CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering
Department of Computer Science

Bachelor's thesis

Heuristic Solution of the Close Enough Orienteering Problem

Petra Štefaníková

May 2020
Supervisor: prof. Ing. Jan Faigl, Ph.D.

“No temptation has overtaken you except what is common to mankind. And God is faithful; he will not let you be tempted beyond what you can bear. But when you are tempted, he will also provide a way out so that you can endure it.”

– 1 Co. 10,13

“Nepotkala vás zkouška nad lidské síly. Bůh je věrný: nedopustí, abyste byli podrobeni zkoušce, kterou byste nemohli vydržet, nýbrž se zkouškou vám připraví i východisko a dá vám sílu, abyste mohli obstát.”

– 1K 10,13

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Štefaníková** Jméno: **Petra** Osobní číslo: **474384**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Otevřená informatika**

Studijní obor: **Software**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Heuristická řešení směrovacích problémů s profitem a diskovým okolím

Název bakalářské práce anglicky:

Heuristic Solution of the Close Enough Orienteering Problem

Pokyny pro vypracování:

- Seznamte se s problematikou Orienteering Problem (OP) [1] existujícími algoritmy a způsoby řešení rozšířené varianty s diskovým okolím - Close Enough OP (CEOP) [2].
- Navrhněte rozšíření GRASP [3] heuristiky pro řešení CEOP.
- Implementovaná rozšíření porovnejte s existujícími přístupy řešení [4, 5, 6].

Seznam doporučené literatury:

[1] Gunawan, A., Lau, H.C., and Vansteenwegen, P.: Orienteering Problem: A survey of recent variants, solution approaches and applications, European Journal of Operational Research, 255(2):315-332, 2016.

[2] Best, G., Faigl, J., and Fitch, R.: Online planning for multi-robot active perception with self-organising maps, Autonomous Robots, 42(4):715-738, 2018.

[3] Keshtkaran, M. and Koorush, Z.: A novel GRASP solution approach for the Orienteering Problem, Journal of Heuristics 22:699-726, 2016.

[4] Faigl, J., Pěnička, R., and Best, G.: Self-organizing map-based solution for the Orienteering problem with neighborhoods, IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2016, pp. 1315-1321.

[5] Faigl, J.: Data collection path planning with spatially correlated measurements using growing self-organizing array, Applied Soft Computing, 75:130-147, 2019.

[6] Pěnička, R., Faigl, J., and Saska, M.: Variable Neighborhood Search for the Set Orienteering Problem and its application to other Orienteering Problem variants, European Journal of Operational Research, 276(3):816-825, 2019.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

doc. Ing. Jan Faigl, Ph.D-, ČVUT FEL, Centrum umělé inteligence

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **03.02.2020** Termín odevzdání bakalářské práce: **22.05.2020**

Platnost zadání bakalářské práce: **30.09.2021**

doc. Ing. Jan Faigl, Ph.D- podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Studentka bere na vědomí, že je povinna vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací.
Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

.

Datum převzetí zadání

Podpis studentky

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 ethiclal principles in the preparation of university theses.

Prague, May 22, 2020

. . . .
Petra Štefaníková

Acknowledgement

I would like to thank my supervisor, prof. Ing. Jan Faigl, Ph.D. for the enthusiasm for the work which helped me to keep going; for sacrificing an unexpected amount of time to explain, clarify and solve different kinds of problems; and also for the "but it's only my opinion". I would also like to thank Ing. Petr Váňa who has taught me the difference between tour and journey and made me realize the time pressure. Many thanks to my fiancé for eating all the baked bread and donuts I made instead of writing the thesis and my parents for full support in whatever I do. And thanks to all my friends, especially Kristýna Kučerová, who have shared the struggles with me.

Abstract

In this thesis, the combinatorial meta-heuristic Greedy Randomized Adaptive Search Procedure (GRASP) with Segment Remove is extended to solve the Close Enough Orienteering Problem (CEOP). The addressed problem stands to find the most rewarding path visiting a set of disk-shaped regions such that the path does not exceed the given travel budget. The CEOP includes a discrete combinatorial problem to determine a subset of regions and its sequence of visits together with a continuous optimization problem to determine the optimal waypoint location of each visit to the regions. Three new heuristics are proposed to improve the performance of the GRASP algorithm. All the GRASP-based approaches have been evaluated on existing benchmarks and compared with two existing methods to the CEOP.

Keywords: Close Enough Orienteering Problem, Greedy Randomized Adaptive Search Procedure, Routing problem with profit and with neighborhood, Selective Traveling Salesman Problem

Abstrakt

V této práci je rozšířena kombinatorická meta-heuristika Greedy Randomize Adaptive Search Procedure (GRASP) pro řešení úloh Close Enough Orienteering Problem (CEOP). V této úloze je cílem nalézt cestu maximalizující profit navštívením diskových regionů, která zároveň není delší než dané omezení. CEOP kombinuje diskrétní kombinatorický problém určení podmnožiny regionů a jejich pořadí navštívení a spojitý optimalizační problém nalezení optimálních míst navštívení regionů. V práci jsou navrženy tři nové heuristiky zlepšující řešení úlohy CEOP metodou GRASP. Všechny přístupy byly empiricky vyhodnoceny na existujících datasetech a porovnány s existujícími metodami řešení CEOP.

Klíčová slova: Close Enough Orienteering Problem, Greedy Randomized Adaptive Search Procedure, Směrovací problém s profitem a okolím, Selektivní problém obchodního cestujícího

Used Abbreviations

CEOP	Close Enough Orienteering Problem
CP	Construction Phase
CPLEX	IBM ILOG CPLEX Optimization Studio
CL	Candidate List
GRASP	Greedy Randomized Adaptive Search Procedure
GRASP-SR	Greedy Randomized Adaptive Search Procedure with the Segment Remove
GSOA	Growing Self-Organizing Array
HOP	Heuristic of the Ordered Placing
ILP	Integer Linear Programming
LIO	Local Iterative Optimization
LSP	Local Search Phase
OP	Orienteering Problem
OPN	Orienteering Problem with Neighborhoods
RVNS	Randomized Variable Neighborhood Search
SOCP	Second-Order Cone Program
SR	Segment Remove
TS	Tabu Search
TSP	Traveling Salesmen Problem
VNS	Variable Neighborhood Search

Used Symbols

$\|\boldsymbol{p}_1, \boldsymbol{p}_2\|$	Euclidean distance between locations $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$
n	Number of locations
$\boldsymbol{v}_i$	i-th location
$\mathrm{T}_{\max}$	Travel budget
ϱ	Sensing radius
k	Number of locations in a path
Σ	Permutation of location indexes
P	Path as a sequence of waypoint locations
$\sigma_1, \ldots, \sigma_k$	Location indexes of the path, $1 \leq \sigma_i \leq n$
$\boldsymbol{p}_1, \ldots, \boldsymbol{p}_k$	Waypoint locations
$\mathcal{L}(\Sigma)$	Length of OP path determined by Σ
$\mathcal{L}(\Sigma, P)$	Length of CEOP path determined by Σ and P
$R(\Sigma)$ or R	Total sum of rewards collected by visiting locations Σ
R_{best}	Best found total rewards within the Candidate List
c_{best}	Restriction parameter for the Candidate List
G	Gap performance indicator
$\overline{G}$	Average gap performance indicator
R_{ref}	Best found solution over all found solutions of a single problem instance

Contents

List of Figures

List of Tables

List of Algorithms

CHAPTER 1

Introduction

Let us imagine a vehicle requested to collect data from a given set of sites which may represent remote sensing or wireless communication with sensors. In practice, the operating time of the vehicle can be limited, e.g., by limited flight time of a small aerial vehicle due to battery payload. Therefore, the vehicle can only collect data from a subset of the sites within its limited travel budget. Hence, assigning a reward to each site is a suitable option to prioritize the more important locations. Then the problem is to determine the most rewarding tour that does not exceed the given travel budget. Such a problem can be found in the orienteering sport game where competitors run from the start location, navigate through a number of control locations using a map and a compass and the goal is to arrive at the final destination. In a type of orienteering, the contestants are required to arrive at the final destination within a fixed time limit. For that reason, they only have to choose a subset of the control points and arrive on time, or they are penalized or disqualified. Such a problem corresponds to the motivational data collection planning that has been introduced in the literature as Orienteering Problem (OP).

The OP derived from the orienteering races was firstly introduced by Golden, Levy, and Vohra in [1]. The OP stands to find a tour from the initial location, visiting the most rewarding locations and terminating in the final location so that the tour does not exceed the given travel budget. The OP can be considered as a combination of two well-known combinatorial problems. The Knapsack problem [2] where we have to determine a subset of locations satisfying the given constraint and the Traveling Salesman Problem (TSP) [3] where the goal is to find the shortest closed tour visiting all the given sites.

The regular OP has been generalized to the OP with Neighborhood (OPN) [4]. In the OP, the vehicle has to visit the location at the exact position. In contrast, in the OPN, the vehicle may approach the area surrounding the location to collect the reward. Thus, the travel cost can be saved, and the total reward can be increased by using the travel budget for visiting more locations. The OPN with a disk-shaped sensing area is called the Close Enough Orienteering Problem (CEOP) to emphasize the restricted shape of the sensing area to a disk. Thus, the rewards might be collected from any point closer to the center of the region than the particular radius of the area.

Several solvers have been proposed to address the OP [5],[6] including optimal branch-and-cut algorithm [7], heuristic approaches such as S-Algorithm or D-Algorithm [8], and combinatorial meta-heuristics the Variable Neighborhood Search (VNS) [9] and Greedy Randomized Adaptive Search Procedure (GRASP) [10], but also unsupervised learning-based approaches based on the Hopfield Neural Networks [11] and Self-Organizing Map (SOM) [12]. However, the CEOP has only been addressed by two algorithms so far. The first, Growing Self-Organizing Array (GSOA) [13], is fast but unable to escape local optima. The second is based on the VNS combinatorial meta-heuristic [14] that utilizes sampling of the neighborhood into a finite set of locations. The VNS-based approach provides better results than the GSOA, but it is reported to be more computationally demanding.

Motivated by the VNS-based solution to the CEOP and reported results on the GRASP-based solution to the OP, the GRASP has been considered as a suitable approach to be extended for solving the CEOP. We aim to address the drawbacks of the two existing approaches and provide solutions with competitive quality to the VNS-based method but with the computational requirements of the GSOA. In practice, we propose three heuristics for determining the waypoint location within the disk-shaped sensing area. The first straightforward heuristic [15] is based on determining the waypoint location as the closest point of the disk to the path segment. The second approach uses Local Iterative Optimization (LIO) [16] to set the waypoint location by a continuous local descent. Lastly, the optimal solver CPLEX [17] is employed to determine the waypoint location as the Second-Order Cone Program (SOCP), which is an optimization problem with quadratic constraints. Since the LIO and SOCP are computationally demanding, we propose the Heuristic of the Ordered Placing (HOP) to prevent excessive waypoint location determination. All these approaches are compared to both existing algorithms, the GSOA and VNS.

The thesis is organized as follows. The OP is formally defined in the following chapter. Then the algorithms for the OP and CEOP are described in Chapter 3. The proposed GRASP based approach for the CEOP is introduced in Chapter 4 and the results are presented in Chapter 5. The whole work is summarized in Chapter 6.

CHAPTER 2

Problem Statement

The addressed variant of the Orienteering Problem (OP) is motivated by data collection missions where a vehicle collects data by visiting particular locations. In the Close Enough Orienteering Problem (CEOP), the data can be collected from the disk-shaped sensing area of the location. Thus, the vehicle can visit the region at any point of the disk with the radius ϱ centered at the location $\boldsymbol{v}_i$. For clarity, all the locations are considered to be $\boldsymbol{v}_i \in \mathbb{R}^2$.

