutions to most of the problem instances. However, both the TSPN-based methods perform relatively poorly for the overlapping instance, see Fig. 6.18(a).

6.4 Generalized TSP with Neighborhoods

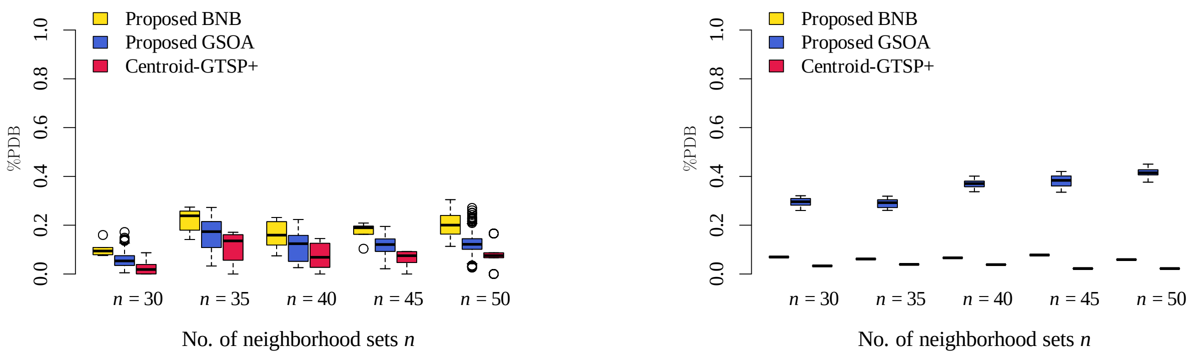
The instances of the GTSPN [53] are randomly generated with varying number of neighborhood sets $n \in \{30, 35, 40, 45, 50\}$, the constant number of the regions per neighborhood set $m_i = 6$, $i \in \{1, \dots, n\}$, and the dimension $d = 3$ and $d = 7$. The regions generated within the particular configuration of the lower and upper bounding box are approximated to simulate complex robotics neighborhoods using linear constraints (to create polyhedra), quadratic constraints (to create ellipsoids), or a combination of both (to create hybrid regions) [43]. An example of the examined GTSPN instances is depicted in Fig. 6.19.



(a) Problem `3D_40_6_c`:
$\mathcal{LB} = 2370.80$, $\mathcal{UB} = 4663.88$, %GAP = 49.16

(b) Problem `3D_45_6_f`:
$\mathcal{LB} = 2608.16$, $\mathcal{UB} = 5193.72$, %GAP = 49.78

Figure 6.19: Problem instance of the GTSPN with solution and lower bound obtained by the BNB. The solution is depicted in blue and the lower bound in red.



(a) 3D GTSPN instances **(b)** 7D GTSPN instances

Figure 6.20: Aggregated quality of solutions %PDB depicted as five-number summary obtained by the GTSPN solvers in groups of problems based on the number of regions.

Results for the evaluated methods the HRGKA [12], Centroid-GTSP$^+$ [23], and the proposed GSOA solver are reported in Table A.15 and Table A.16. The lower bound $\mathcal{LB}$ and upper bound $\mathcal{UB}$ values are obtained by the proposed BNB solver. The BNB solver is executed for a single trial since it is deterministic, and the GSOA solver is executed for 20 trials.

Regarding the reported results for the 3D instances, the proposed GSOA solver provides competitive solutions with the HRGKA; although the Centroid-GTSP$^+$ outperforms both methods, see the aggregated results in Fig. 6.20(a). In the 7D instances of the GTSPN, the HRGKA outperforms both the Centroid-GTSP$^+$ and GSOA, see Fig. 6.20(b).

Chapter 7

Conclusion

In this thesis, we study the generalized routing problems by estimating the lower bounds on the optimal solution values and heuristic solutions. We study three problems motivated by different robotics tasks and formulated with various continuous and convex neighborhoods; namely, we study the Close Enough Traveling Salesman Problem (CETSP), the Traveling Salesman Problem with Neighborhoods on a Sphere (TSPNS), and the Generalized Traveling Salesman Problem with Neighborhoods (GTSPN). We establish the lower bounds on the optimal solution values by the proposed BNB solver for each of these problems. The solver is designed to be easily extended to address more generalized routing problems since it includes a general *branching rule*. Each addressed problem is modeled using programming models such as the Second-Order Cone Programming, Non-Linear Programming, or Mixed Integer Quadratically Constrained Programming. Moreover, we established the solutions of the herein studied problems by the Growing Self-Organizing Array (GSOA) [22] extended to the address these problems.

Regarding the reported results, we utilize the performance indicator %GAP to indicate the optimality of the reported solutions. The optimality indicator %GAP is measured as a relative gap of the solution values to the established lower bound values. The proposed GSOA solver provides competitive solutions in a short computational time compared to the existing approaches. For several instances, it provides optimal solutions to the herein studied problems. The established lower bounds are competitive to the existing lower bounds; however, only lower bounds for the CETSP are reported in the literature. The newly established lower bound values for the TSPNS and GTSPN are the first such results for these problems, and therefore, they can be used for further developed heuristic and benchmarking.

There are two possible streams in the developed BNB-based solver for the generalized routing problems in future work. First, the computational requirements can be decreased by additional branching rules and initialization using heuristics solutions. Besides, the framework might be further generalized to challenging robotic routing problems such as the Dubins Traveling Salesman Problem with Neighborhoods.

References

[1] K. A. Eldrandaly, A. H. N. Ahmed, and A. AbdAllah, “Routing problems: A survey,” in *The 43rd Annual Conference on Statistics, Computer Sciences and Operations Research*, 12 2008, pp. 51–70.

[2] S. Alatartsev, S. Stellmacher, and F. Ortmeier, “Robotic Task Sequencing Problem: A Survey,” *Journal of Intelligent & Robotic Systems*, vol. 80, no. 2, pp. 279–298, 2015.

[3] P. Oberlin, S. Rathinam, and S. Darbha, “Today’s traveling salesman problem,” *IEEE Robotics & Automation Magazine*, vol. 17, no. 4, pp. 70–77, 2010.

[4] I. Gentilini, K. Nagamatsu, and K. Shimada, “Cycle time based multi-goal path optimization for redundant robotic systems,” in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 1786–1792.

[5] V. Krátký, P. Petráček, V. Spurný, and M. Saska, “Autonomous reflectance transformation imaging by a team of unmanned aerial vehicles,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2302–2309, 2020.

[6] D. J. Gulczynski, J. W. Heath, and C. C. Price, *The Close Enough Traveling Salesman Problem: A Discussion of Several Heuristics.* Springer US, 2006, pp. 271–283.

[7] B. Yuan, M. Orlowska, and S. Sadiq, “On the Optimal Robot Routing Problem in Wireless Sensor Networks,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 9, pp. 1252–1261, 2007.

[8] M. Dunbabin and L. Marques, “Robots for Environmental Monitoring: Significant Advancements and Applications,” *IEEE Robotics & Automation Magazine*, vol. 19, no. 1, pp. 24–39, 2012.

[9] J. Faigl and G. A. Hollinger, “Autonomous data collection using a self-organizing map,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 5, pp. 1703–1715, 2018.

[10] J. Deckerová, V. Krátký, and J. Faigl, “Traveling salesman problem with neighborhoods on a sphere in reflectance transformation imaging scenarios,” 2020, (in review).

[11] I. Gentilini, F. Margot, and K. Shimada, “The travelling salesman problem with neighbourhoods: MINLP solution,” *Optimization Methods and Software*, vol. 28, no. 2, pp. 364–378, 2013.

[12] K. Vicencio, B. Davis, and I. Gentilini, “Multi-goal path planning based on the generalized traveling salesman problem with neighborhoods,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014, pp. 2985–2990.

[13] A. Kovács, “Integrated task sequencing and path planning for robotic remote laser welding,” *International Journal of Production Research*, vol. 54, no. 4, pp. 1210–1224, Feb. 2016.

[14] A. Dumitrescu and J. S. B. Mitchell, “Approximation algorithms for TSP with neighborhoods in the plane,” *Journal of Algorithms*, vol. 48, no. 1, pp. 135–159, 2003.

[15] E. M. Arkin and R. Hassin, “Approximation algorithms for the geometric covering salesman problem,” *Discrete Applied Mathematics*, vol. 55, no. 3, pp. 197–218, 1994.

[16] M. de Berg, J. Gudmundsson, M. J. Katz, C. Levcopoulos, M. H. Overmars, and A. F. van der Stappen, “TSP with neighborhoods of varying size,” *Journal of Algorithms*, vol. 57, no. 1, pp. 22–36, 2005.

[17] K. Elbassioni, A. V. Fishkin, and R. Sitters, “Approximation algorithms for the Euclidean traveling salesman problem with discrete and continuous neighborhoods,” *International Journal of Computational Geometry & Applications*, vol. 19, no. 02, pp. 173–193, 2009.

[18] B. Behdani and J. C. Smith, “An integer-programming-based approach to the close-enough traveling salesman problem,” *INFORMS Journal on Computing*, vol. 26, no. 3, pp. 415–432, 2014.

[19] 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*, vol. 28, no. 4, pp. 752–765, 2016.

[20] W. K. Mennell, “Heuristics for solving three routing problems: Close-enough traveling salesman problem, close-enough vehicle routing problem, sequence-dependent team orienteering problem,” Ph.D. dissertation, University of Maryland, 2009.

[21] S. Alatartsev, M. Augustine, and F. Ortmeier, “Constricting insertion heuristic for traveling salesman problem with neighborhoods,” in *International Conference on Automated Planning and Scheduling (ICAPS)*, 2013, pp. 2–10.

[22] J. Faigl, “GSOA: growing self-organizing array - unsupervised learning for the close-enough traveling salesman problem and other routing problems,” *Neurocomputing*, vol. 312, pp. 120–134, 2018.

[23] J. Faigl, P. Váňa, and J. Deckerová, “Fast heuristics for the 3-d multi-goal path planning based on the generalized traveling salesman problem with neighborhoods,” *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2439–2446, 2019.

[24] D. L. Applegate, R. E. Bixby, V. Chvátal, and W. J. Cook, *The Traveling Salesman Problem: A Computational Study.* Princeton, NJ, USA: Princeton University Press, 2007.

[25] F. Carrabs, C. Cerrone, R. Cerulli, and M. Gaudioso, “A novel discretization scheme for the close enough traveling salesman problem,” *Computers & Operations Research*, vol. 78, pp. 163–171, 2017.

[26] F. Carrabs, C. Cerrone, R. Cerulli, and C. D’Ambrosio, “Improved upper and lower bounds for the close enough traveling salesman problem,” in *Green, Pervasive, and Cloud Computing*, M. H. A. Au, A. Castiglione, K.-K. R. Choo, F. Palmieri, and K.-C. Li, Eds. Springer International Publishing, 2017, pp. 165–177.

[27] F. Carrabs, C. Cerrone, R. Cerulli, and B. Golden, “An adaptive heuristic approach to compute upper and lower bounds for the close-enough traveling salesman problem,” *INFORMS Journal on Computing*, 04 2020.

[28] J. Faigl and J. Deckerová, “On unsupervised learning based multi-goal path planning for visiting 3d regions,” in *International Conference on Robotics and Artificial Intelligence (ICRAI)*, 2018, pp. 45–50.

[29] S. Arora, “Polynomial time approximation schemes for euclidean tsp and other geometric problems,” in *37th Conference on Foundations of Computer Science.* IEEE, 1996, pp. 2–11.

[30] J. Faigl, “On the performance of self-organizing maps for the non-euclidean traveling salesman problem in the polygonal domain,” *Information Sciences*, vol. 181, pp. 4214–4229, 2011.

[31] G. Best, J. Faigl, and R. Fitch, “Online planning for multi-robot active perception with self-organising maps,” *Autonomous Robots*, vol. 42, no. 4, pp. 715–738, 2018.

[32] J. Faigl, “Data collection path planning with spatially correlated measurements using growing self-organizing array,” *Applied Soft Computing*, vol. 75, pp. 130–147, 2019.

[33] J. Dong, N. Yang, and M. Chen, “Heuristic approaches for a tsp variant: The automatic meter reading shortest tour problem,” in *Extending the Horizons: Advances in Computing, Optimization, and Decision Technologies.* Springer US, 2007, pp. 145–163.

[34] X. Wang, B. Golden, and E. Wasil, “A steiner zone variable neighborhood search heuristic for the close-enough traveling salesman problem,” *Computers & Operations Research*, vol. 101, pp. 200–219, 2019.

[35] M. Antonescu and C. Bîră, “Discrete gravitational search algorithm (dgsa) applied for the close-enough travelling salesman problem (tsp/cetsp),” in *International Semiconductor Conference (CAS).* IEEE, 2019, pp. 123–126.

[36] S. Somhom, A. Modares, and T. Enkawa, “A self-organising model for the travelling salesman problem,” *Journal of the Operational Research Society*, pp. 919–928, 1997.

[37] G. Laporte, A. Asef-Vaziri, and C. Sriskandarajah, “Some applications of the generalized travelling salesman problem,” *Journal of the Operational Research Society*, vol. 47, no. 12, pp. 1461–1467, 1996.

[38] C.-M. Pintea, P. C. Pop, and C. Chira, “The generalized traveling salesman problem solved with ant algorithms,” *Journal of Universal Computer Science*, vol. 13, no. 7, pp. 1065–1075, 2007.

[39] J. Silberholz and B. Golden, “The generalized traveling salesman problem: A new genetic algorithm approach,” in *Extending the horizons: advances in computing, optimization, and decision technologies.* Springer, 2007, pp. 165–181.

[40] C. E. Noon and J. C. Bean, “An efficient transformation of the generalized traveling salesman problem,” *INFOR: Information Systems and Operational Research*, vol. 31, no. 1, pp. 39–44, 1993.

[41] K. Helsgaun, “Solving the equality generalized traveling salesman problem using the Lin-Kernighan-Helsgaun Algorithm,” *Mathematical Programming Computation*, vol. 7, no. 3, pp. 269–287, 2015.

[42] S. L. Smith and F. Imeson, “GLNS: An effective large neighborhood search heuristic for the generalized traveling salesman problem,” *Computers & Operations Research*, vol. 87, no. C, pp. 1–19, Nov. 2017.

[43] I. Gentilini, “Multi-goal path optimization for robotic systems with redundancy based on the traveling salesman problem with neighborhoods,” Ph.D. dissertation, Carnegie Mellon University, 2012.

[44] M. Held and R. M. Karp, “A dynamic programming approach to sequencing problems,” *Journal of the Society for Industrial and Applied mathematics*, vol. 10, no. 1, pp. 196–210, 1962.

[45] T. Smith, T. Meyer, and G. Thompson, “Lower bounds for the symmetric travelling salesman problem from lagrangean relaxations,” *Discrete Applied Mathematics*, vol. 26, no. 2, pp. 209–217, 1990.

[46] M. Karpinski and R. Schmied, “On approximation lower bounds for tsp with bounded metrics,” 2012, (preprint).

[47] I. Bercea, “Improved bounds for the traveling salesman problem with neighborhoods on uniform disks,” in *Canadian Conference on Computational Geometry (CCCG)*, 2018.

[48] J. Clausen, “Branch and bound algorithms-principles and examples,” *Department of Computer Science, University of Copenhagen*, pp. 1–30, 1999.

[49] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, “Julia: A fresh approach to numerical computing,” *SIAM review*, vol. 59, no. 1, pp. 65–98, 2017.

[50] I. Dunning, J. Huchette, and M. Lubin, “Jump: A modeling language for mathematical optimization,” *SIAM Review*, vol. 59, no. 2, pp. 295–320, 2017.

[51] I. I. Cplex, “V12. 1: User’s manual for cplex,” *International Business Machines Corporation*, vol. 46, no. 53, p. 157, 2009.

[52] A. Wächter and L. Biegler, “On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming,” *Mathematical programming*, vol. 106, pp. 25–57, 03 2006.

[53] K. Vicencio, “Randomly generated GTSPN instances,” http://robotics.pr.erau.edu/data/gtspn.zip, 2014, [cited 21 May 2021]. [Online]. Available: http://robotics.pr.erau.edu/data/gtspn.zip

Appendix A
Computational Results

Table A.1: Lower bound values $\mathcal{LB}$ obtained for the 2D CETSP problem instances with various *Overlap ratio.*

Problem	Mennell [20]		Branch and Bound [19]		$(lb/ub)Alg$ [27]		**Proposed BNB**	
	$\mathcal{LB}$	T	$\mathcal{LB}$	T	$\mathcal{LB}$	T	$\mathcal{LB}$	T
			Overlap ratios (0.02)					
d493-or0.02	57.46	< 0.5 s	**146.33**	4 hours	129.05	4 min	137.92	16 hours
dsj1000-or0.02	100.19	< 0.5 s	**559.14**	4 hours	521.52	16 min	496.15	16 hours
kroD100-or0.02	78.83	< 0.5 s	**142.87**	4 hours	118.64	2 min	140.80	16 hours
lin318-or0.02	977.43	< 0.5 s	**1 990.90**	4 hours	1 830.97	4 min	1 902.99	16 hours
pcb442-or0.02	32.51	< 0.5 s	**185.85**	4 hours	177.36	13 min	174.51	16 hours
rat195-or0.02	28.20	< 0.5 s	**108.10**	4 hours	93.72	9 min	103.69	16 hours
rd400-or0.02	284.03	< 0.5 s	567.19	4 hours	**609.79**	11 min	534.41	16 hours
			Overlap ratios (0.10)					
d493-or0.10	31.73	< 0.5 s	**100.72**	53 s	91.90	33 s	**100.72**	11 hours
dsj1000-or0.10	10.17	< 0.5 s	**373.73**	4 hours	317.33	1 min	361.82	16 hours
kroD100-or0.10	0.00	< 0.5 s	**89.67**	1 s	85.39	3 min	**89.67**	3 min
lin318-or0.10	0.00	< 0.5 s	**1 394.63**	2 hours	1 139.23	26 s	1 358.76	16 hours
pcb442-or0.10	1.87	< 0.5 s	**137.45**	4 hours	110.99	3 min	131.54	16 hours
rat195-or0.10	0.00	< 0.5 s	**67.99**	17 s	65.41	1 min	**67.99**	48 min
rd400-or0.10	0.00	< 0.5 s	**432.80**	4 hours	329.82	1 min	406.58	16 hours
			Overlap ratios (0.30)					
d493-or0.30	16.75	< 0.5 s	**69.76**	< 0.5 s	68.64	2 s	**69.76**	1 min
dsj1000-or0.30	0.00	< 0.5 s	**199.95**	< 0.5 s	193.50	6 s	179.69	30 min
kroD100-or0.30	0.00	< 0.5 s	**58.54**	< 0.5 s	56.89	< 0.5 s	**58.54**	12 s
lin318-or0.30	0.00	< 0.5 s	**765.96**	< 0.5 s	754.21	1 s	**765.96**	1 min
pcb442-or0.30	0.00	< 0.5 s	**83.54**	< 0.5 s	81.88	< 0.5 s	**83.54**	2 min
rat195-or0.30	0.00	< 0.5 s	**45.70**	< 0.5 s	44.51	< 0.5 s	**45.70**	26 s
rd400-or0.30	0.00	< 0.5 s	**224.84**	< 0.5 s	219.29	4 s	**224.84**	2 min

Table A.2: Lower bound values $\mathcal{LB}$ obtained for the 2D CETSP problem instances with various *Varied ratio.*