2.1 Orienteering Problem

Let $V = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ be a set of n locations $\boldsymbol{v_i} \in \mathbb{R}^2$ with Euclidean distance between the locations $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ denoted $\|\boldsymbol{v}_i, \boldsymbol{v}_j\|$. Each location $\boldsymbol{v}_i$ has assigned positive reward $r_i \in \mathbb{R}^+$ but the initial location $\boldsymbol{v}_1$ and final location $\boldsymbol{v}_n$ have zero reward, and thus $r_1 = r_n = 0$. The OP stands to plan the most rewarding route from $\boldsymbol{v}_1$ to $\boldsymbol{v}_n$ such that the length does not exceed the travel budget $\mathrm{T}_{\max}$. Hence, in the OP, it has to be determined a subset $S_k \subseteq V$ of $2 \leq k \leq n$ locations and the optimal sequence of their visits to ensure $\mathrm{T}_{\max}$ is satisfied. The solution can be described as a sequence of k locations indexes $\Sigma = \langle \sigma_1, \ldots, \sigma_k \rangle$ where $\sigma_1 = 1$ and $\sigma_k = n$ with the length

$$\mathcal{L}(\Sigma) = \sum_{i=2}^{k} \left\| \boldsymbol{v}_{\sigma_{i-1}}, \boldsymbol{v}_{\sigma_i} \right\|. \tag{2.1}$$

The OP can be formulated as an optimization Problem 2.1 where the total collected reward $R(\Sigma)$ is being maximized such that the length of the path $\mathcal{L}(\Sigma) \leq \mathrm{T}_{\max}$. The OP is known to be NP-hard [1].

Problem 2.1 Orienteering Problem (OP).

$$\max_{k,\Sigma} \quad R(\Sigma) = \sum_{i=1}^{k} r_{\sigma_i}$$

$$\begin{aligned}
&\text{s. t.}\\
&\quad \mathcal{L}(\Sigma) \leq \mathrm{T}_{\max}\\
&\quad 2 \leq k \leq n\\
&\quad \Sigma = \langle \sigma_1, \ldots, \sigma_k \rangle\,, \quad 1 \leq \sigma_i \leq n,\ \sigma_i \neq \sigma_j \text{ for } i \neq j\\
&\quad \sigma_1 = 1\ ,\ \sigma_k = n
\end{aligned}$$

2.2 Close Enough Orienteering Problem

The CEOP generalizes the OP in the way that the vehicle does not have to reach the precise location $\boldsymbol{v}_i$ to get the data. The reward can be collected from any point $\boldsymbol{p}_i$ that is within the ϱ distance from $\boldsymbol{v}_i$ where ϱ is the radius of the disk-shaped sensing area centered at $\boldsymbol{v}_i$. In the CEOP, we thus need to determine not only the subset of k locations and the sequence $\Sigma = \langle \sigma_1, \ldots, \sigma_k \rangle$ but also the best waypoint locations $P = \langle \boldsymbol{p}_1, \ldots, \boldsymbol{p}_k \rangle$ in the disk area of each location such that $\|\boldsymbol{v}_{\sigma_i}, \boldsymbol{p}_i\| \leq \varrho$. Hence, the tour length can be expressed as

$$\mathcal{L}(\Sigma, P) = \sum_{i=2}^{k} \|\boldsymbol{p}_{i-1}, \boldsymbol{p}_i\|\,. \tag{2.2}$$

The initial and final locations $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$ are prescribed as in the OP, and thus, $\boldsymbol{p}_1 = \boldsymbol{v}_1$, $\boldsymbol{p}_k = \boldsymbol{v}_n$ because the vehicle has to start from the specific location and might return to a different one.

The CEOP can be formulated as the optimization Problem 2.2, which includes a continuous optimization of the waypoint locations $P = \langle \boldsymbol{p}_1, \ldots, \boldsymbol{p}_k \rangle$ up to $2n$ variables in addition to the OP combinatorial part, i.e., determining the subset of k sites and the sequence of their visits. Notice, the CEOP is also NP-hard as for the $\varrho = 0$, it reduces to the OP.

Problem 2.2 Close Enough Orienteering Problem (CEOP).

$$\max_{k,\Sigma,P} \quad R(\Sigma) = \sum_{i=1}^{k} r_{\sigma_i}$$

$$\begin{aligned}
&\text{s. t.}\\
&\quad \mathcal{L}(\Sigma, P) \leq \mathrm{T}_{\max}\\
&\quad 2 \leq k \leq n\\
&\quad \Sigma = \langle \sigma_1, \ldots, \sigma_k \rangle\,, \quad 1 \leq \sigma_i \leq n,\ \sigma_i \neq \sigma_j \text{ for } i \neq j\\
&\quad P = \langle \boldsymbol{p}_1, \ldots, \boldsymbol{p}_k \rangle\\
&\quad \sigma_1 = 1\ ,\ \sigma_k = n\ , \quad \boldsymbol{p}_1 = \boldsymbol{v}_1\ ,\ \boldsymbol{p}_k = \boldsymbol{v}_n\\
&\quad \|\boldsymbol{p}_i, \boldsymbol{v}_{\sigma_i}\| \leq \varrho\,, \quad \text{for } \boldsymbol{p}_i \in P \text{ and } \boldsymbol{v}_{\sigma_i} \in V
\end{aligned}$$

CHAPTER 3

Related Work

Since the first work solving the OP was introduced in the late 20th century, many approaches were proposed and later improved over time. Firstly, optimal solvers based on branch-and-bound [18] or branch-and-cut [7] approaches were introduced, which formulates the OP as the Integer Linear Programming (ILP). The approaches use Lagrangian relaxation of the ILP to determine the lower and upper bounds and thus reduce the rooted search tree of solutions. Since the OP is NP-hard, the optimal approaches are very demanding and computationally intractable for an increasing number of locations. Therefore, methods based on meta-heuristics or unsupervised learning have been proposed, which balance the computational time with the quality of the provided results.

The first proposed heuristic approaches to the OP are the S-Algorithm and D-Algorithm by Tsiligirides [8]. The stochastic S-Algorithm generates many routes using the Monte Carlo search method. The D-Algorithm is a deterministic algorithm based on the idea of Wren and Holliday for the vehicle-scheduling problem [19]. The routes are built up in specific separated sectors of the location space, which minimize the total path length by predetermined rules. The modifications of the sectors obtain 48 possibilities for each iteration that are further evaluated. Besides, a route improvement algorithm has been proposed to improve further found solutions by exchange, insertion and exclusion locations in the route.

Further, Golden et al. introduced the Center of Gravity heuristic [1] containing three main steps: 1) the route construction step; 2) the route improvement step; and 3) the center-of-gravity step. The first two steps construct the initial path, improve it by the 2-Opt [20], which reduces the crossed segments of the path, and insert the closest locations within the travel budget. The initial tour is further adjusted according to the center-of-gravity, where the assigned reward represents the location weight. Thus, the remaining locations are inserted into the path based on the rewards and the distance to the center-of-gravity. The center-of-gravity and the improvement steps repeat until the path changes. The final solution is selected as the best path from all produced ones.

The Four-phase algorithm [21] was presented by Ramesh and Brown. In the first insertion phase, a location is selected based on the ratio between the minimal prolongation and its reward. The second phase applies the 2-Opt [20] or 3-Opt [22] optimization for the TSP based on the quality of the solution. The method repeats the first phase when the provided route might obtain more locations without exceeding the travel budget. In the third phase, the algorithm deletes specific locations due to the possibility of a new insertion. The three phases are iteratively repeated to generate the solution. In the final phase, the remaining locations are attempted to be added into the final path based only on their prizes such that they do not violate the travel budget.

The Tabu Search (TS) [23] has been employed to the OP in [24], where operations over clusters of locations are proposed. It tries to avoid including location with seemingly high profit, but the distance between the current path and the location prevents the insertion of other closer sites, which may have higher total rewards. Hence, the algorithm is more likely to escape the local optima. The idea of the TS insertion is to determine the candidate cluster based on the dispersion index. Then, from the cluster, all unreachable sites are removed and included in the current route using the TSP heuristic US [25]. The locations are also being removed according to the length of the edges between locations. The removed sites receive a tabu status for several iterations which restrict their insertion.

The Ant Colony meta-heuristic [26] has been deployed to the OP in [27], where agents called ants leave their pheromone scent on visited edges between the sites. A state transition rule is implemented based on the left pheromone and the rate between the location profit and its distance to balance the solution space exploration with the path improvement.

The Variable Neighborhood Search (VNS) meta-heuristic [28] has been utilized for the OP in [29] to systematically search through the solution space using predetermined neighborhood structures. The VNS consists of two main procedures Shake and Local Search, which try to improve the generated initial solution. The Shake procedure chooses the next route from a set neighborhood structure, and the path is further improved in the Local Search procedure. If the newly produced solution has a better sum of rewards than the current one, the path is used in the next iteration. When the route does not improve, the algorithm continues with a different neighborhood structure. The method performance is highly connected to the demands of the neighborhood structures, and for the OP, four of them were proposed. The first Insert specifies the closest solution neighbors obtained by a single location insertion, deletion, or a location reordering. The second structure is the Exchange, which swaps two sites, not depending on whether the locations are included in the route or not. The next structure is the Path Insert, which is similar to the simple Insert, but sub-paths are inserted or deleted. The fourth structure is the Path Exchange that swaps two sub-paths of the same number of sites. Several variants of VNS were proposed in [30] to address the computational requirements of the systematical search. Such modifications are the Variable Neighborhood Descent, which uses only the Local Search phase, the Reduced VNS that runs only the Shake part, and Randomized VNS (RVNS), which randomly chooses the next neighborhood structure instead of systematical search.

The last herein reviewed meta-heuristic approach is based on the Greedy Randomized Adaptive Search Procedure (GRASP) [31] that is a highly effective, constructive meta-heuristic used for different types of optimization problems. In 2014, Campos et al. presented GRASP with Path Relinking [10] and applied it to the OP. It provides high-quality solutions in a reasonable time compared to the state-of-the-art algorithms. Combining the GRASP approach and local combinatorial optimization is further employed in the novel GRASP with Segment Remove (GRASP-SR) [32], which further outperformed the existing approaches.

In contrast to the combinatorial heuristic approaches, there are also approaches based on unsupervised learning. Wang et al. proposed the Hopfield Neural Network to solve the OP [11] using a path represented as a two-dimensional matrix. The matrix cell (i, j) represents the i-th region at the j-th position in the path. The goal is to minimize the energy function that takes into consideration the OP constraints as the visitation of each location no more than once, the initial and final locations, and not exceeding the travel budget.

The addressed CEOP is highly connected with the solution of the OP because the existing methods for the CEOP are extensions of the existing OP solvers. Therefore, not only the existing methods to the CEOP are detailed in the following sections to make the thesis more self-contained, but also the perspective GRASP-based solution to the OP is detailed to provide insights to the developed solutions. In particular, the Growing-Self Organizing Array (GSOA) [13] is described in Section 3.1 and the VNS-based approach [14] in Section 3.2. Finally, the herein proposed approach is based on GRASP, and therefore, it is detailed in Section 3.3.

3.1 Growing Self-Organizing Array (GSOA)

The Growing Self-Organizing Array (GSOA) [13] is an unsupervised learning technique to address routing problems. It is developed from the Self-Organizing Map (SOM) for the TSP [33], later generalized to the OPN in [34]. The GSOA is an array of nodes where each node has its position, and it is associated with the region, and the particular waypoint location visiting the region. Since nodes are organized in an array, the GSOA naturally encodes the path as a sequence of regions with their waypoint locations. The learning of the GSOA is an iterative adaptation of the array to the locations in a fixed number of learning epochs. During each learning epoch, a new node might be created for each region together with the corresponding waypoint location as the closest point of the node to the region. The new node is kept if the path length does not exceed the travel budget. Otherwise, the node is not added to the array and continues with the next site. At the end of each learning epoch, the GSOA only keeps the new nodes, and nodes from the previous epoch are removed. The GSOA can be considered as a constructive heuristic because it provides a solution very quickly (in several epochs); however, once the array converges to a stable solution, it cannot escape the local optima.