Problem	Mennell [20]		Branch and Bound [19]		$(lb/ub)Alg$ [27]		**Proposed BNB**	
	$\mathcal{LB}$	T	$\mathcal{LB}$	T	$\mathcal{LB}$	T	$\mathcal{LB}$	T
bonus1000	0.00	< 0.5 s	**359.38**	18 min	285.32	1 min	332.06	16 hours
bubbles1	60.62	< 0.5 s	**349.14**	< 0.5 s	NA*	NA*	**349.13**	4 s
bubbles2	60.62	< 0.5 s	**428.28**	1 s	NA*	NA*	**428.28**	10 s
bubbles3	60.62	< 0.5 s	**529.96**	2 s	NA*	NA*	**529.95**	24 min
bubbles4	60.62	< 0.5 s	**690.58**	9 s	NA*	NA*	678.53	16 hours
bubbles5	60.62	< 0.5 s	**851.82**	37 s	NA*	NA*	820.43	16 hours
bubbles6	60.62	< 0.5 s	**993.98**	14 s	NA*	NA*	945.30	16 hours
bubbles7	60.62	< 0.5 s	**1 123.52**	50 s	NA*	NA*	1 055.25	16 hours
bubbles8	60.62	< 0.5 s	**1 252.71**	1 min	NA*	NA*	1 158.90	16 hours
bubbles9	60.62	< 0.5 s	**1 374.41**	2 min	NA*	NA*	1 260.83	16 hours
chaoSingleDep	439.26	< 0.5 s	**1 000.15**	12 s	831.06	1 min	972.14	16 hours
concentricCircles1	14.00	< 0.5 s	**53.16**	< 0.5 s	45.84	6 s	**53.16**	42 s
concentricCircles2	43.59	< 0.5 s	**149.87**	< 0.5 s	114.56	51 s	148.68	16 hours
concentricCircles3	115.28	< 0.5 s	**247.62**	1 s	191.50	6 min	244.76	16 hours
concentricCircles4	161.11	< 0.5 s	**358.89**	5 s	289.36	1 min	352.02	16 hours
concentricCircles5	249.98	< 0.5 s	**459.41**	10 s	392.89	5 min	450.15	16 hours
rotatingDiamonds1	6.20	< 0.5 s	**32.39**	< 0.5 s	30.73	7 s	**32.39**	2 s
rotatingDiamonds2	13.87	< 0.5 s	**140.48**	< 0.5 s	111.44	2 min	**140.48**	1 hour
rotatingDiamonds3	27.87	< 0.5 s	**348.61**	8 s	272.76	10 min	339.11	16 hours
rotatingDiamonds4	35.57	< 0.5 s	**593.35**	18 s	572.85	4 min	569.46	16 hours
rotatingDiamonds5	69.57	< 0.5 s	**1 106.58**	53 s	1 098.37	3 min	1 011.90	16 hours
team1_100	33.52	< 0.5 s	**307.34**	3 s	270.49	1 min	**307.34**	1 min
team2_200	0.00	< 0.5 s	**246.68**	16 s	232.87	36 s	**246.68**	1 min
team3_300	8.75	< 0.5 s	**447.53**	31 s	330.92	56 s	424.41	16 hours
team4_400	20.59	< 0.5 s	**507.30**	43 s	391.45	2 min	477.33	16 hours
team5_499	231.21	< 0.5 s	**524.59**	1 min	481.49	13 min	492.52	16 hours
team6_500	0.00	< 0.5 s	**225.22**	11 min	217.74	8 s	**225.22**	4 min

* NA - Results not reported in [27]

Table A.3: Lower bound values $\mathcal{LB}$ obtained for the 2D CETSP problem instances with various *Arbitrary radius.*

Problem	Mennell [20]		Branch and Bound [19]		$(lb/ub)Alg$ [27]		**Proposed BNB**	
	$\mathcal{LB}$	T	$\mathcal{LB}$	T	$\mathcal{LB}$	T	$\mathcal{LB}$	T
bonus1000rdmRad	NA*	NA*	506.13	4 hours	**700.54**	12 min	442.17	16 hours
d493rdmRad	NA*	NA*	**125.31**	4 hours	116.57	1 min	120.83	16 hours
dsj1000rdmRad	NA*	NA*	509.74	4 hours	**545.38**	4 min	435.45	16 hours
kroD100rdmRad	NA*	NA*	136.62	4 hours	119.46	46 s	**137.06**	16 hours
lin318rdmRad	NA*	NA*	**1 807.68**	4 hours	1 719.54	59 s	1 672.64	16 hours
pcb442rdmRad	NA*	NA*	175.83	4 hours	**181.21**	2 min	165.93	16 hours
rat195rdmRad	NA*	NA*	**68.22**	5 s	65.12	9 s	**68.22**	1 hour
rd400rdmRad	NA*	NA*	571.48	4 hours	**880.91**	18 min	555.73	16 hours
team1_100rdmRad	NA*	NA*	**388.54**	4 min	350.78	20 s	**388.54**	44 min
team2_200rdmRad	NA*	NA*	488.18	4 hours	428.10	3 min	**495.17**	16 hours
team3_300rdmRad	NA*	NA*	**378.09**	11 min	342.65	16 s	367.11	16 hours
team4_400rdmRad	NA*	NA*	549.91	4 hours	**737.16**	10 min	540.06	16 hours
team5_499rdmRad	NA*	NA*	**442.64**	4 hours	405.46	51 s	412.87	16 hours
team6_500rdmRad	NA*	NA*	489.61	4 hours	**507.43**	5 min	471.62	16 hours

* NA - Results not reported in [20]

Table A.4: Lower bound values $\mathcal{LB}$ obtained for the 3D CETSP problem instances with various *Overlap ratio*.

Problem	Branch and Bound [19]		**Proposed BNB**	
	$\mathcal{LB}$	T	$\mathcal{LB}$	T
	Overlap ratios (0.02)			
d493-or0.02	**496.79**	4 hours	441.76	16 hours
dsj1000-or0.02	**751.51**	4 hours	620.84	16 hours
kroD100-or0.02	**148.23**	4 hours	145.86	16 hours
lin318-or0.02	**1 994.37**	4 hours	1 902.80	16 hours
pcb442-or0.02	**186.38**	4 hours	174.85	16 hours
rat195-or0.02	**126.49**	4 hours	122.54	16 hours
rd400-or0.02	**868.19**	24 min	810.65	16 hours
	Overlap ratios (0.10)			
d493-or0.10	**421.16**	4 hours	396.59	16 hours
dsj1000-or0.10	**602.99**	4 hours	477.17	16 hours
kroD100-or0.10	**91.66**	7 s	**91.66**	2 min
lin318-or0.10	**1 398.25**	4 hours	1 361.17	16 hours
pcb442-or0.10	**137.95**	4 hours	132.10	16 hours
rat195-or0.10	**88.72**	4 hours	85.28	16 hours
rd400-or0.10	**752.42**	4 hours	672.62	16 hours
	Overlap ratios (0.30)			
d493-or0.30	**325.21**	31 s	**325.21**	5 hours
dsj1000-or0.30	**267.75**	25 s	225.83	16 hours
kroD100-or0.30	**58.93**	< 0.5 s	**58.93**	18 s
lin318-or0.30	**766.83**	< 0.5 s	**766.83**	1 min
pcb442-or0.30	**83.72**	< 0.5 s	**83.72**	2 min
rat195-or0.30	**47.89**	< 0.5 s	**47.89**	36 s
rd400-or0.30	**450.72**	4 hours	441.94	16 hours

Table A.5: Lower bound values $\mathcal{LB}$ obtained for the 3D CETSP problem instances with various *Varied ratio.*

Problem	Branch and Bound [19]		**Proposed BNB**	
	$\mathcal{LB}$	T	$\mathcal{LB}$	T
bonus1000	**472.56**	4 hours	386.16	16 hours
bubbles1	NA*	NA*	**349.13**	6 s
bubbles2	NA*	NA*	**428.28**	16 s
bubbles3	NA*	NA*	**529.95**	26 min
bubbles4	NA*	NA*	**677.95**	16 hours
bubbles5	NA*	NA*	**819.23**	16 hours
bubbles6	NA*	NA*	**944.20**	16 hours
bubbles7	NA*	NA*	**1 053.57**	16 hours
bubbles8	NA*	NA*	**1 158.06**	16 hours
bubbles9	NA*	NA*	NA*	NA*
chaoSingleDep	NA*	NA*	**971.18**	16 hours
concentricCircles1	NA*	NA*	**53.16**	1 min
concentricCircles2	NA*	NA*	**148.72**	16 hours
concentricCircles3	NA*	NA*	**244.84**	16 hours
concentricCircles4	NA*	NA*	**351.71**	16 hours
concentricCircles5	NA*	NA*	**450.18**	16 hours
rotatingDiamonds1	NA*	NA*	**32.39**	3 s
rotatingDiamonds2	NA*	NA*	**140.48**	1 hour
rotatingDiamonds3	NA*	NA*	**338.77**	16 hours
rotatingDiamonds4	NA*	NA*	**568.48**	16 hours
rotatingDiamonds5	NA*	NA*	**1 009.42**	16 hours
team1_100	**690.30**	4 hours	677.41	16 hours
team2_200	**273.38**	9 min	**273.38**	13 hours
team3_300	**762.68**	4 hours	714.44	16 hours
team4_400	**509.80**	4 hours	477.71	16 hours
team5_499	**705.63**	4 hours	637.46	16 hours
team6_500	**230.92**	< 0.5 s	**230.92**	10 min

* NA - Results not reported in [19]; results by the BNB have not been determined in predefined time

Table A.6: Lower bound values $\mathcal{LB}$ obtained for the 3D CETSP problem instances with various *Arbitrary radius.*

Problem	Branch and Bound [19]		**Proposed BNB**	
	$\mathcal{LB}$	T	$\mathcal{LB}$	T
bonus1000rdmRad	**578.64**	4 hours	488.23	16 hours
d493rdmRad	**438.70**	4 hours	410.85	16 hours
dsj1000rdmRad	**696.29**	4 hours	566.88	16 hours
kroD100rdmRad	137.76	4 hours	**143.03**	16 hours
lin318rdmRad	**1 807.68**	4 hours	1 670.27	16 hours
pcb442rdmRad	**177.23**	4 hours	167.49	16 hours
rat195rdmRad	**82.10**	13 min	**82.10**	2 hours
rd400rdmRad	**876.28**	4 hours	833.38	16 hours
team1_100rdmRad	**726.68**	4 hours	712.23	16 hours
team2_200rdmRad	525.31	4 hours	**525.73**	16 hours
team3_300rdmRad	**676.18**	4 hours	619.06	16 hours
team4_400rdmRad	**551.05**	4 hours	541.83	16 hours
team5_499rdmRad	**599.74**	4 hours	519.46	16 hours
team6_500rdmRad	**507.12**	4 hours	481.89	16 hours

Table A.7: Lower bound values $\mathcal{LB}$ obtained for the TSPNS problem instances and GTSPN problem instances.