3.2 Variable Neighborhood Search

The VNS-based combinatorial meta-heuristic has been employed to the CEOP in its RVNS variant in [14]. Since the solution of the CEOP needs to determine the waypoint locations of visits to the regions, the sensing areas are sampled into a finite set of locations. Then, the problem can be solved as a poorly discrete combinatorial optimization. However, the approach has been further extended in [35] to use the Local Iterative Optimization (LIO) that continuously determines locally optimal waypoint locations. The VNS-based solver is reported to provide high-quality solutions, but it is computationally demanding.

3.3 Greedy Randomize Adaptive Search Procedure

The GRASP algorithm consists of two main parts. The first is the Construction Phase (CP) that is applied to build the initial path by inserting new locations to the current solution, which at the beginning contains the initial and final locations. Then, the route is improved in the Local Search Phase (LSP) that might remove sites from the solution, uses 2-Opt [20] to shorten the path and attempts new insertions.

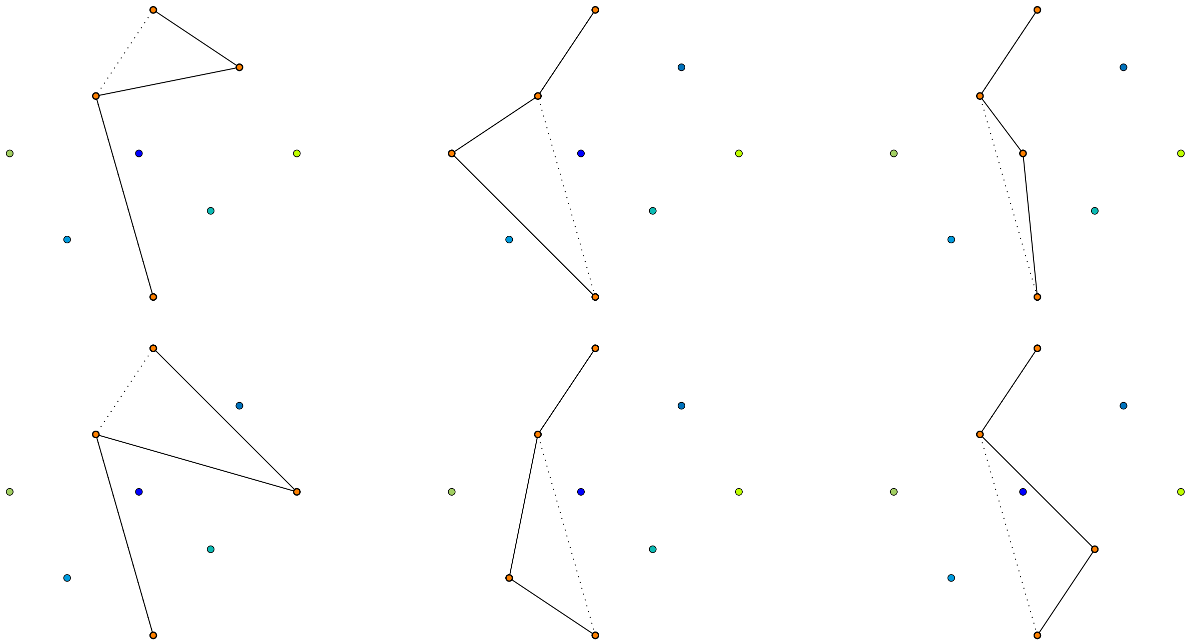
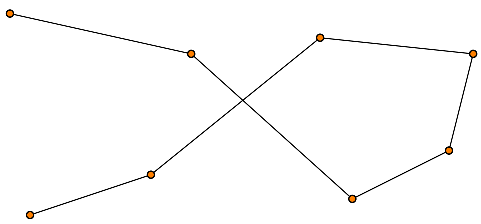
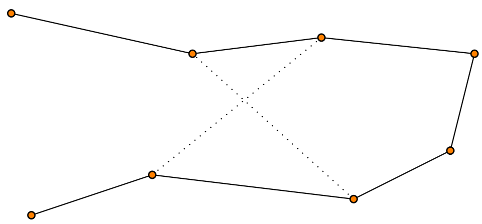


Figure 3.1: An example of the Candidate List before restriction based on the greedy insertion of unvisited locations, which are added at a position of the shortest path prolongation. The dotted line visualizes the path before the insertion.

At the beginning of each iteration of the CP, an unvisited location is greedily inserted into the current path such that the route prolongation is minimal. If the newly created path does not exceed the travel budget, this path is considered as a candidate for the next CP iteration. The path is stored in the Candidate List (CL) containing promising paths. Nevertheless, when the budget is not met, the path cannot be included in the CL. Hence, the part named Segment Remove (SR) tries to remove sub-paths to examine whether the inserted location is more beneficial to the total rewards. When the value of the route increases, it is also added into the CL. The SR continues until all meaningful segments are examined, and every improving solution is marked as a candidate. At the end of a single CP iteration, the CL is restricted in the way that each solution with at least 20 % of the best found total rewards are kept. Notice that the threshold value of 20 % is recommended in [10] based on the empirical evaluation. Finally, the ongoing path is randomly chosen from the restricted CL. The whole CP phase terminates when the CL is empty, which means no improving path has been found. An example of the CL before the restriction is depicted in Figure 3.1.



(a) The old path



(b) The improved path

Figure 3.2: An example a path improved by 2-Opt optimization heuristic.

After the CP, the solution is further improved in the LSP. Firstly, a single location is removed from the current path to escape the local extreme. Subsequently, the GRASP utilizes the 2-Opt heuristic [20] to eliminate crossed segments of the path, which may shorten the route, see Figure 3.2. At the end of a single iteration of the LSP, unvisited locations are attempted to be inserted in the same way as in the CP. If the current path improves the total collected rewards or has the same total rewards and the path length is shortened, it becomes the new current path for the next iteration. The whole GRASP algorithm terminates when no improving path is found in the LSP.

CHAPTER 4

Extension of GRASP to the Close Enough Orienteering Problem

For this thesis, the GRASP combinatorial meta-heuristic has been chosen to address the drawbacks of the existing VNS and GSOA-based approaches for the CEOP. The VNS-based algorithm provides high-quality solutions, but its computational employment is very demanding. Unlike VNS, the GSOA produces results quickly; however, it is unable to escape the local optima once it converges to a stable solution. The GRASP-SR approach for the OP managed to outperform the state-of-the-art algorithms in the quality of solutions where it repeatedly tries to escape the local optima. Furthermore, it is reported that its computational requirements are low.

Algorithm 1: GRASP

Input: $V = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ – n locations to be visited, each $\boldsymbol{v}_i \in V$ with the reward r_i, where $\boldsymbol{v}_1$ and $\boldsymbol{v}_n$ are the specified initial and final locations, respectively.

Output: (P, Σ) – Final path from $\boldsymbol{v}_1$ to $\boldsymbol{v}_n$ with the waypoint locations P and the sequence of visits Σ to the subset of V.

```
Σ ← ⟨1, n⟩;                                  // Initial path
P ← ⟨v_1, v_n⟩
repeat                                        // Construction Phase
    (P, Σ) ← addLocation(P, Σ);
until P is changed
(P, Σ) ← localSearch(P, Σ)                    // Local Search Phase
return (P, Σ)
```

The main difference between the GRASP-based approach for the OP and the CEOP is in determining the waypoint locations in the disk-shaped sensing areas, which may shorten the total path length. Thus, it might enable the addition of new locations, and the total sum of rewards may increase. Therefore, the GRASP for the CEOP is mainly about determining the waypoint locations $\boldsymbol{p}_j$ for $j \in \{1, \ldots, k\}$ while locations $\boldsymbol{v}_i$ are being inserted by the operation `addLocation`. However, once the waypoint location is set, it is not further improved. Thus, in the `localSearch` procedure, an `optimization` operator is implemented to improve the waypoint locations of the whole path. The GRASP is overviewed in Algorithm 1 and the proposed modifications in `addLocation` and `localSearch` are detailed in the rest of the chapter.

In a single iteration of the Construction Phase (CP) depicted in Algorithm 2, an unvisited region centered in $\boldsymbol{v}_i$ is greedily inserted into the current path by the `insertion` operator defined in (4.1).

Definition 4.1 `insertion`$(P, \Sigma, j, \boldsymbol{v}_i)$**.**

$$\begin{aligned} \Sigma^* &= \langle \sigma_1, \ldots, \sigma_{j-1}, i, \sigma_j, \ldots, \sigma_{|\Sigma|} \rangle \\ P^* &= \left\langle \boldsymbol{p}_{\sigma_1}, \ldots, \boldsymbol{p}_{\sigma_{j-1}}, \boldsymbol{p}^*, \boldsymbol{p}_{\sigma_j}, \ldots, \boldsymbol{p}_{\sigma_{|\Sigma|}} \right\rangle \end{aligned} \tag{4.1}$$

The `insertion` puts the region of $\boldsymbol{v}_i$ at the position j' defined in (4.2) such that the prolongation of the path (P^*, Σ^*) is minimal.

$$j' = \underset{j=2}{\overset{|\Sigma|}{\arg\min}} \|\mathcal{L}(P^*, \Sigma^*) - \mathcal{L}(P, \Sigma)\| \tag{4.2}$$

For the OP the waypoint location $\boldsymbol{p}^*$ is directly the location $\boldsymbol{v}_i$. In the case of CEOP, it has to be determined as the most suitable location of the region. Three approaches detailed in Section 4.1–4.3 are proposed. Afterward, the newly created path (P', Σ') is determined whether the path length is within the travel budget $\mathrm{T}_{\max}$ and it is decided to include the route in the Candidate List (CL) or not.

For the current insertion, if the travel budget is exceeded, the Segment Remove (SR) is employed to remove different sub-paths named segments to test whether the newly added location is more valuable to the total reward. In practice, the SR can be implemented very effectively with a linear time complexity to the number of sites in the current path $\mathcal{O}(k)$. It starts by excluding a single location after the initial location such that two segments of the tour are removed. Then, the path length is checked, whether it reached the travel budget. If the travel budget is satisfied and the sum of the rewards of the new route (P'', Σ'') improves compared to the current path (P', Σ'), the route is included in the CL. Then, the first location of the removed sub-path is returned to the solution. When the travel budget is exceeded, the segments are extended with the following one. The whole cycle repeats until the end of the path is not reached. An example of the SR is depicted in Figure 4.1.

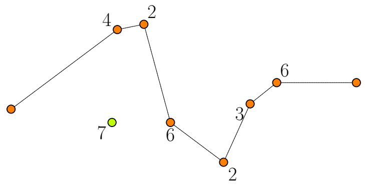
At the end of the `addLocation` procedure, the final candidate list CL' is created by `restrict` operation (4.3) which keeps only solutions with a greater total reward then c_{best} of the highest found rewards R_{best}. According to [10], the c_{best} is set to 20 %, which removes only the least promising routes and maintains the possibility of escaping the local optima.

Definition 4.2 `restrict`(CL)**.**

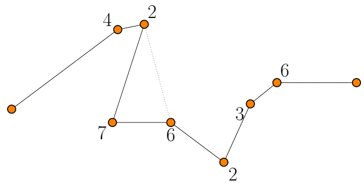
$$\begin{aligned} R_{\text{best}} &= \max \{R(\Sigma) \mid (P, \Sigma) \in CL\} \\ CL' &= \{(P, \Sigma) \mid (P, \Sigma) \in CL, R(\Sigma) \geq c_{\text{best}} R_{\text{best}}\} \end{aligned} \tag{4.3}$$

Finally, the successor to the current path is randomly chosen from the restricted CL'. The CP continues while an improved route is found, which means that the CL is not empty. The time complexity of a single CP iteration depicted in Algorithm 2 can be bounded by $\mathcal{O}(n^2)$.