Problem	Proposed BNB		Problem	Proposed BNB	
	$\mathcal{LB}$	T		$\mathcal{LB}$	T
`GTSPN - 3D`			`TSPNS - 3D`		
3D_30_6_a	2 355.65	16 hours	sphere_cap_rand_nol_10x1	6.12	3 s
3D_30_6_b	2 366.88	16 hours	sphere_cap_reg_nol_10x1	5.01	7 s
3D_30_6_c	2 469.97	16 hours	sphere_rand_nol_10x1	8.50	27 s
3D_30_6_d	2 399.34	16 hours	sphere_rect_cap_reg_nol_10x1	4.91	7 s
3D_30_6_e	2 080.82	16 hours	sphere_reg_nol_10x1	10.11	1 min
3D_30_6_f	2 001.75	16 hours	sphere_band_rand_nol_25x1	7.41	1 min
3D_35_6_a	2 510.82	16 hours	sphere_cap_rand_nol_25x1	6.18	3 min
3D_35_6_b	2 593.22	16 hours	sphere_cap_reg_nol_25x1	6.74	1 hour
3D_35_6_c	2 220.29	16 hours	sphere_rand_nol_25x1	10.62	52 min
3D_35_6_d	2 543.07	16 hours	sphere_rect_cap_rand_nol_25x1	8.56	29 min
3D_35_6_e	2 369.09	16 hours	sphere_reg_nol_25x1	13.23	16 hours
3D_35_6_f	NA*	NA*	sphere_rect_cap_reg_nol_25x1	6.46	29 min
3D_40_6_a	2 285.08	16 hours	sphere_band_rand_nol_50x1	9.62	16 hours
3D_40_6_b	2 676.64	16 hours	sphere_cap_rand_nol_50x1	10.46	16 hours
3D_40_6_c	2 370.80	16 hours	sphere_cap_reg_nol_50x1	7.70	16 hours
3D_40_6_d	2 531.37	16 hours	sphere_rand_nol_50x1	12.04	16 hours
3D_40_6_e	2 495.36	16 hours	sphere_rect_cap_rand_nol_50x1	7.22	16 hours
3D_40_6_f	2 191.26	16 hours	sphere_rect_cap_reg_nol_50x1	7.58	16 hours
3D_45_6_a	2 363.86	16 hours	sphere_reg_nol_50x1	13.15	16 hours
3D_45_6_b	2 856.89	16 hours	sphere_rand_nol_100x1	13.11	16 hours
3D_45_6_c	2 676.80	16 hours	sphere_reg_nol_100x1	13.28	16 hours
3D_45_6_d	2 418.94	16 hours	sphere_rand_nol_500x1	12.24	16 hours
3D_45_6_e	2 639.43	16 hours	sphere_rand_ol_10x1	7.56	10 s
3D_45_6_f	2 608.16	16 hours	sphere_cap_reg_ol_25x1	3.96	42 s
3D_50_6_a	2 397.78	16 hours	sphere_rand_ol_25x1	9.86	9 min
3D_50_6_b	2 346.24	16 hours	sphere_band_rand_ol_50x1	9.39	5 hours
3D_50_6_c	2 296.86	16 hours	sphere_rand_ol_50x1	11.82	16 hours
3D_50_6_d	2 545.90	16 hours	sphere_rand_ol_100x1	12.02	16 hours
3D_50_6_e	2 673.90	16 hours			
3D_50_6_f	2 733.57	16 hours			
`GTSPN - 7D`					
7D_30_6_a	3 959.10	16 hours			
7D_35_6_a	3 674.35	16 hours			
7D_40_6_a	3 746.87	16 hours			
7D_45_6_a	3 643.74	16 hours			
7D_50_6_a	3 897.62	16 hours			

* NA - Results by the BNB have not been determined in predefined time

Table A.8: Performance indicators of CETSP solvers in solution of problems with specified *Overlap ratio* in 2D. Together with the perfomance indicators, the lower bound value $\mathcal{LB}$ and the upper bound value $\mathcal{UB}$ obtained by the proposed BNB solver are reported, whereas the gap (57) between the $\mathcal{LB}$ and $\mathcal{UB}$ is denoted as %GAP.

Problem	$\mathcal{L}_{ref}$	$\mathcal{LB}$	$\mathcal{UB}$	%GAP	T	SZ2 [20]				$(lb/ub)Alg$ [27]				GSOA [22]			
						%PDB	%PDM	%GAP	T	%PDB	%PDM	%GAP	T	%PDB	%PDM	%GAP	T
						Overlap ratios (0.02)											
d493-or0.02	205.39	137.92	261.91	47.34	16 hours	3.95	3.95	35.40	< 0.5 s	**0.00**	**0.00**	**32.85**	4 min	1.03	2.14	33.53	< 0.5 s
dsj1000-or0.02	911.58	496.15	1 380.77	64.07	16 hours	10.89	10.89	50.92	< 0.5 s	4.83	4.83	48.08	16 min	**0.00**	**1.82**	**45.57**	1 s
kroD100-or0.02	160.09	140.80	168.88	16.63	16 hours	2.29	2.29	14.02	< 0.5 s	**0.00**	**0.00**	**12.05**	2 min	0.47	2.16	12.46	< 0.5 s
lin318-or0.02	2 902.53	1 902.99	3 652.78	47.90	16 hours	0.41	0.41	34.71	< 0.5 s	**0.00**	**0.00**	**34.44**	4 min	1.46	3.18	35.38	< 0.5 s
pcb442-or0.02	333.24	174.51	465.29	62.49	16 hours	**0.00**	**0.00**	**47.63**	< 0.5 s	1.28	1.28	48.29	13 min	1.31	2.74	48.31	< 0.5 s
rat195-or0.02	162.70	103.69	199.28	47.97	16 hours	5.28	5.28	39.47	< 0.5 s	2.34	2.34	37.73	9 min	**0.00**	**2.03**	**36.27**	< 0.5 s
rd400-or0.02	1 049.33	534.41	1 368.32	60.94	16 hours	1.60	1.60	49.87	< 0.5 s	3.48	3.48	50.78	11 min	**0.00**	**1.53**	**49.07**	< 0.5 s
Mean values						*3.49*	*3.49*	*38.86*	< 0.5 s	*1.70*	*1.70*	*37.75*	8 min	*0.61*	*2.23*	*37.23*	< 0.5 s
						Overlap ratios (0.10)											
d493-or0.10	100.72	**100.72**	**100.72**	**0.00**	11 hours	10.11	10.11	9.18	< 0.5 s	**0.00**	**0.00**	**0.00**	33 s	3.60	6.64	3.47	< 0.5 s
dsj1000-or0.10	374.06	361.82	401.99	9.99	16 hours	8.16	8.16	10.57	< 0.5 s	**0.00**	**0.00**	**3.27**	1 min	2.13	6.94	5.29	< 0.5 s
kroD100-or0.10	89.67	**89.67**	**89.67**	**0.00**	3 min	7.82	7.82	7.25	< 0.5 s	**0.00**	**0.00**	**0.00**	3 min	1.12	2.10	1.10	< 0.5 s
lin318-or0.10	1 405.07	1 358.76	1 479.23	8.14	16 hours	10.80	10.80	12.72	< 0.5 s	**0.00**	**0.00**	**3.30**	26 s	3.73	6.19	6.77	< 0.5 s
pcb442-or0.10	146.03	131.54	165.25	20.40	16 hours	11.11	11.11	18.94	< 0.5 s	**0.00**	**0.00**	**9.93**	3 min	5.30	6.68	14.46	< 0.5 s
rat195-or0.10	67.99	**67.99**	**67.99**	**0.00**	48 min	9.07	9.07	8.32	< 0.5 s	**0.22**	**0.22**	**0.22**	1 min	1.48	4.00	1.46	< 0.5 s
rd400-or0.10	460.21	406.58	553.96	26.60	16 hours	13.41	13.41	22.10	< 0.5 s	**0.00**	**0.00**	**11.65**	1 min	4.51	6.15	15.47	< 0.5 s
Mean values						*6.78*	*6.78*	*25.79*	< 0.5 s	*0.87*	*0.87*	*20.90*	5 min	*1.87*	*3.88*	*22.04*	< 0.5 s
						Overlap ratios (0.30)											
d493-or0.30	69.76	**69.76**	**69.76**	**0.00**	1 min	1.42	1.42	1.40	< 0.5 s	**0.00**	**0.00**	**0.00**	2 s	0.94	2.06	0.93	< 0.5 s
dsj1000-or0.30	179.69	**179.69**	**179.69**	**0.00**	30 min	12.37	12.37	11.01	< 0.5 s	**11.28**	**11.28**	**10.13**	6 s	15.27	19.17	13.25	< 0.5 s
kroD100-or0.30	58.54	**58.54**	**58.54**	**0.00**	12 s	**0.00**	**0.00**	**0.00**	< 0.5 s	**0.00**	**0.00**	**0.00**	< 0.5 s	1.17	2.25	1.15	< 0.5 s
lin318-or0.30	765.96	**765.96**	**765.96**	**0.00**	1 min	2.15	2.15	2.10	< 0.5 s	**0.00**	**0.00**	**0.00**	1 s	3.09	5.13	3.00	< 0.5 s
pcb442-or0.30	83.54	**83.54**	**83.54**	**0.00**	2 min	1.89	1.89	1.86	< 0.5 s	**0.00**	**0.00**	**0.00**	< 0.5 s	2.60	6.03	2.53	< 0.5 s
rat195-or0.30	45.70	**45.70**	**45.70**	**0.00**	26 s	0.20	0.20	0.19	< 0.5 s	**0.00**	**0.00**	**0.00**	< 0.5 s	1.99	3.18	1.95	< 0.5 s
rd400-or0.30	224.84	**224.84**	**224.84**	**0.00**	2 min	3.76	3.76	3.62	< 0.5 s	**0.00**	**0.00**	**0.00**	4 s	3.12	6.29	3.03	< 0.5 s
Mean values						*5.56*	*5.56*	*18.16*	< 0.5 s	*1.12*	*1.12*	*14.42*	3 min	*2.59*	*4.69*	*15.93*	< 0.5 s

Table A.9: Performance indicators of CETSP solvers in solution of problems with *Varied overlap ratio* in 2D. Together with the perfomance indicators, the lower bound value $\mathcal{LB}$ and the upper bound value $\mathcal{UB}$ obtained by the proposed BNB solver are reported, whereas the gap (57) between the $\mathcal{LB}$ and $\mathcal{UB}$ is denoted as %GAP.

Problem	$\mathcal{L}_{ref}$	$\mathcal{LB}$	$\mathcal{UB}$	%GAP	T	SZ2 [20]				$(lb/ub)Alg$ [27]				GSOA [22]			
						%PDB	%PDM	%GAP	T	%PDB	%PDM	%GAP	T	%PDB	%PDM	%GAP	T
bonus1000	387.13	332.06	481.01	30.97	16 hours	13.55	13.55	24.46	< 0.5 s	**0.00**	**0.00**	**14.23**	1 min	5.79	9.97	18.92	< 0.5 s
bubbles1	349.13	**349.13**	**349.13**	**0.00**	4 s	0.70	0.70	0.70	< 0.5 s	NA*	NA*	NA*	NA*	**0.25**	**0.69**	**0.25**	< 0.5 s
bubbles2	428.28	**428.28**	**428.28**	**0.00**	10 s	**0.24**	**0.24**	**0.24**	< 0.5 s	NA*	NA*	NA*	NA*	0.97	1.71	0.96	< 0.5 s
bubbles3	529.95	**529.95**	**529.95**	**0.00**	24 min	12.38	12.38	11.02	< 0.5 s	NA*	NA*	NA*	NA*	**0.29**	**0.74**	**0.29**	< 0.5 s
bubbles4	817.69	678.53	917.57	26.05	16 hours	5.60	5.60	21.42	< 0.5 s	NA*	NA*	NA*	NA*	**0.00**	**0.81**	**17.02**	< 0.5 s
bubbles5	1 095.08	820.43	1 374.82	40.32	16 hours	4.27	4.27	28.15	< 0.5 s	NA*	NA*	NA*	NA*	**0.00**	**1.14**	**25.08**	< 0.5 s
bubbles6	1 367.07	945.30	1 817.29	47.98	16 hours	7.11	7.11	35.44	< 0.5 s	NA*	NA*	NA*	NA*	**0.00**	**1.03**	**30.85**	< 0.5 s
bubbles7	1 654.02	1 055.25	2 379.84	55.66	16 hours	11.24	11.24	42.65	< 0.5 s	NA*	NA*	NA*	NA*	**0.00**	**1.83**	**36.20**	< 0.5 s
bubbles8	1 914.96	1 158.90	3 021.19	61.64	16 hours	17.53	17.53	48.51	< 0.5 s	NA*	NA*	NA*	NA*	**0.00**	**1.32**	**39.48**	< 0.5 s
bubbles9	2 152.66	1 260.83	3 715.75	66.07	16 hours	21.99	21.99	51.99	< 0.5 s	NA*	NA*	NA*	NA*	**0.00**	**2.28**	**41.43**	< 0.5 s
chaoSingleDep	1 022.98	972.14	1 091.83	10.96	16 hours	**0.00**	**0.00**	**4.97**	< 0.5 s	1.63	1.63	6.49	1 min	4.92	8.14	9.43	< 0.5 s
concentricCircles1	53.16	**53.16**	**53.16**	**0.00**	42 s	3.30	3.30	3.19	< 0.5 s	**0.00**	**0.00**	**0.00**	6 s	0.25	0.88	0.25	< 0.5 s
concentricCircles2	154.42	148.68	154.42	3.72	16 hours	4.84	4.84	8.17	< 0.5 s	0.25	**0.25**	3.96	51 s	**0.16**	1.71	**3.87**	< 0.5 s
concentricCircles3	272.69	244.76	278.16	12.01	16 hours	0.80	0.80	10.96	< 0.5 s	**0.00**	**0.00**	**10.24**	6 min	0.53	1.04	10.71	< 0.5 s
concentricCircles4	457.78	352.02	500.82	29.71	16 hours	4.85	4.85	26.66	< 0.5 s	1.91	1.91	24.54	1 min	**0.00**	**0.69**	**23.10**	< 0.5 s
concentricCircles5	656.78	450.15	738.26	39.03	16 hours	1.51	1.51	32.48	< 0.5 s	0.39	**0.39**	31.73	5 min	**0.00**	1.71	**31.46**	< 0.5 s
rotatingDiamonds1	32.39	**32.39**	**32.39**	**0.00**	2 s	2.35	2.35	2.30	< 0.5 s	**0.00**	**0.00**	**0.00**	7 s	0.51	1.14	0.51	< 0.5 s
rotatingDiamonds2	140.48	**140.48**	**140.48**	**0.00**	1 hour	1.26	1.26	1.25	< 0.5 s	**0.00**	**0.00**	**0.00**	2 min	0.29	2.18	0.28	< 0.5 s
rotatingDiamonds3	380.89	339.11	390.28	13.11	16 hours	0.65	0.65	11.54	< 0.5 s	**0.00**	**0.00**	**10.97**	10 min	1.32	2.88	12.13	< 0.5 s
rotatingDiamonds4	772.00	569.46	859.08	33.71	16 hours	0.36	0.36	26.50	< 0.5 s	**0.00**	**0.00**	**26.24**	4 min	1.54	9.05	27.36	< 0.5 s
rotatingDiamonds5	1 515.44	1 011.90	1 807.26	44.01	16 hours	**0.00**	**0.00**	**33.23**	< 0.5 s	1.08	1.08	33.94	3 min	5.42	14.34	36.66	< 0.5 s
team1_100	307.34	**307.34**	**307.34**	**0.00**	1 min	6.31	6.31	5.94	< 0.5 s	**0.00**	**0.00**	**0.00**	1 min	1.70	2.63	1.67	< 0.5 s
team2_200	246.68	**246.68**	**246.68**	**0.00**	1 min	12.54	12.54	11.14	< 0.5 s	**0.00**	**0.00**	**0.00**	36 s	2.33	4.72	2.27	< 0.5 s
team3_300	476.43	424.41	536.98	20.96	16 hours	6.13	6.13	16.07	< 0.5 s	**0.00**	**0.00**	**10.92**	56 s	4.03	5.82	14.37	< 0.5 s
team4_400	695.18	477.33	910.75	47.59	16 hours	6.86	6.86	35.74	< 0.5 s	1.08	**1.08**	32.07	2 min	**0.00**	1.30	**31.34**	< 0.5 s
team5_499	708.45	492.52	921.83	46.57	16 hours	2.70	2.70	32.31	< 0.5 s	**0.00**	**0.00**	**30.48**	13 min	2.00	5.87	31.84	< 0.5 s
team6_500	225.22	**225.22**	**225.22**	**0.00**	4 min	3.43	3.43	3.32	< 0.5 s	**0.00**	**0.00**	**0.00**	8 s	2.94	7.56	2.86	< 0.5 s
Mean values						*5.65*	*5.65*	*19.64*	< 0.5 s	*0.35*	*0.35*	*13.10*	3 min	*1.31*	*3.45*	*16.69*	< 0.5 s

* NA - Results not reported in [27]

Table A.10: Performance indicators of CETSP solvers in solution of problems with *Arbitrary radius* in 2D. Together with the perfomance indicators, the lower bound value $\mathcal{LB}$ and the upper bound value $\mathcal{UB}$ obtained by the proposed BNB solver are reported, whereas the gap (57) between the $\mathcal{LB}$ and $\mathcal{UB}$ is denoted as %GAP.

Problem	$\mathcal{L}_{ref}$	$\mathcal{LB}$	$\mathcal{UB}$	%GAP	T	SZ2 [20]				$(lb/ub)Alg$ [27]				GSOA [22]			
						%PDB	%PDM	%GAP	T	%PDB	%PDM	%GAP	T	%PDB	%PDM	%GAP	T
bonus1000rdmRad	955.41	442.17	1 361.69	67.53	16 hours	3.89	3.89	55.45	41 s	**0.00**	**0.00**	**53.72**	12 min	1.31	3.73	54.32	1 s
d493rdmRad	134.74	120.83	152.02	20.51	16 hours	5.45	5.45	14.95	7 s	**0.00**	**0.00**	**10.32**	1 min	2.79	6.28	12.75	< 0.5 s
dsj1000rdmRad	625.25	435.45	859.70	49.35	16 hours	4.50	4.50	33.35	24 s	**0.00**	**0.00**	**30.36**	4 min	4.46	8.43	33.33	< 0.5 s
kroD100rdmRad	142.36	137.06	144.65	5.25	16 hours	1.43	1.43	5.07	4 s	**0.00**	**0.00**	**3.72**	46 s	2.29	3.54	5.87	< 0.5 s
lin318rdmRad	2 055.77	1 672.64	2 434.86	31.30	16 hours	5.83	5.83	23.12	8 s	**0.00**	**0.00**	**18.64**	59 s	5.25	9.77	22.69	< 0.5 s
pcb442rdmRad	220.44	165.93	281.70	41.10	16 hours	10.45	10.45	31.85	1 min	**0.00**	**0.00**	**24.73**	2 min	2.16	4.84	26.32	< 0.5 s
rat195rdmRad	68.22	**68.22**	**68.22**	**0.00**	1 hour	0.91	0.91	0.89	8 s	**0.00**	**0.00**	**0.00**	9 s	1.26	2.26	1.24	< 0.5 s
rd400rdmRad	1 276.08	555.73	1 562.71	64.44	16 hours	**0.00**	**0.00**	**56.45**	57 s	2.30	2.30	57.43	18 min	0.36	1.80	56.61	< 0.5 s
team1_100rdmRad	388.54	**388.54**	**388.54**	**0.00**	44 min	0.62	0.62	0.62	9 s	**0.00**	**0.00**	**0.00**	20 s	3.83	4.68	3.69	< 0.5 s
team2_200rdmRad	616.82	495.17	697.10	28.97	16 hours	6.32	6.32	24.49	6 s	**0.00**	**0.00**	**19.72**	3 min	1.58	3.48	20.97	< 0.5 s
team3_300rdmRad	378.51	367.11	404.75	9.30	16 hours	4.78	4.78	7.44	5 s	**0.00**	**0.00**	**3.01**	16 s	3.89	9.70	6.64	< 0.5 s
team4_400rdmRad	1 024.97	540.06	1 305.39	58.63	16 hours	0.08	**0.08**	47.35	22 s	0.08	**0.08**	47.35	10 min	**0.00**	1.81	**47.31**	< 0.5 s
team5_499rdmRad	446.19	412.87	502.79	17.88	16 hours	2.11	2.11	9.38	2 min	**0.00**	**0.00**	**7.47**	51 s	4.21	8.20	11.21	< 0.5 s
team6_500rdmRad	626.18	471.62	808.82	41.69	16 hours	8.28	8.28	30.44	11 s	**0.00**	**0.00**	**24.68**	5 min	3.42	6.67	27.17	< 0.5 s
Mean values						*3.90*	*3.90*	*24.35*	28 s	*0.17*	*0.17*	*21.51*	4 min	*2.63*	*5.37*	*23.58*	< 0.5 s

Table A.11: Performance indicators of CETSP solvers in solution of problems with specified *Overlap ratio* in 3D. Together with the perfomance indicators, the lower bound value $\mathcal{LB}$ and the upper bound value $\mathcal{UB}$ obtained by the proposed BNB solver are reported, whereas the gap (57) between the $\mathcal{LB}$ and $\mathcal{UB}$ is denoted as %GAP.