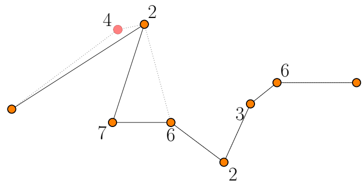
```
Algorithm 2: addLocation(P, Σ, b = −1)
Input: P, Σ – the current path.
Input: b – blocked index; if not specified b = −1 is used.
Output: P̄, Σ̄ – the updated path.

CL ← ∅                                              // A candidate list of solutions
for i ∈ {1, ..., |V|} do
    if i ∉ Σ ∧ i ≠ b then
        for j ∈ {2, ..., |Σ|} do
            (P*, Σ*) ← insertion(P, Σ, j, v_i)              // Using (4.1)
            if (j = 2) ∨ (ℒ(P*, Σ*) < ℒ(P′, Σ′)) then
                (P′, Σ′) ← (P*, Σ*)
            end
        end
        if ℒ(P′, Σ′) < T_max then
            CL ← CL ∪ {(P′, Σ′)}
        end
        else                                          // Segment Remove
            β ← 2
            for α ∈ {2, ..., |Σ′| − 1} do
                if β < α then
                    β ← α
                end
                Σ″ ← Σ′ \ {σ_α, ..., σ_β}
                P″ ← P′ \ {p_σα, ..., p_σβ}
                while (β + 1 < |Σ|) ∧ (ℒ(P″, Σ″) > T_max) do
                    β ← β + 1
                    Σ″ ← Σ′ \ {σ_α, ..., σ_β}
                    P″ ← P′ \ {p_σα, ..., p_σβ}
                end
                if ℒ(P′, Σ′) ≤ T_max then
                    if (R(Σ″) > R(Σ)) ∨ (R(Σ″) = R(Σ) ∧ ℒ(P″, Σ″) < ℒ(P, Σ)) then
                        CL ← CL ∪ {(P″, Σ″)}
                    end
                end
            end
        end
    end
end
if CL ≠ ∅ then
    restrict(CL)                                      // Using (4.3)
    (P̄, Σ̄) ← random(CL)
end
return (P̄, Σ̄)
```



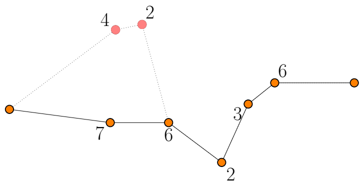
(a) The current path with $\mathcal{L}(P, \Sigma) \leq \mathrm{T}_{\max}$ and $R(\Sigma) = 23$ before insertion of the new green location.



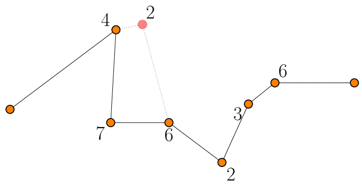
(b) A greedy insertion of the new location but with the exceeded travel budget $\mathcal{L}(P, \Sigma) > \mathrm{T}_{\max}$.



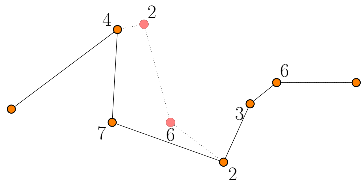
(c) A removal of a single location at the beginning of the path still with $\mathcal{L}(P, \Sigma) > \mathrm{T}_{\max}$.



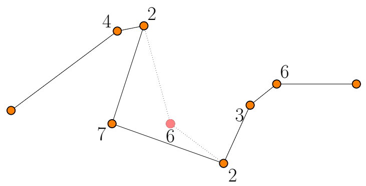
(d) An expansion of the removed segment with $\mathcal{L}(P, \Sigma) \leq \mathrm{T}_{\max}$ and $R(\Sigma) = 24$. This path is included into the CL.



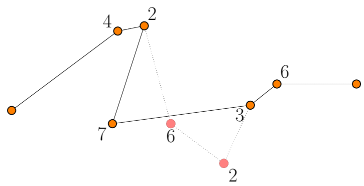
(e) A shortening of the segment but with exceeded travel budget $\mathcal{L}(P, \Sigma) > \mathrm{T}_{\max}$ again.



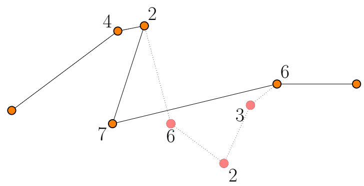
(f) An expansion of the segment to the following location with $\mathcal{L}(P, \Sigma) \leq \mathrm{T}_{\max}$ and $R(\Sigma) = 22$. This path is not better then the current path.



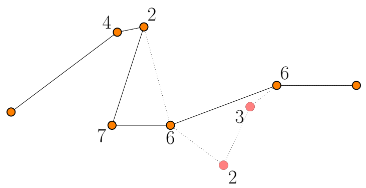
(g) An expansion of the removed segment with $\mathcal{L}(P, \Sigma) > \mathrm{T}_{\max}$.



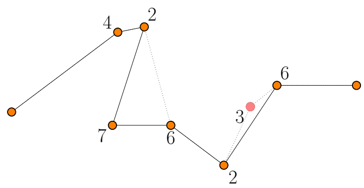
(h) Another expansion still with $\mathcal{L}(P, \Sigma) > \mathrm{T}_{\max}$.



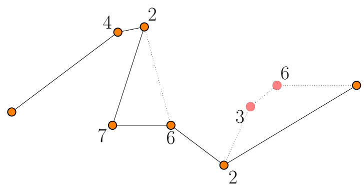
(i) Obtained travel budget $\mathcal{L}(P, \Sigma) \leq \mathrm{T}_{\max}$ but with the reward $R(\Sigma) = 19$.



(j) A shortening of the removed segment with $\mathcal{L}(P, \Sigma) \leq \mathrm{T}_{\max}$ and $R(\Sigma) = 25$. Thus, the path is put into the CL.



(k) Another shortening but with the exceeded travel budget $\mathcal{L}(P, \Sigma) > \mathrm{T}_{\max}$.



(l) The last shortening of the removed segment with $\mathcal{L}(P, \Sigma) > \mathrm{T}_{\max}$. Thus, the Segment Remove terminates.

Figure 4.1: An example of the Segment Remove for the radius $\varrho = 0$ and insertion of the location shown in green.

Definition 4.3 `remove`$(\overline{P}, \overline{\Sigma}, i)$**.**

$$\begin{aligned} \Sigma^* &= \left\langle \sigma_1, \ldots, \sigma_{i-1}, \sigma_{i+1}, \ldots, \sigma_{|\Sigma|} \right\rangle \\ P^* &= \left\langle \boldsymbol{p}_{\sigma_1}, \ldots, \boldsymbol{p}_{\sigma_{i-1}}, \boldsymbol{p}_{\sigma_{i+1}}, \ldots, \boldsymbol{p}_{\sigma_{|\Sigma|}} \right\rangle \end{aligned} \tag{4.4}$$

Subsequently, the Local Search Phase (LSP) depicted in Algorithm 3 is launched. In a

Algorithm 3: `localSearch`(P, Σ)

Input: P, Σ – the current path.
Output: $\overline{P}, \overline{\Sigma}$ – the updated path.

$(\overline{P}, \overline{\Sigma}) \leftarrow (P, \Sigma)$
repeat
  for $i \in \{2, \dots, |\overline{\Sigma}| - 1\}$ **do**
    $P', \Sigma' \leftarrow$ `remove`$(\overline{P}, \overline{\Sigma}, i)$ `// Using (4.4)`
    `2-opt`(P', Σ') `// See [20]`
    `optimization`(P', Σ') `// Using (4.5)`
    $b \leftarrow \overline{\sigma}_i$ `// Blocked index` $\overline{\sigma}_i \in \overline{\Sigma}$
    repeat
      $P', \Sigma' \leftarrow$ `addLocation`(P', Σ', b)
    until (P', Σ') is changed
    if $(R(\Sigma') > R(\overline{\Sigma})) \vee (R(\Sigma') = R(\overline{\Sigma}) \wedge \mathcal{L}(P', \Sigma') < \mathcal{L}(\overline{P}, \overline{\Sigma}))$ **then**
      $(\overline{P}, \overline{\Sigma}) \leftarrow (P', \Sigma')$
    end
  end
until $(\overline{P}, \overline{\Sigma})$ is changed

single LSP iteration, a location $\boldsymbol{v}_b$ at i-th position in the path is removed according to (4.4) to escape the local extreme. Then the path is being improved by the 2-Opt optimization, which removes the crossed segments.

Definition 4.4 `optimization`(P', Σ')**.**

$$
\begin{aligned}
(P^*, \Sigma^*) &= \texttt{remove}(P', \Sigma', j) \\
(P', \Sigma') &= \texttt{insertion}(P^*, \Sigma^*, j, \boldsymbol{v}_{\sigma'_j}), \quad \forall j \in \{2, \dots, |P'| - 1\}
\end{aligned}
\tag{4.5}
$$

After that, the `optimization` operator is executed to shorten the path length. It iteratively adjusts the path waypoint locations $\boldsymbol{p}_j$ for $j \in \{2, \dots, |P'| - 1\}$ by removal and insertion at the same position j based on the three proposed approaches detailed in Section 4.1–4.3. The position is updated to $\boldsymbol{p}'_j$ only if (4.6) holds.

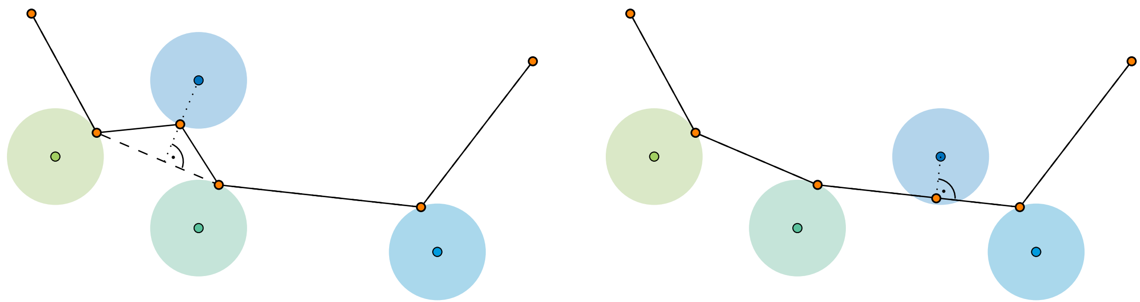
$$
\|\boldsymbol{p}_{j-1}, \boldsymbol{p}_j\| + \|\boldsymbol{p}_j, \boldsymbol{p}_{j+1}\| > \|\boldsymbol{p}_{j-1}, \boldsymbol{p}'_j\| + \|\boldsymbol{p}'_j, \boldsymbol{p}_{j+1}\| \tag{4.6}
$$

The `optimization` converges quickly, and three iterations provide suitable trade-off between the solution quality and computational requirements. Finally, the insertion of new locations might be enabled. The unvisited locations are attempted to be inserted in the same way as in the CP, but the location $\boldsymbol{v}_b$ is not included. The LSP procedure is more complex due to the usage of the `addLocation` procedure and a single iteration can be bounded by $\mathcal{O}(n^4)$.

4.1 Naive Approach

The Naive approach of the waypoint location determination is based on a straightforward idea of the closest point of a disk to a line, and it has already been employed in the GSOA [13].

There may occur two situations that are depicted in Figure 4.2. In the first situation, the current path does not cross the disk-shaped neighborhood area. Thus, the waypoint location is determined as the closest point of the sensing area to the path segment. In the second situation, the path goes through the disk-shaped area. Hence, the waypoint location is set to the closest point to the disk center of the crossing route part. Although the idea is relatively simple, it provides competitive solutions to the other approaches, as reported in [15] and Chapter 5.



(a) The location neighborhood area do not cross the path.

(b) The path go through the location neighborhood area.

Figure 4.2: Determination of a waypoint locations by the Naive approach.

Furthermore, the naive insertion can be employed in the CP only, or it can also be employed during the LSP. Thus, two variants are considered in the performed evaluation reported in Section 5.1 to study the influence of the `optimization` procedure.