Problem	$\mathcal{L}_{ref}$	$\mathcal{LB}$	$\mathcal{UB}$	%GAP	T	SZ2 [20]				**Proposed GSOA**			
						%PDB	%PDM	%GAP	T	%PDB	%PDM	%GAP	T
					Overlap ratios (0.02)								
d493-or0.02	1 358.42	441.76	1 726.04	74.41	16 hours	**0.00**	**0.00**	**67.48**	6 s	2.61	4.37	68.31	< 0.5 s
dsj1000-or0.02	3 172.35	620.84	4 651.73	86.65	16 hours	**0.00**	**0.00**	**80.43**	26 s	1.52	2.47	80.72	2 s
kroD100-or0.02	201.95	145.86	222.94	34.57	16 hours	0.36	**0.36**	28.04	< 0.5 s	**0.00**	1.02	**27.78**	< 0.5 s
lin318-or0.02	3 078.35	1 902.80	3 941.35	51.72	16 hours	**0.00**	**0.00**	**38.19**	3 s	0.32	2.04	38.39	< 0.5 s
pcb442-or0.02	413.92	174.85	571.43	69.40	16 hours	0.79	**0.79**	58.09	9 s	**0.00**	1.02	**57.76**	< 0.5 s
rat195-or0.02	291.59	122.54	367.87	66.69	16 hours	**0.00**	**0.00**	**57.97**	2 s	3.69	5.05	59.47	< 0.5 s
rd400-or0.02	3 228.07	810.65	4 090.26	80.18	16 hours	**0.00**	**0.00**	**74.89**	4 s	3.99	5.38	75.85	< 0.5 s
Mean values						*0.16*	*0.16*	*57.87*	7 s	*1.73*	*3.05*	*58.33*	< 0.5 s
					Overlap ratios (0.10)								
d493-or0.10	654.54	396.59	975.46	59.34	16 hours	7.92	7.92	43.86	1 s	**0.00**	**3.93**	**39.41**	< 0.5 s
dsj1000-or0.10	901.19	477.17	1 506.60	68.33	16 hours	25.72	25.72	57.88	4 s	**0.00**	**5.11**	**47.05**	1 s
kroD100-or0.10	91.66	**91.66**	**91.66**	**0.00**	2 min	8.63	8.63	7.94	1 s	**1.75**	**2.82**	**1.72**	< 0.5 s
lin318-or0.10	1 469.77	1 361.17	1 486.68	8.44	16 hours	11.24	11.24	16.75	< 0.5 s	**0.00**	**2.65**	**7.39**	< 0.5 s
pcb442-or0.10	155.36	132.10	164.51	19.70	16 hours	17.16	17.16	27.42	1 s	**0.00**	**2.81**	**14.97**	< 0.5 s
rat195-or0.10	102.23	85.28	130.66	34.73	16 hours	20.61	20.61	30.84	< 0.5 s	**0.00**	**1.39**	**16.58**	< 0.5 s
rd400-or0.10	1 466.23	672.62	2 345.80	71.33	16 hours	17.92	17.92	61.10	2 s	**0.00**	**3.72**	**54.13**	< 0.5 s
Mean values						*7.88*	*7.88*	*46.49*	4 s	*0.99*	*3.13*	*42.11*	< 0.5 s
					Overlap ratios (0.30)								
d493-or0.30	325.21	**325.21**	**325.21**	**0.00**	5 hours	6.91	6.91	6.46	8 s	**4.46**	**6.67**	**4.27**	< 0.5 s
dsj1000-or0.30	230.53	225.83	230.53	2.04	16 hours	39.92	39.92	29.98	5 s	**26.03**	**31.89**	**22.27**	1 s
kroD100-or0.30	58.93	**58.93**	**58.93**	**0.00**	18 s	**0.01**	**0.01**	**0.01**	3 s	1.34	2.78	1.33	< 0.5 s
lin318-or0.30	766.83	**766.83**	**766.83**	**0.00**	1 min	**0.02**	**0.02**	**0.02**	5 s	2.50	5.62	2.44	< 0.5 s
pcb442-or0.30	83.72	**83.72**	**83.72**	**0.00**	2 min	**0.33**	**0.33**	**0.33**	4 s	4.14	8.43	3.97	< 0.5 s
rat195-or0.30	47.89	**47.89**	**47.89**	**0.00**	36 s	**0.00**	**0.00**	**0.00**	3 s	3.11	5.58	3.02	< 0.5 s
rd400-or0.30	474.30	441.94	503.96	12.31	16 hours	23.55	23.55	24.58	5 s	**0.00**	**10.09**	**6.82**	< 0.5 s
Mean values						*8.62*	*8.62*	*33.92*	4 s	*2.64*	*5.47*	*30.17*	< 0.5 s

Table A.12: Performance indicators of CETSP solvers in solution of problems with *Varied overlap ratio* in 3D. Together with the perfomance indicators, the lower bound value $\mathcal{LB}$ and the upper bound value $\mathcal{UB}$ obtained by the proposed BNB solver are reported, whereas the gap (57) between the $\mathcal{LB}$ and $\mathcal{UB}$ is denoted as %GAP.

Problem	$\mathcal{L}_{ref}$	$\mathcal{LB}$	$\mathcal{UB}$	%GAP	T	SZ2 [20]				**Proposed GSOA**			
						%PDB	%PDM	%GAP	T	%PDB	%PDM	%GAP	T
bonus1000	793.05	386.16	1 369.51	71.80	16 hours	38.93	38.93	64.95	13 s	**0.00**	**5.82**	**51.31**	1 s
bubbles1	349.13	**349.13**	**349.13**	**0.00**	6 s	NA*	NA*	NA*	NA*	**0.26**	**0.56**	**0.26**	< 0.5 s
bubbles2	428.28	**428.28**	**428.28**	**0.00**	16 s	NA*	NA*	NA*	NA*	**1.30**	**2.04**	**1.28**	< 0.5 s
bubbles3	529.95	**529.95**	**529.95**	**0.00**	26 min	NA*	NA*	NA*	NA*	**0.38**	**0.88**	**0.38**	< 0.5 s
bubbles4	817.27	677.95	918.10	26.16	16 hours	NA*	NA*	NA*	NA*	**0.00**	**0.89**	**17.05**	< 0.5 s
bubbles5	1 090.14	819.23	1 376.81	40.50	16 hours	NA*	NA*	NA*	NA*	**0.00**	**1.12**	**24.85**	< 0.5 s
bubbles6	1 369.10	944.20	1 825.22	48.27	16 hours	NA*	NA*	NA*	NA*	**0.00**	**1.22**	**31.03**	< 0.5 s
bubbles7	1 653.59	1 053.57	2 373.48	55.61	16 hours	NA*	NA*	NA*	NA*	**0.00**	**2.08**	**36.29**	< 0.5 s
bubbles8	1 996.08	1 158.06	3 005.49	61.47	16 hours	NA*	NA*	NA*	NA*	**0.00**	**2.12**	**41.98**	< 0.5 s
bubbles9	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*
chaoSingleDep	1 075.75	971.18	1 091.83	11.05	16 hours	NA*	NA*	NA*	NA*	**0.00**	**3.21**	**9.72**	< 0.5 s
concentricCircles1	53.16	**53.16**	**53.16**	**0.00**	1 min	NA*	NA*	NA*	NA*	**0.26**	**0.54**	**0.26**	< 0.5 s
concentricCircles2	154.37	148.72	154.42	3.70	16 hours	NA*	NA*	NA*	NA*	**0.00**	**1.55**	**3.66**	< 0.5 s
concentricCircles3	273.81	244.84	278.16	11.98	16 hours	NA*	NA*	NA*	NA*	**0.00**	**0.57**	**10.58**	< 0.5 s
concentricCircles4	458.47	351.71	500.82	29.77	16 hours	NA*	NA*	NA*	NA*	**0.00**	**0.92**	**23.29**	< 0.5 s
concentricCircles5	657.27	450.18	738.26	39.02	16 hours	NA*	NA*	NA*	NA*	**0.00**	**1.69**	**31.51**	< 0.5 s
rotatingDiamonds1	32.39	**32.39**	**32.39**	**0.00**	3 s	NA*	NA*	NA*	NA*	**0.27**	**0.82**	**0.27**	< 0.5 s
rotatingDiamonds2	140.48	**140.48**	**140.48**	**0.00**	1 hour	NA*	NA*	NA*	NA*	**0.32**	**1.88**	**0.32**	< 0.5 s
rotatingDiamonds3	384.06	338.77	390.28	13.20	16 hours	NA*	NA*	NA*	NA*	**0.00**	**1.81**	**11.79**	< 0.5 s
rotatingDiamonds4	799.61	568.48	842.19	32.50	16 hours	NA*	NA*	NA*	NA*	**0.00**	**6.22**	**28.91**	< 0.5 s
rotatingDiamonds5	1 678.19	1 009.42	1 807.26	44.15	16 hours	NA*	NA*	NA*	NA*	**0.00**	**3.74**	**39.85**	< 0.5 s
team1_100	828.41	677.41	919.37	26.32	16 hours	**0.00**	**0.00**	**18.23**	4 s	1.18	4.57	19.18	< 0.5 s
team2_200	273.38	**273.38**	**273.38**	**0.00**	13 hours	18.02	18.02	15.27	11 s	**3.06**	**4.48**	**2.97**	< 0.5 s
team3_300	1 507.68	714.44	2 058.69	65.30	16 hours	1.74	**1.74**	53.43	9 s	**0.00**	2.17	**52.61**	< 0.5 s
team4_400	737.72	477.71	1 071.13	55.40	16 hours	9.65	9.65	40.94	7 s	**0.00**	**2.14**	**35.25**	< 0.5 s
team5_499	1 937.18	637.46	2 620.37	75.67	16 hours	**0.00**	**0.00**	**67.09**	30 s	1.91	4.08	67.71	< 0.5 s
team6_500	230.92	**230.92**	**230.92**	**0.00**	10 min	17.70	17.70	15.04	34 s	**8.55**	**15.83**	**7.88**	< 0.5 s
Mean values						*12.29*	*12.29*	*39.28*	15 s	*0.67*	*2.81*	*21.16*	< 0.5 s

* NA - Results not reported in [20]; results by the BNB have not been determined in predefined time

Table A.13: Performance indicators of CETSP solvers in solution of problems with *Arbitrary radius* in 3D. Together with the perfomance indicators, the lower bound value $\mathcal{LB}$ and the upper bound value $\mathcal{UB}$ obtained by the proposed BNB solver are reported, whereas the gap (57) between the $\mathcal{LB}$ and $\mathcal{UB}$ is denoted as %GAP.

Problem	$\mathcal{L}_{ref}$	$\mathcal{LB}$	$\mathcal{UB}$	%GAP	T	SZ2 [20]				**Proposed GSOA**			
						%PDB	%PDM	%GAP	T	%PDB	%PDM	%GAP	T
bonus1000rdmRad	2 179.55	488.23	3 121.72	84.36	16 hours	47.41	47.41	84.80	7 min	**0.00**	**1.88**	**77.60**	1 s
d493rdmRad	744.14	410.85	969.51	57.62	16 hours	10.44	10.44	50.01	1 hour	**0.00**	**1.35**	**44.79**	< 0.5 s
dsj1000rdmRad	1 665.87	566.88	2 416.50	76.54	16 hours	54.48	54.48	77.97	6 min	**0.00**	**1.51**	**65.97**	1 s
kroD100rdmRad	173.45	143.03	183.86	22.21	16 hours	0.63	**0.63**	18.05	12 s	**0.00**	0.85	**17.54**	< 0.5 s
lin318rdmRad	2 170.64	1 670.27	2 557.20	34.68	16 hours	7.55	7.55	28.45	8 min	**0.00**	**6.53**	**23.05**	< 0.5 s
pcb442rdmRad	250.23	167.49	324.48	48.38	16 hours	14.95	14.95	41.77	9 min	**0.00**	**2.29**	**33.06**	< 0.5 s
rat195rdmRad	82.10	**82.10**	**82.10**	**0.00**	2 hours	27.80	27.80	21.75	2 min	**2.38**	**5.55**	**2.32**	< 0.5 s
rd400rdmRad	3 595.97	833.38	4 346.17	80.82	16 hours	**0.00**	**0.00**	**76.82**	3 min	4.98	6.24	77.92	< 0.5 s
team1_100rdmRad	914.86	712.23	1 050.55	32.20	16 hours	**0.00**	**0.00**	**22.15**	12 s	0.77	6.14	22.74	< 0.5 s
team2_200rdmRad	1 053.28	525.73	1 290.61	59.26	16 hours	1.10	**1.10**	50.63	1 min	**0.00**	1.24	**50.09**	< 0.5 s
team3_300rdmRad	1 027.51	619.06	1 265.28	51.07	16 hours	16.59	16.59	48.32	4 min	**0.00**	**2.93**	**39.75**	< 0.5 s
team4_400rdmRad	1 277.76	541.83	1 629.70	66.75	16 hours	**0.00**	**0.00**	**57.60**	17 min	0.52	2.07	57.82	< 0.5 s
team5_499rdmRad	815.98	519.46	1 041.39	50.12	16 hours	15.21	15.21	44.74	1 hour	**0.00**	**4.82**	**36.34**	< 0.5 s
team6_500rdmRad	1 010.06	481.89	1 416.52	65.98	16 hours	16.71	16.71	59.12	26 min	**0.00**	**2.47**	**52.29**	< 0.5 s
Mean values						*15.21*	*15.21*	*48.73*	17 min	*0.62*	*3.28*	*42.95*	< 0.5 s

Table A.14: Performance indicators of TSPNS solvers in solution of problems with *Non-overlapping* and *Overlapping* regions. Together with the perfomance indicators, the lower bound value $\mathcal{LB}$ and the upper bound value $\mathcal{UB}$ obtained by the proposed BNB solver are reported, whereas the gap (57) between the $\mathcal{LB}$ and $\mathcal{UB}$ is denoted as %GAP.

Problem	$\mathcal{L}_{ref}$	$\mathcal{LB}$	$\mathcal{UB}$	%GAP	T	TSPN-LKH [10]				TSPN-GLKH [10]				**Proposed GSOA**			
						%PDB	%PDM	%GAP	T	%PDB	%PDM	%GAP	T	%PDB	%PDM	%GAP	T
						Non-overlaping regions											
sphere_cap_rand_nol_10x1	6.12	**6.12**	**6.12**	**0.00**	3 s	**0.00**	**0.00**	**0.00**	< 0.5 s	**0.00**	**0.00**	**0.00**	< 0.5 s	0.11	0.27	0.11	< 0.5 s
sphere_cap_reg_nol_10x1	5.01	**5.01**	**5.01**	**0.00**	7 s	13.14	13.14	11.61	< 0.5 s	**0.00**	**0.00**	**0.00**	< 0.5 s	0.21	0.48	0.21	< 0.5 s
sphere_rand_nol_10x1	8.50	**8.50**	**8.50**	**0.00**	27 s	28.18	28.18	21.98	< 0.5 s	**0.00**	**0.00**	**0.00**	< 0.5 s	0.49	1.94	0.49	< 0.5 s
sphere_rect_cap_reg_nol_10x1	4.91	**4.91**	**4.91**	**0.00**	7 s	13.44	13.44	11.84	< 0.5 s	**0.00**	**0.00**	**0.00**	< 0.5 s	0.20	0.77	0.20	< 0.5 s
sphere_reg_nol_10x1	10.11	**10.11**	**10.11**	**0.00**	1 min	49.83	49.83	33.26	< 0.5 s	**0.00**	**0.00**	**0.00**	< 0.5 s	0.17	2.48	0.17	< 0.5 s
sphere_band_rand_nol_25x1	7.41	**7.41**	**7.41**	**0.00**	1 min	251.42	251.42	71.55	< 0.5 s	**0.00**	**0.00**	**0.00**	2 s	0.28	0.83	0.28	< 0.5 s
sphere_cap_rand_nol_25x1	6.18	**6.18**	**6.18**	**0.00**	3 min	9.14	9.14	8.38	< 0.5 s	**0.00**	**0.00**	**0.00**	3 s	0.65	1.33	0.65	< 0.5 s
sphere_cap_reg_nol_25x1	6.74	**6.74**	**6.74**	**0.00**	1 hour	5.98	5.98	5.64	< 0.5 s	**0.39**	**0.39**	**0.39**	3 s	1.49	3.07	1.47	< 0.5 s
sphere_rand_nol_25x1	10.62	**10.62**	**10.62**	**0.00**	52 min	32.51	32.51	24.53	< 0.5 s	36.42	36.42	26.70	2 s	**0.46**	**3.86**	**0.46**	< 0.5 s
sphere_rect_cap_rand_nol_25x1	8.56	**8.56**	**8.56**	**0.00**	29 min	34.77	34.77	25.80	< 0.5 s	**0.00**	**0.00**	**0.00**	2 s	0.87	2.20	0.86	< 0.5 s
sphere_reg_nol_25x1	14.99	13.23	14.99	11.75	16 hours	45.63	45.63	39.40	< 0.5 s	1.45	**1.45**	13.01	3 s	**0.95**	4.59	**12.58**	< 0.5 s
sphere_rect_cap_reg_nol_25x1	6.46	**6.46**	**6.46**	**0.00**	29 min	12.73	12.73	11.29	< 0.5 s	**0.00**	**0.00**	**0.00**	3 s	1.10	2.38	1.09	< 0.5 s
sphere_band_rand_nol_50x1	10.14	9.62	10.28	6.44	16 hours	289.96	289.96	75.68	< 0.5 s	**0.00**	**0.00**	**5.16**	19 s	0.68	2.37	5.79	< 0.5 s
sphere_cap_rand_nol_50x1	10.75	10.46	10.88	3.85	16 hours	44.83	44.83	32.78	< 0.5 s	**0.00**	**0.00**	**2.65**	22 s	0.65	5.26	3.28	< 0.5 s
sphere_cap_reg_nol_50x1	8.57	7.70	8.69	11.39	16 hours	9.34	9.34	17.88	< 0.5 s	0.22	**0.22**	10.40	< 0.5 s	**0.00**	2.64	**10.20**	< 0.5 s
sphere_rand_nol_50x1	15.48	12.04	16.82	28.37	16 hours	65.16	65.16	52.90	< 0.5 s	**0.00**	**0.00**	**22.20**	24 s	0.31	3.19	22.44	< 0.5 s
sphere_rect_cap_rand_nol_50x1	7.63	7.22	7.84	7.93	16 hours	8.30	8.30	12.70	< 0.5 s	**0.00**	**0.00**	**5.46**	23 s	1.16	3.23	6.54	< 0.5 s
sphere_rect_cap_reg_nol_50x1	8.14	7.58	8.18	7.41	16 hours	6.45	6.45	12.52	< 0.5 s	1.06	1.06	7.85	22 s	**0.00**	2.85	**6.87**	< 0.5 s
sphere_reg_nol_50x1	20.41	13.15	21.36	38.41	16 hours	60.08	60.08	59.74	< 0.5 s	0.24	**0.24**	35.71	22 s	**0.00**	2.69	**35.56**	< 0.5 s
sphere_rand_nol_100x1	23.43	13.11	26.97	51.37	16 hours	38.88	38.88	59.71	< 0.5 s	0.56	**0.56**	44.36	2 min	**0.00**	2.22	**44.04**	< 0.5 s
sphere_reg_nol_100x1	26.35	13.28	30.37	56.27	16 hours	74.31	74.31	71.09	< 0.5 s	3.21	3.21	51.17	2 min	**0.00**	**2.00**	**49.60**	< 0.5 s
sphere_rand_nol_500x1	51.35	12.24	71.03	82.77	16 hours	77.70	77.70	86.59	25 s	1.39	**1.39**	76.49	2 hours	**0.00**	1.70	**76.16**	9 s
						Overlaping regions											
sphere_rand_ol_10x1	7.56	**7.56**	**7.56**	**0.00**	10 s	62.17	62.17	38.34	< 0.5 s	61.08	61.08	37.92	< 0.5 s	**0.15**	**0.52**	**0.15**	< 0.5 s
sphere_cap_reg_ol_25x1	4.10	3.96	4.18	5.26	42 s	30.53	30.53	25.93	< 0.5 s	**0.00**	**0.00**	**3.31**	4 s	0.60	1.49	3.89	< 0.5 s
sphere_rand_ol_25x1	9.86	**9.86**	**9.86**	**0.00**	9 min	32.13	32.13	24.32	< 0.5 s	0.01	0.01	**0.01**	4 s	0.34	3.17	0.34	< 0.5 s
sphere_band_rand_ol_50x1	9.55	9.39	9.60	2.17	5 hours	129.30	129.30	57.14	< 0.5 s	**0.00**	**0.00**	**1.73**	21 s	1.02	2.32	2.72	< 0.5 s
sphere_rand_ol_50x1	12.95	11.82	13.47	12.25	16 hours	73.83	73.83	47.51	< 0.5 s	**0.00**	**0.00**	**8.76**	24 s	2.89	5.78	11.32	< 0.5 s
sphere_rand_ol_100x1	16.19	12.02	18.73	35.84	16 hours	92.71	92.71	61.49	< 0.5 s	5.79	5.79	29.85	2 min	**0.00**	**1.95**	**25.79**	< 0.5 s
Mean values						*56.87*	*56.87*	*35.77*	1 s	*3.99*	*3.99*	*13.68*	4 min	*0.53*	*2.41*	*11.54*	< 0.5 s

Table A.15: Performance indicators of GTSPN solvers in solution of 3D problems. Together with the perfomance indicators, the lower bound value $\mathcal{LB}$ and the upper bound value $\mathcal{UB}$ obtained by the proposed BNB solver are reported, whereas the gap (57) between the $\mathcal{LB}$ and $\mathcal{UB}$ is denoted as %GAP.