4.2 LIO Approach

The second approach to the waypoint determination is based on the Local Iterative Optimization (LIO) [16] which is a continuous descendant procedure converging to the local optima. The initial position of the waypoint location is set in the same way as in the Naive approach. When the path does not go through the disk-shaped area, the LIO tries to shift the initial position within the edge of the sensing area by a small step, and path prolongation is examined. If the path is shortened, the waypoint location is updated, and the step is increased. Otherwise, the waypoint is not changed, the step is decreased and changes its direction. The descent is repeated until a given threshold for the step size is not reached. The particular value of the step threshold is 10^{-10} for all the results reported in this thesis. The idea is visualized in Figure 4.3.

4.3 SOCP Approach

Finding optimal waypoint location can be formulated as an optimization problem with quadratic constraints called the Second-Order Cone Program (SOCP). The path is determined for the sequence of k locations $\Sigma = \langle \sigma_1, \ldots, \sigma_k \rangle$ where we are searching for $|k| - 2$ waypoint locations $\boldsymbol{p}_j$ that the distance between each waypoint location $\|\boldsymbol{p}_i, \boldsymbol{p}_{i+1}\|$ for $i \in \{1, \ldots, k-1\}$ is minimal. The optimization problem is formally defined as Problem 4.1 with the notation visualized in Figure 4.4.

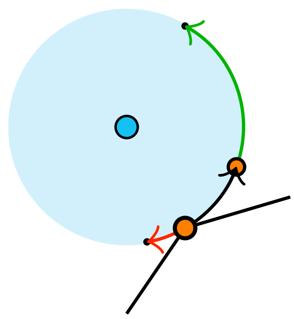


Figure 4.3: The Local Iterative Optimization (LIO) method refines the waypoint location. The black arrow shows the examined step of the adjustment. When it is accepted, the next step is the green one. Otherwise, the waypoint location do not change and the next examined step is the red one.

The SOCP can be solved using IBM ILOG CPLEX Optimization Studio [17]. The solution of the SOCP provides the optimal waypoint locations for the particular order of visit to the currently selected regions. Even though it can be computationally demanding, it has been implemented to observe the influence of determining optimal waypoint location to the solution quality of the CEOP.

Notice that the SOCP formulation in the CP is only for the insertion of the new location between two existing waypoints. Therefore, the SOCP is for a sub-path of the size $k = 3$. On the other hand, the `optimization` operator in the LSP finds the optimal waypoint locations of the whole path $k = |P|$.

Problem 4.1 The Second-Order Cone Program (SOCP) for the waypoint locations.

$$
\begin{aligned}
\min_{\boldsymbol{p}_j} \quad & \sum_{i=1}^{k-1} d_i \\
\text{s. t.} \quad & d_i = \|\boldsymbol{p}_i, \boldsymbol{p}_{i+1}\| \,, \quad \forall i \in \{1, \dots, k-1\} \\
& \left\|\boldsymbol{v}_{\sigma_j}, \boldsymbol{p}_j\right\| \le \varrho \,, \quad \forall j \in \{2, \dots, k-1\} \\
& \boldsymbol{p}_1 = \boldsymbol{v}_{\sigma_1} \,, \quad \boldsymbol{p}_k = \boldsymbol{v}_{\sigma_k}
\end{aligned}
$$

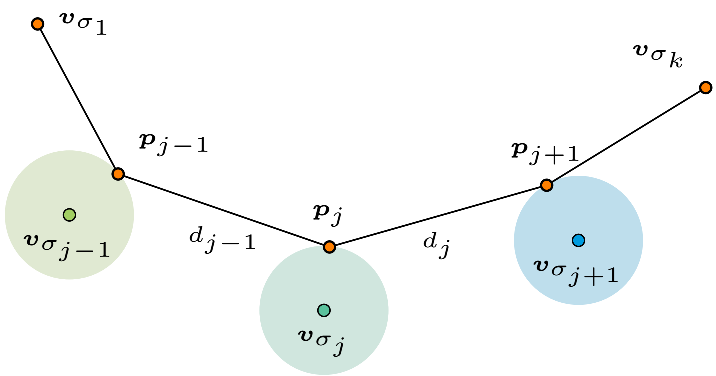


Figure 4.4: Relations of the variables for the SOCP formulation of Problem 4.1.

4.4 Heuristic of the Ordered Placing

Due to the relatively demanding waypoint location determination based on the LIO and SOCP, we propose the so-called Heuristic of the Ordered Placing (HOP) to prevent excessive determination of new waypoint locations. The idea is based on sorting the prospective prolongations to save the computational time, and it works as follows.

Before a possible insertion of a new location, the lower bound on the expected prolongation Δ_L is precalculated based on the distance to the center of the new sensing area according to

$$\Delta_L = \|\boldsymbol{p}_i, \boldsymbol{v}\| + \|\boldsymbol{v}, \boldsymbol{p}_j\| - 2\varrho - \|\boldsymbol{p}_i, \boldsymbol{p}_j\| \ , \tag{4.7}$$

where the relation between the waypoint locations is visualized in Figure 4.5. Afterward, the new location is attempted to be inserted into the path segment in ascending order of the lower bound values. The possible insertion is examined only if the estimated lower bound of the prolongation Δ_L is shorter than the current shortest prolongation determined so far. Therefore, the LIO or more computationally expensive SOCP is examined only for the most promising insertions. Determination of (4.7) is computationally efficient, and thus, the real impact on the computational requirements is reported in the next chapter.

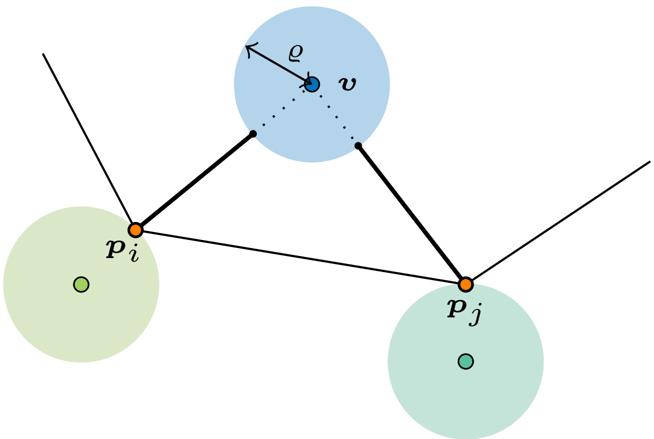


Figure 4.5: Relation of the waypoint location and possible insertion of the $\boldsymbol{v}$ between $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ in the computation of the lower bound estimation of the path prolongation utilized in the proposed HOP.

CHAPTER 5

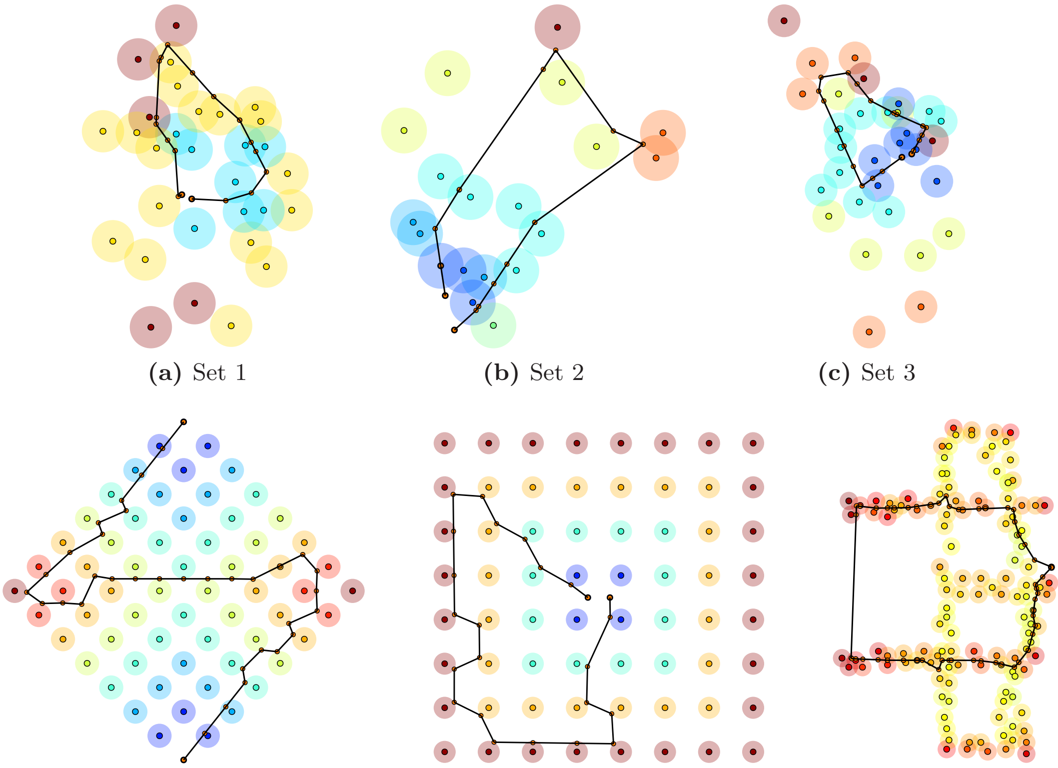
Results

The proposed GRASP-based algorithms to the CEOP have been empirically evaluated using three datasets of existing benchmark instances. The first three scenarios proposed by Tsiligirides [8] are called Set 1, Set 2, and Set 3, each with 31, 20, and 32 locations, respectively, which is rather small. Therefore the second sets include larger instances of the diamond-shaped Set 64 and the square-shaped Set 66 introduced by Chao et al. in [36]. The last dataset is Set 130 [13], where the number of locations increases to 130. Examples of instances are depicted in Figure 5.1. Each instance is defined by the scenario, the travel budget $\mathrm{T}_{\max}$, and the sensing range ϱ. For each dataset, a set of particular instance values is established. Increasing sensing range and the travel budget allow the visitation of all the sites. Thus, if one approach with a specific sensing radius has obtained all locations, the higher values of the travel budget are excluded because the solution would be the maximal reward possible. By avoiding such instances, it limits the bias of the average relative value representing the quality of the solutions. The sensing range is selected from the set $\varrho \in \{0.0, 0.5, 1.0, 1.5, 2.0\}$, where $\varrho = 0$ stands for the regular OP.

The developed GRASP approaches are compared to the existing heuristics, the GSOA [13] based on unsupervised learning, and the combinatorial VNS [14], which searches through the solution space with eight samples per each disk-shaped neighborhood area. The VNS-based algorithm is terminated after 1000 iterations or 200 iterations without improvement. All the algorithms have been implemented in C++ and run within the same computational environment using a single core of the Intel Core i5-4460 CPU running at 3.2 GHz.

Each test instance is solved 20 times by each solver because all the methods are randomized algorithms. The reported solution R is the maximal collected reward overall particular runs as utilized in the previous OP and CEOP studies [14]. However, due to several runs of many instances defined by the number of the travel budget $\mathrm{T}_{\max}$ and sensing radius ϱ, aggregated results are reported as the average gap $\overline{G}$ among the solutions of the particular scenario. The gap G represents the relative difference to the best-found solution R_{ref} among all performed runs by all examined methods. Thus, for each problem instance, defined by the scenario, the travel budget $\mathrm{T}_{\max}$ and the sensing range ϱ, the particular value of the reference solution R_{ref} is established. Then, the value of the gap G is calculated as

$$G(\Sigma) = \left(1 - \frac{R(\Sigma)}{R_{\mathrm{ref}}}\right) \cdot 100\ [\%] \tag{5.1}$$



(a) Set 1 **(b)** Set 2 **(c)** Set 3

(d) Set 64 **(e)** Set 66 **(f)** Set 130

Figure 5.1: Example of examined instances of the selected benchmark datasets where the colored disks represent the sensing area of each particular location. The red color shows the most valuable locations and the blue stands for the least rewarding ones.

where $R(\Sigma)$ is the reward of the solution found by the individual algorithm for the particular instance.

5.1 Influence of the Waypoint Locations Optimization

Firstly, the influence of the `optimization` operator on the final solution is studied, and two variants of the Naive approach proposed in Section 4.1 have been evaluated. The first variant is denoted **GRASP-Naive**$_{\text{simple}}$, and it employs the waypoint location heuristic only for determining the waypoint location while inserting. The second approach denoted **GRASP-Naive** employs the waypoint determination also in the LSP to improve the waypoint locations of the whole path.