Problem	$\mathcal{L}_{ref}$	$\mathcal{LB}$	$\mathcal{UB}$	%GAP	T	HRGKA [12]				Centroid-GTSP+ [23]				**Proposed GSOA**			
						%PDB	%PDM	%GAP	T	%PDB	%PDM	%GAP	T	%PDB	%PDM	%GAP	T
3D_30_6_a	3 549.35	2 355.65	3 817.34	38.29	16 hours	0.33	0.33	33.85	52 s	**0.00**	**0.00**	**33.63**	< 0.5 s	3.30	5.41	35.75	< 0.5 s
3D_30_6_b	3 858.63	2 366.88	4 116.71	42.51	16 hours	0.54	0.54	38.99	51 s	**0.00**	**0.00**	**38.66**	< 0.5 s	1.05	4.02	39.30	< 0.5 s
3D_30_6_c	3 687.17	2 469.97	3 935.53	37.24	16 hours	0.74	0.74	33.50	55 s	**0.00**	**0.00**	**33.01**	< 0.5 s	0.58	2.39	33.40	< 0.5 s
3D_30_6_d	3 607.69	2 399.34	3 854.24	37.75	16 hours	**0.00**	**0.00**	**33.49**	40 s	0.11	0.11	33.56	< 0.5 s	1.17	2.99	34.26	< 0.5 s
3D_30_6_e	3 551.66	2 080.82	3 830.42	45.68	16 hours	1.17	1.17	42.09	52 s	**0.00**	**0.00**	**41.41**	< 0.5 s	0.44	2.88	41.67	< 0.5 s
3D_30_6_f	3 618.60	2 001.75	3 913.15	48.85	16 hours	0.18	0.18	44.78	53 s	**0.00**	**0.00**	**44.68**	< 0.5 s	1.30	2.91	45.39	< 0.5 s
3D_35_6_a	4 313.53	2 510.82	4 562.21	44.96	16 hours	0.12	0.12	41.86	1 min	**0.00**	**0.00**	**41.79**	< 0.5 s	1.96	4.00	42.91	< 0.5 s
3D_35_6_b	4 183.49	2 593.22	4 693.18	44.74	16 hours	0.74	0.74	38.47	52 s	**0.00**	**0.00**	**38.01**	< 0.5 s	0.89	4.99	38.56	< 0.5 s
3D_35_6_c	3 683.72	2 220.29	4 203.80	47.18	16 hours	0.45	0.45	40.00	1 min	**0.00**	**0.00**	**39.73**	< 0.5 s	3.29	7.21	41.65	< 0.5 s
3D_35_6_d	4 276.24	2 543.07	4 632.60	45.10	16 hours	0.63	0.63	40.90	1 min	**0.00**	**0.00**	**40.53**	< 0.5 s	1.22	1.88	41.25	< 0.5 s
3D_35_6_e	3 892.16	2 369.09	4 345.75	45.48	16 hours	0.18	0.18	39.24	1 min	**0.00**	**0.00**	**39.13**	< 0.5 s	0.62	6.67	39.51	< 0.5 s
3D_35_6_f	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*	NA*
3D_40_6_a	4 476.27	2 285.08	4 828.63	52.68	16 hours	0.29	0.29	49.10	56 s	**0.00**	**0.00**	**48.95**	< 0.5 s	1.17	2.80	49.54	< 0.5 s
3D_40_6_b	4 138.35	2 676.64	4 555.87	41.25	16 hours	0.92	0.92	35.91	1 min	**0.00**	**0.00**	**35.32**	< 0.5 s	0.78	2.33	35.82	< 0.5 s
3D_40_6_c	4 358.70	2 370.80	4 663.45	49.16	16 hours	0.48	0.48	45.87	1 min	**0.00**	**0.00**	**45.61**	< 0.5 s	2.26	4.44	46.81	< 0.5 s
3D_40_6_d	4 550.32	2 531.37	4 896.45	48.30	16 hours	**0.00**	**0.00**	**44.37**	1 min	0.07	0.07	44.41	< 0.5 s	1.77	4.40	45.34	< 0.5 s
3D_40_6_e	4 085.43	2 495.36	4 447.93	43.90	16 hours	0.80	0.80	39.41	1 min	**0.00**	**0.00**	**38.92**	< 0.5 s	3.48	7.91	40.97	< 0.5 s
3D_40_6_f	3 976.35	2 191.26	4 272.69	48.71	16 hours	1.52	1.52	45.72	1 min	**0.00**	**0.00**	**44.89**	< 0.5 s	2.59	3.54	46.29	< 0.5 s
3D_45_6_a	4 466.40	2 363.86	4 929.01	52.04	16 hours	0.79	0.79	47.49	1 min	**0.00**	**0.00**	**47.07**	< 0.5 s	2.15	4.04	48.19	< 0.5 s
3D_45_6_b	4 879.96	2 856.89	5 398.40	47.08	16 hours	0.82	0.82	41.93	1 min	**0.00**	**0.00**	**41.46**	< 0.5 s	1.00	5.56	42.04	< 0.5 s
3D_45_6_c	4 784.13	2 676.80	5 287.22	49.37	16 hours	0.63	0.63	44.40	1 min	**0.00**	**0.00**	**44.05**	< 0.5 s	1.88	3.91	45.08	< 0.5 s
3D_45_6_d	4 818.70	2 418.94	5 338.43	54.69	16 hours	1.20	1.20	50.40	1 min	**0.00**	**0.00**	**49.80**	< 0.5 s	2.96	4.98	51.24	< 0.5 s
3D_45_6_e	4 874.99	2 639.43	5 342.24	50.59	16 hours	1.32	1.32	46.56	1 min	**0.00**	**0.00**	**45.86**	< 0.5 s	0.72	4.64	46.25	< 0.5 s
3D_45_6_f	4 676.46	2 608.16	5 193.72	49.78	16 hours	1.68	1.68	45.15	1 min	**0.00**	**0.00**	**44.23**	< 0.5 s	2.80	5.14	45.75	< 0.5 s
3D_50_6_a	4 904.16	2 397.78	5 313.13	54.87	16 hours	1.40	1.40	51.78	1 min	**0.00**	**0.00**	**51.11**	< 0.5 s	1.37	4.75	51.77	< 0.5 s
3D_50_6_b	4 962.53	2 346.24	5 661.51	58.56	16 hours	2.36	2.36	53.81	1 min	**0.00**	**0.00**	**52.72**	< 0.5 s	1.06	2.98	53.22	< 0.5 s
3D_50_6_c	4 871.39	2 296.86	5 535.59	58.51	16 hours	1.40	1.40	53.50	1 min	**0.00**	**0.00**	**52.85**	< 0.5 s	1.87	5.30	53.72	< 0.5 s
3D_50_6_d	4 566.25	2 545.90	5 083.65	49.92	16 hours	0.06	0.06	44.28	1 min	**0.00**	**0.00**	**44.25**	< 0.5 s	2.61	3.69	45.66	< 0.5 s
3D_50_6_e	5 323.81	2 673.90	5 957.19	55.11	16 hours	0.83	0.83	50.19	1 min	**0.00**	**0.00**	**49.77**	< 0.5 s	3.06	5.37	51.26	< 0.5 s
3D_50_6_f	4 919.06	2 733.57	5 426.26	49.62	16 hours	1.63	1.63	45.32	1 min	**0.00**	**0.00**	**44.43**	< 0.5 s	3.17	4.25	46.14	< 0.5 s
Mean values						*0.80*	*0.80*	*43.53*	1 min	*0.01*	*0.01*	*43.10*	< 0.5 s	*1.81*	*4.32*	*44.09*	< 0.5 s

* NA - Results by the BNB have not been determined in predefined time

Table A.16: Performance indicators of GTSPN solvers in solution of 7D problems. Together with the perfomance indicators, the lower bound value $\mathcal{LB}$ and the upper bound value $\mathcal{UB}$ obtained by the proposed BNB solver are reported, whereas the gap (57) between the $\mathcal{LB}$ and $\mathcal{UB}$ is denoted as %GAP.

Problem	$\mathcal{L}_{ref}$	$\mathcal{LB}$	$\mathcal{UB}$	%GAP	T	HRGKA [12]				Centroid-GTSP+ [23]				**Proposed GSOA**			
						%PDB	%PDM	%GAP	T	%PDB	%PDM	%GAP	T	%PDB	%PDM	%GAP	T
7D_30_6_a	9 263.10	3 959.10	9 907.60	60.04	16 hours	**0.00**	**0.00**	**57.26**	1 min	3.33	3.33	58.64	< 0.5 s	26.03	29.38	66.09	< 0.5 s
7D_35_6_a	10 704.00	3 674.35	11 363.12	67.66	16 hours	**0.00**	**0.00**	**65.67**	1 min	3.95	3.95	66.98	< 0.5 s	26.08	28.94	72.77	< 0.5 s
7D_40_6_a	10 871.40	3 746.87	11 592.86	67.68	16 hours	**0.00**	**0.00**	**65.53**	1 min	3.82	3.82	66.80	< 0.5 s	33.72	36.94	74.22	< 0.5 s
7D_45_6_a	12 142.20	3 643.74	13 086.50	72.16	16 hours	**0.00**	**0.00**	**69.99**	2 min	2.22	2.22	70.64	< 0.5 s	33.52	38.19	77.52	< 0.5 s
7D_50_6_a	13 868.40	3 897.62	14 683.14	73.46	16 hours	**0.00**	**0.00**	**71.90**	2 min	2.21	2.21	72.50	< 0.5 s	37.63	41.60	79.58	< 0.5 s
Mean values						*0.00*	*0.00*	*66.07*	2 min	*3.11*	*3.11*	*67.11*	< 0.5 s	*31.40*	*35.01*	*74.04*	< 0.5 s

Appendix B
Content of the Enclosed CD

```
CD
 ├── README.md
 ├── bnb.tspn.zip
 ├── gsoa.3d.zip
 ├── gsoa.tspns.zip
 └── gsoa.gtspn.zip
```