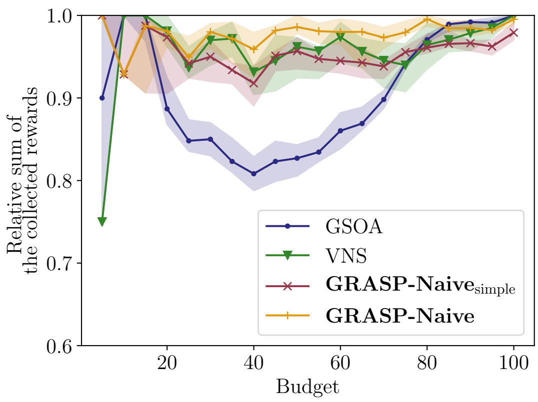
The average gap for each dataset is listed in Table 5.1 to show the importance of the optimization procedure for finding high-quality solutions. The best-provided results for each dataset are highlighted in bold. An example instance of the square-shaped Chao Set 66 with the sensing radius $\varrho = 0.5$ is depicted in Figure 5.2. It shows the solution quality according to the mean collected rewards normalized to the reference R_{ref} with an 80 % non-parametric confidence interval. The right plot depicts the required computational time in milliseconds on a logarithmic scale.

Even though the **GRASP-Naive** approach is based on a simple and straightforward

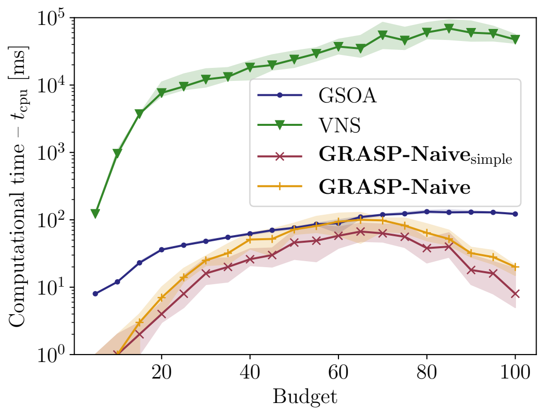
Table 5.1: Aggregated results for the optimization of the waypoint locations

Instances	GSOA G [%]	VNS G [%]	**GRASP-Naive**$_{\text{simple}}$ G [%]	**GRASP-Naive** G [%]
Set 1	9.27	2.94	1.53	**0.64**
Set 2	2.17	2.15	0.79	**0.40**
Set 3	0.75	1.46	0.67	**0.16**
Set 64	2.69	3.29	0.97	**0.54**
Set 66	15.70	3.02	3.24	**0.55**
Set 130	6.44	**0.01**	5.00	1.26

heuristic, it provides better solutions than the GSOA with competitive computational requirements. Furthermore, the proposed **GRASP-Naive** provides solutions with the competitive quality to the VNS while it is only slightly more demanding than **GRASP-Naive**$_{\text{simple}}$.



(a) Relative sum of the collected rewards - R/R_{ref}



(b) Required computational time - t_{cpu}

Figure 5.2: Solution quality and computational requirements of the existing methods and proposed approaches with and without the waypoint optimization for Set 66 with the sensing radius $\varrho = 0.5$. The quality is shown as the relative sum of the collected rewards R normalized by the reference solution R_{ref} for each specific problem instance. The showed curve represents the mean value, and the semi-transparent area stands for 80 % non-parametric confidence interval.

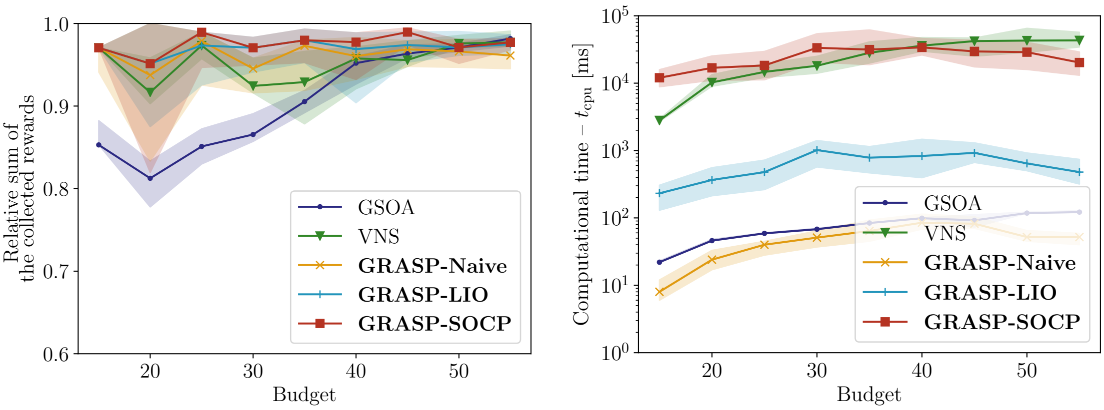
5.2 Influence of the Waypoint Location Determination

The three proposed methods of the waypoint location determination proposed in Section 4 are the Naive approach, the LIO approach denoted **GRASP-LIO**, and the SOCP approach denoted **GRASP-SOCP**. Based on the results on variants of the Naive approach reported in Section 5.1, only the **GRASP-Naive** is considered here as it provides noticeable better results with only a minor increase in the computational requirements. For now, the **GRASP-LIO** and **GRASP-SOCP** are both without the proposed HOP.

The aggregated results with the average gap $\overline{G}$ for each examined benchmark scenario are listed in Table 5.2. The results for Set 64 with the sensing radius $\varrho = 0.5$ are depicted in Figure 5.3. The plots show the average relative sum of rewards normalized to referential R_{ref}

Table 5.2: Aggregated results for the existing methods and proposed approaches

Instances	GSOA G [%]	VNS G [%]	**GRASP-Naive** G [%]	**GRASP-LIO** G [%]	**GRASP-SOCP** G [%]
Set 1	10.42	4.33	2.02	0.92	**0.00**
Set 2	3.15	3.13	1.38	**0.00**	0.79
Set 3	1.69	2.40	1.09	**0.00**	**0.00**
Set 64	3.96	4.55	1.83	1.20	**0.00**
Set 66	17.17	4.64	2.17	0.68	**0.05**
Set 130	6.78	0.37	1.62	0.73	**0.16**

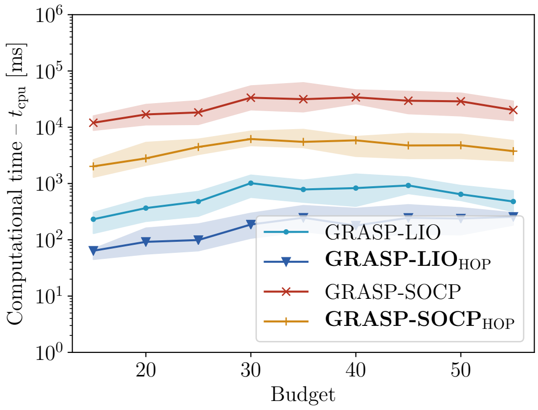


(a) Relative sum of the collected rewards - R/R_{ref} **(b)** Required computational time - t_{cpu}

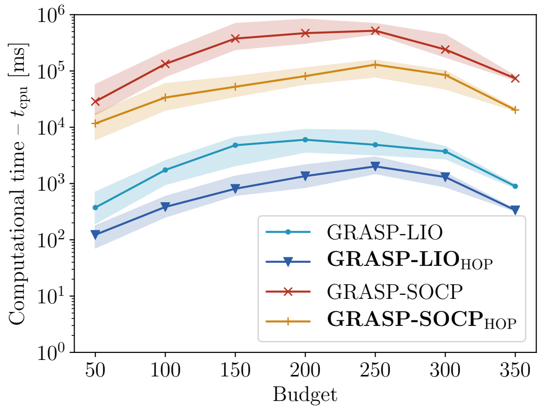
Figure 5.3: Solution quality and computational requirements of the existing methods and proposed approaches for Set 64 with the sensing radius $\varrho = 0.5$. The quality is shown as the relative sum of the collected rewards R normalized by the reference value R_{ref} for each specific problem instance. The showed curve represents the mean value and the semi-transparent area stands for 80 % non-parametric confidence interval.

and the mean computational time. Detailed results for the selected travel budgets $\mathrm{T}_{\max}$ and sensing radii ϱ are reported in Table A.1, Table A.2 and Table A.3.

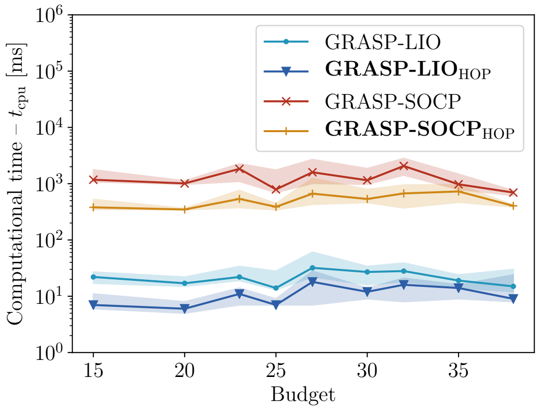
The LIO and SOCP based approaches improve the solution quality and grant more stable collected rewards within all batches of a single instance. Although the SOCP provides an optimal solution, its results correspond to the VNS method solution quality and computational requirements. Furthermore, the SOCP-based approach usually obtained the reference R_{ref} overall runs. On the other hand, the LIO-based approach is able to reach half of the computational time of the VNS with no significant change in the solution quality. All three approaches outperformed the GSOA with the total collected rewards, where only the Naive approach competes with the computational requirements. However, the proposed **GRASP-LIO** seems to provide a suitable trade-off between the solution quality and computational requirements.



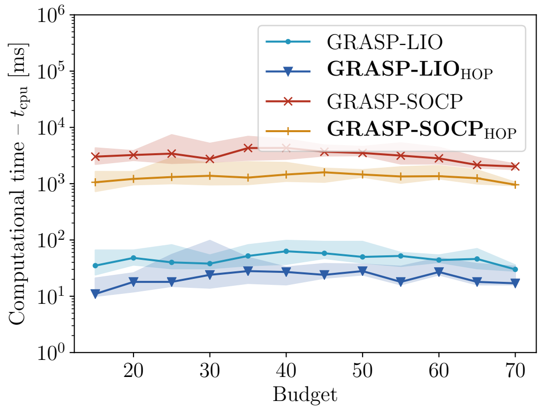
(a) Set 64 with the sensing radius $\varrho = 0.5$



(b) Set 130 with the sensing radius $\varrho = 1.0$



(c) Set 2 with the sensing radius $\varrho = 0.5$

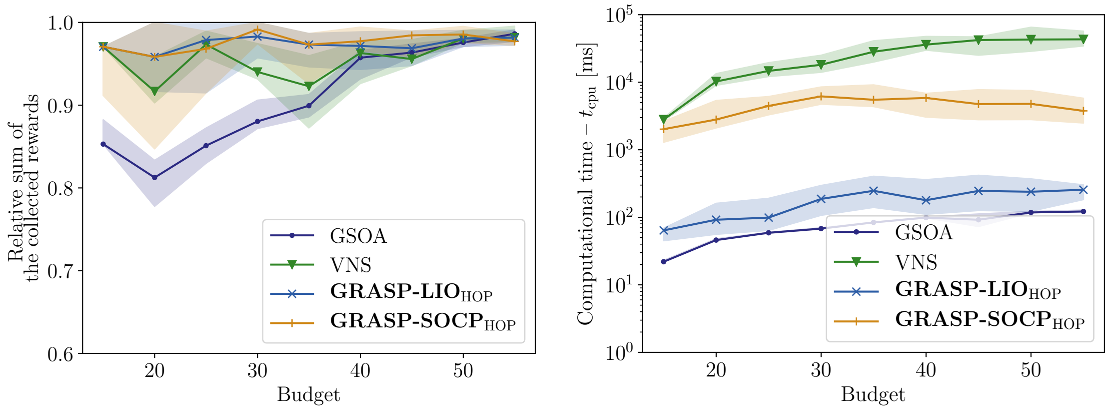


(d) Set 3 with the sensing radius $\varrho = 1.5$

Figure 5.4: Required computational time of the developed LIO and SOCP-based GRASP methods with and without the proposed Heuristic of the Ordered Placing in selected scenarios. The showed curves represent the mean values and the semi-transparent areas stand for 80 % non-parametric confidence intervals.

5.3 Efficiency of the Heuristic of the Ordered Placing (HOP)

The Heuristic of the Ordered Placing (HOP) has been proposed to reduce real-time computational requirements of the LIO and SOCP based approaches. Thus, the variants with the HOP are denoted **GRASP-LIO**$_{\text{HOP}}$ and **GRASP-SOCP**$_{\text{HOP}}$. The HOP addresses only the computational requirements and does not change the solution quality. Therefore only the required computational times are reported in Figure 5.4 that empirically support the efficiency of the proposed HOP. In all instances, the average computational time of both methods has decreased. For a comparison with the existing approaches, the solution quality and computational time are reported for the HOP-based improved and **GRASP-SOCP**$_{\text{HOP}}$ in Figure 5.5 together with the solutions for the GSOA and VNS based methods. By employing the proposed HOP, the computational requirements are reduced such that the average computational time of the **GRASP-SOCP**$_{\text{HOP}}$ is lower than the one of VNS. Also, the **GRASP-LIO**$_{\text{HOP}}$ becomes similarly demanding as the GSOA.



(a) Relative sum of the collected rewards - R/R_{ref}

(b) Required computational time - t_{cpu}

Figure 5.5: Conparison of the improved HOP approaches to the existing methods. It shows relative sum of the collected rewards R and the required computational time of the evaluated approaches of Set 64 with the sensing radius $\varrho = 0.5$. The sum of the collected reward is normalized to the referential value R_{ref} for each specific instance of the evaluated methods. The line represents the mean and the semi-transparent area stands for 80 % non-parametric confidence interval.

5.4 Candidate List Restriction

Finally, we examined the restriction of the CL to the solution quality. Campos et al. [10] suggested the optimal value of the restriction parameter is $c_{\mathrm{best}} = 20\,\%$. Since the proposed GRASP-based approach for the CEOP differs from the original approach to the OP, we studied the influence of c_{best} to the solution quality of the proposed GRASP-LIO and GRASP-SOCP. The results for different values of the restriction parameter are reported in Table 5.3 and Table 5.4 as the average gap $\overline{G}$. Four values of c_{best} are considered for the LIO-based approach: $c_{\mathrm{best}} = 0\,\%$, $c_{\mathrm{best}} = 10\,\%$, $c_{\mathrm{best}} = 20\,\%$, $c_{\mathrm{best}} = 40\,\%$, $c_{\mathrm{best}} = 60\,\%$ and $c_{\mathrm{best}} = 80\,\%$. For the SOCP-based approach, we examined its performance for $c_{\mathrm{best}} = 20\,\%$, $40\,\%$, $60\,\%$ and $80\,\%$.

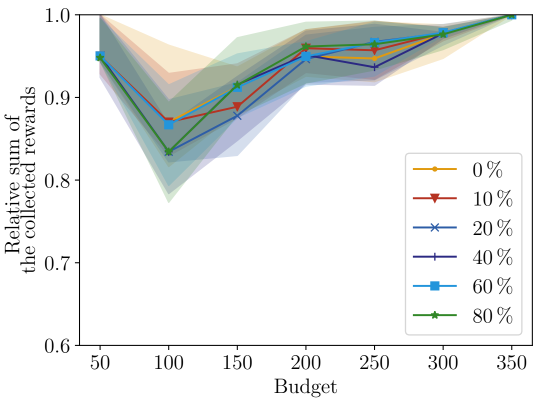
The LIO-based approach performs best (in average) for $c_{\mathrm{best}} = 10\,\%$, but $c_{\mathrm{best}} = 20\,\%$ (as suggested in [10]) does not provide significantly worse solutions. On the other hand, for the SOCP-based approach, the best results are provided with $c_{\mathrm{best}} = 60\,\%$. Nevertheless, when we look at all the aggregated instances and mean results in Figure 5.6, the restriction parameter does not significantly influence either solution quality nor computational time. Thus, we cannot uniformly choose the best c_{best} for the GRASP-based CEOP approaches. However, the aggregated results for $c_{\mathrm{best}} = 0\,\%$ show that some restriction is essential.

Table 5.3: Aggregated results for the restriction parameter of the LIO approach

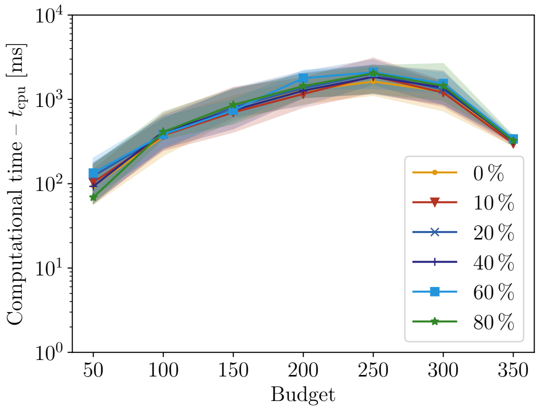
Instances	LIO c_{best}	0 % G [%]	10 % G [%]	20 % G [%]	40 % G [%]	60 % G [%]	80 % G [%]
Set 1		**0.53**	**0.53**	**0.53**	0.72	**0.53**	1.77
Set 2		0.62	**0.00**	0.62	**0.00**	2.09	1.60
Set 3		**0.00**	**0.00**	**0.00**	0.15	0.30	**0.00**
Set 64		1.31	1.29	1.08	1.16	0.74	**0.18**
Set 66		1.02	0.83	0.61	0.94	0.73	**0.54**
Set 130		1.12	1.07	**1.05**	1.38	1.50	1.10
Average		0.77	**0.62**	0.65	0.73	0.98	0.87

Table 5.4: Aggregated results for the restriction parameter of the SOCP approach

Instances	SOCP c_{best}	20 % G [%]	40 % G [%]	60 % G [%]	80 % G [%]
Set 1		**0.00**	**0.00**	**0.00**	0.39
Set 2		**0.62**	**0.62**	**0.62**	**0.62**
Set 3		0.15	**0.00**	**0.00**	**0.00**
Set 64		0.53	0.22	**0.14**	0.59
Set 66		0.21	0.39	**0.09**	0.14
Set 130		0.53	0.82	0.28	**0.11**
Average		0.34	0.34	**0.18**	0.31



(a) Relative sum of the collected rewards - R/R_{ref}



(b) Required computational time - t_{cpu}

Figure 5.6: Relative sum of the collected rewards R and the required computational time of the evaluated approaches of Set 130 with the sensing radius $\varrho = 1.0$. The sum of the collected reward is normalized to the highest found collected rewards R_{ref} for each specific instance of the evaluated methods. The curve represents the mean and the semi-transparent area stands for 80 % non-parametric confidence interval.

CHAPTER 6

Conclusion

In this thesis, we address the Close Enough Orienteering Problem (CEOP) motivated by data collection planning. A survey of the existing methods has outlined the drawbacks of existing algorithms for the CEOP, where the GSOA [13] is unable to escape the local extreme and the VNS [14] is computationally demanding. Based on reported results of the Greedy Randomized Adaptive Search (GRASP) with Segment Remove [32] in the solution of the OP, we proposed to generalize the GRASP-based approach to the solution of the CEOP, and thus address drawbacks fo the GSOA and VNS based methods.

The GRASP for the OP might be extended for the CEOP in two parts. The first is in the determination of the suitable waypoint location during the possible insertion of the site into the solution. However, when such a waypoint is determined, it is not further updated. Therefore we further propose to optimize the waypoint locations during the GRASP finding of the solution. We propose three approaches on how to determine waypoint locations. The first is the Naive approach that has been published in [15]. Although it utilizes an easy and straightforward heuristic that sets the waypoint location as the closest point of the sensing area to the current path, it provides competitive performance to the existing methods to the CEOP. The next proposed approach is based on the Local Iterative Optimization (LIO) [16], which by continuous descent, determines the waypoint location to the position with the shortest prolongation, and it further improves the quality of the found solutions. Finally, the third approach is based on the optimal waypoint location determined by the Second-Order Cone Program (SOCP), which is an optimization problem with quadratic constraints. The SOCP is solved by IBM ILOG CPLEX Optimization Studio [17], and the SOCP-based approach provides the best results of the examined benchmarks of the CEOP. However, LIO and SOCP are computationally demanding. Therefore, we propose the Heuristic of the Ordered Placing (HOP) to decrease the computational burden by excluding excessive waypoint determination.

The performance of the proposed GRASP-based approaches has been studied for existing benchmarks on the CEOP already utilized in the literature, and solutions have been compared to the existing GSOA and VNS based approaches. At first, the importance of the waypoint location optimization has been examined for the proposed Naive approach, and according to the reported results, the proposed optimization improves the solution quality noticeably. Although the Naive approach is based on a relatively simple heuristic, the results are competitive to the VNS-based solver, but the computational requirements are competitive to the fast GSOA. The proposed LIO and SOCP based approaches are able to grant more rewarding solutions than the GSOA and more stable than the VNS approach. Furthermore, the SOCP based approach provides the best solutions overall methods for most of the examined instances. The increased computational requirements of the LIO and SOCP based approaches are addressed by the proposed Heuristic of the Ordered Placing (HOP). Thus, the developed **GRASP-LIO**$_{\text{HOP}}$ has computational requirements similar to the GSOA, but it provides significantly better solutions. Moreover, the **GRASP-SOCP**$_{\text{HOP}}$ outperforms the examined VNS-based solution in the solution quality, and it is less demanding in most of the cases. Therefore, the proposed **GRASP-SOCP**$_{\text{HOP}}$ can be recommended to obtain high-quality solutions when computational time is not a limiting factor. In contrast, the proposed **GRASP-LIO**$_{\text{HOP}}$ provides a suitable trade-off between the computational requirements and solution quality.

For future work, the GRASP-based method can be further extended to 3D instances. Besides, it can be modified to the Dubins OP, where the connections path connecting the sensing sites has to satisfy curvature constraints, and which has also been addressed by the GSOA and VNS-based approaches. Based on the herein reported results, the GRASP-based method can outperform the current existing solvers similarly as in the addressed CEOP.

Bibliography

[1] Bruce L. Golden, Larry Levy, and Rakesh Vohra. The orienteering problem. *Naval Research Logistics (NRL)*, 34(3):307–318, 1987.

[2] Keith W Ross and Danny HK Tsang. The stochastic knapsack problem. *IEEE Transactions on communications*, 37(7):740–747, 1989.

[3] David L Applegate, Robert E Bixby, Vasek Chvatal, and William J Cook. *The traveling salesman problem: a computational study*. Princeton university press, 2006.

[4] Graeme Best, Jan Faigl, and Robert Fitch. Online planning for multi-robot active perception with self-organising maps. *Autonomous Robots*, 42(4):715–738, 2018.

[5] Aldy Gunawan, Hoong Chuin Lau, and Pieter Vansteenwegen. Orienteering problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research*, 255(2):315–332, 2016.

[6] C Peter Keller. Algorithms to solve the orienteering problem: A comparison. *European Journal of Operational Research*, 41(2):224–231, 1989.

[7] Matteo Fischetti, Juan Jose Salazar Gonzalez, and Paolo Toth. Solving the orienteering problem through branch-and-cut. *INFORMS Journal on Computing*, 10(2):133–148, 1998.

[8] Theodore Tsiligirides. Heuristic methods applied to orienteering. *Journal of the Operational Research Society*, 35(9):797–809, 1984.

[9] Pierre Hansen and Nenad Mladenović. Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, 130(3):449–467, 2001.

[10] Vicente Campos, Rafael Martí, Jesús Sánchez-Oro, and Abraham Duarte. Grasp with path relinking for the orienteering problem. *Journal of the Operational Research Society*, 65(12):1800–1813, 2014.

[11] Qiwen Wang, Xiaoyun Sun, Bruce L Golden, and Jiyou Jia. Using artificial neural networks to solve the orienteering problem. *Annals of Operations Research*, 61(1):111–120, 1995.

[12] Jan Faigl, Robert Pěnička, and Graeme Best. Self-organizing map-based solution for the orienteering problem with neighborhoods. In *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 001315–001321. IEEE, 2016.

[13] Jan Faigl. Data collection path planning with spatially correlated measurements using growing self-organizing array. *Applied Soft Computing*, 75:130–147, 2019.

[14] Robert Pěnička, Jan Faigl, Petr Váňa, and Martin Saska. Dubins orienteering problem. *IEEE Robotics and Automation Letters*, 2(2):1210–1217, 2017.

[15] Petra Štefaníková, Petr Váňa, and Jan Faigl. Greedy randomized adaptive search procedure for close enough orienteering problem. In *35th Annual ACM Symposium on Applied Computing*, pages 808–814, 2020.

[16] Petr Váňa and Jan Faigl. On the dubins traveling salesman problem with neighborhoods. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4029–4034. IEEE, 2015.

[17] IBM ILOG CPLEX Optimization studio CPLEX user's manual, 2017. Version 12 Release 7.

[18] R Ramesh, Yong-Seok Yoon, and Mark H Karwan. An optimal algorithm for the orienteering tour problem. *ORSA Journal on Computing*, 4(2):155–165, 1992.

[19] Anthony Wren and Alan Holliday. Computer scheduling of vehicles from one or more depots to a number of delivery points. *Journal of the Operational Research Society*, 23(3):333–344, 1972.

[20] G. A. Croes. A method for solving traveling-salesman problems. *Operations Research*, 6(6):791–812, 1958.

[21] Ram Ramesh and Kathleen M Brown. An efficient four-phase heuristic for the generalized orienteering problem. *Computers & Operations Research*, 18(2):151–165, 1991.

[22] Shen Lin. Computer solutions of the traveling salesman problem. *Bell System Technical Journal*, 44(10):2245–2269, 1965.

[23] Fred Glover and Eric Taillard. A user's guide to tabu search. *Annals of operations research*, 41(1):1–28, 1993.

[24] Michel Gendreau, Gilbert Laporte, and Frédéric Semet. A tabu search heuristic for the undirected selective travelling salesman problem. *European Journal of Operational Research*, 106(2-3):539–545, 1998.

[25] Michel Gendreau, Alain Hertz, and Gilbert Laporte. New insertion and postoptimization procedures for the traveling salesman problem. *Operations Research*, 40(6):1086–1094, 1992.

[26] Marco Dorigo and Luca Maria Gambardella. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on evolutionary computation*, 1(1):53–66, 1997.

[27] Yun-Chia Liang and Alice E Smith. An ant colony approach to the orienteering problem. *Journal of the Chinese Institute of Industrial Engineers*, 23(5):403–414, 2006.

[28] Nenad Mladenović and Pierre Hansen. Variable neighborhood search. *Computers & operations research*, 24(11):1097–1100, 1997.

[29] Zülal Sevkli and F Erdoğan Sevilgen. Variable neighborhood search for the orienteering problem. In *International Symposium on Computer and Information Sciences*, pages 134–143. Springer, 2006.

[30] Pierre Hansen and Nenad Mladenović. Variable neighborhood search: Principles and applications. *European journal of operational research*, 130(3):449–467, 2001.

[31] Mauricio GC Resende and Celso C Ribeiro. *Optimization by GRASP*. Springer, 2016.

[32] Morteza Keshtkaran and Koorush Ziarati. A novel grasp solution approach for the orienteering problem. *Journal of Heuristics*, 22(5):699–726, 2016.

[33] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[34] Jan Faigl, Robert Pěnička, and Graeme Best. Self-organizing map-based solution for the orienteering problem with neighborhoods. In *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 1315–1321, 2016.

[35] Robert Pěnička, Jan Faigl, and Martin Saska. Variable neighborhood search for the set orienteering problem and its application to other orienteering problem variants. *European Journal of Operational Research*, 276(3):816–825, 2019.

[36] I-Ming Chao, Bruce L. Golden, and Edward A. Wasil. A fast and effective heuristic for the orienteering problem. *European Journal of Operational Research*, 88(3):475–489, 1996.

APPENDIX A

Tables

Table A.1: Results for the CEOP instaces of Set 1, Set 2 and Set 3

CEOP Instances	T_{max}	GSOA		VNS		**GRASP-Naive$_{\text{opt}}$**		**GRASP-LIO**		**GRASP-SOCP**	
		R	t_{cpu} [ms]	R	t_{cpu} [ms]	R	t_{cpu} [ms]	R	t_{cpu} [ms]	R	t_{cpu} [ms]
Set 1 ($\varrho = 0.5$)	25	105	20.2	**110**	2117.8	**110**	2.4	**110**	40.4	**110**	2996.8
	35	155	25.4	**165**	3104.4	160	4.6	**165**	69.0	**165**	5330.8
	46	200	30.5	205	3919.6	205	6.5	**210**	96.2	**210**	6063.0
Set 1 ($\varrho = 1$)	25	125	21.7	**135**	2452.4	130	2.8	**135**	46.6	**135**	2633.6
	35	185	27.0	**190**	3544.5	**190**	6.0	**190**	79.6	**190**	4029.1
	46	230	34.6	230	4153.8	230	5.2	**235**	68.8	**235**	4417.6
Set 1 ($\varrho = 1.5$)	25	155	23.2	155	2573.3	160	3.8	**165**	64.8	**165**	4644.4
	35	205	29.8	210	4397.6	210	5.2	**215**	74.9	**215**	2773.0
	46	260	36.2	255	6055.6	260	4.2	**265**	43.0	**265**	2581.5
Set 1 ($\varrho = 2$)	25	**175**	25.6	170	3080.6	**175**	2.2	**175**	60.4	**175**	3459.6
	35	**225**	33.0	**225**	3908.4	220	4.4	**225**	39.0	**225**	2875.6
	46	**285**	39.1	275	5482.6	275	3.5	275	40.6	**285**	1929.0
Set 2 ($\varrho = 0.5$)	15	**180**	12.6	**180**	449.6	**180**	1.0	**180**	22.0	**180**	1500.8
	30	330	18.8	**340**	1404.6	330	0.9	**340**	27.0	**340**	1457.0
Set 2 ($\varrho = 1$)	15	**200**	12.1	**200**	609.4	**200**	0.4	**200**	15.3	**200**	851.3
	30	**400**	19.6	390	1756.9	390	1.4	**400**	25.5	**400**	1089.4
Set 2 ($\varrho = 1.5$)	15	200	12.6	200	707.5	200	0.6	**210**	16.4	**210**	942.6
	30	**450**	20.0	**450**	1588.5	**450**	2.2	**450**	23.0	**450**	738.6
Set 2 ($\varrho = 2$)	15	200	12.0	**230**	820.3	**230**	0.4	**230**	12.0	**230**	950.3
	30	**450**	20.0	**450**	1695.4	**450**	0.8	**450**	22.8	**450**	678.8
Set 3 ($\varrho = 0.5$)	15	**180**	14.4	**180**	619.3	**180**	0.5	**180**	26.2	**180**	2284.4
	40	490	29.2	**500**	3157.3	490	6.2	490	114.8	490	8883.6
Set 3 ($\varrho = 1$)	15	**210**	14.6	**210**	934.5	200	1.5	**210**	38.2	**210**	3309.5
	40	560	31.8	**570**	4221.9	560	5.2	**570**	89.0	**570**	5330.6
Set 3 ($\varrho = 1.5$)	15	**250**	15.9	230	1104.5	240	2.2	**250**	43.5	**250**	3425.1
	40	**600**	34.5	**600**	4916.0	**600**	5.8	**600**	67.6	**600**	4685.4
Set 3 ($\varrho = 2$)	15	280	17.6	270	1637.0	**290**	1.6	**290**	40.2	**290**	2729.0
	40	630	36.8	630	4803.2	630	4.7	**650**	61.7	**650**	4162.6

Table A.2: Results for the CEOP instaces of Set 64 and Set 66

CEOP Instances	T_{max}	GSOA		VNS		**GRASP-Naive$_{\text{opt}}$**		**GRASP-LIO**		**GRASP-SOCP**	
		R	t_{cpu} [ms]	R	t_{cpu} [ms]	R	t_{cpu} [ms]	R	t_{cpu} [ms]	R	t_{cpu} [ms]
	35	822	84.1	858	30078.6	882	72.0	**888**	783.6	882	39119.3
Set 64 ($\varrho = 1$)	15	288	29.0	300	6352.3	300	8.6	**312**	224.8	**312**	12952.1
	35	1116	108.5	1152	41783.4	1140	54.0	1158	490.8	**1164**	34286.3
Set 64 ($\varrho = 1.5$)	15	372	36.6	324	7446.4	**414**	10.0	**414**	336.7	**414**	20427.4
	35	**1344**	136.8	1284	47363.6	1308	57.0	**1344**	212.7	**1344**	15062.8
	35	**1344**	134.3	**1344**	35763.7	**1344**	12.4	**1344**	128.1	**1344**	11545.8
Set 66 ($\varrho = 0.5$)	5	**20**	8.1	15	123.4	**20**	0.4	**20**	16.4	**20**	680.7
	25	360	42.4	380	11160.8	395	14.5	**400**	275.4	**400**	13025.6
	55	940	86.8	1045	30342.9	1045	85.4	**1050**	1015.4	1045	38833.5
Set 66 ($\varrho = 1$)	5	20	6.8	30	264.6	**35**	0.9	**35**	32.2	**35**	1269.6
	25	495	48.4	**540**	12333.0	535	23.0	**540**	196.6	535	12129.2
	55	1340	106.4	1385	47294.2	**1415**	78.8	**1415**	479.0	**1415**	20694.8
Set 66 ($\varrho = 1.5$)	5	20	6.9	**50**	339.8	**50**	0.7	**50**	32.0	**50**	1882.7
	25	555	55.3	600	16308.6	**625**	31.1	**625**	247.4	**625**	21372.8
	55	1520	125.9	1485	44795.2	**1555**	148.2	1550	365.0	**1555**	17382.4
Set 66 ($\varrho = 2$)	5	20	7.8	80	553.8	**95**	2.7	**95**	61.9	**95**	3906.6
	25	660	63.2	735	19011.5	740	41.5	765	453.9	**785**	19647.7
	55	**1680**	130.8	**1680**	39679.9	1675	70.8	**1680**	459.4	**1680**	7820.7

Table A.3: Results for the CEOP instaces of Set 130

CEOP Instances	T_{max}	GSOA		VNS		**GRASP-Naive$_{\text{opt}}$**		**GRASP-LIO**		**GRASP-SOCP**	
		R	t_{cpu} [ms]	R	t_{cpu} [ms]	R	t_{cpu} [ms]	R	t_{cpu} [ms]	R	t_{cpu} [ms]
Set 130 ($\varrho = 0$)	50	**375**	30.4	**375**	753.4	**375**	11.4	**375**	95.6	**375**	32485.2
	100	824	103.8	**896**	10989.8	878	128.2	857	737.1	872	282445.2
	150	1210	145.4	**1369**	15355.7	1297	265.9	1250	1434.8	1250	596947.4
	200	1566	196.5	**1809**	16740.9	1688	505.6	1728	3273.4	1751	975915.6
	250	2075	264.0	2218	24273.8	**2264**	1073.2	2233	4436.6	2215	1307109.7
	300	2517	339.3	2703	40533.2	**2739**	1202.4	2737	5802.4	2735	1397553.2
	350	3021	409.9	3075	48232.6	3138	1039.6	**3155**	4751.4	**3155**	1167233.0
	400	3504	446.2	3526	55548.4	3512	358.2	**3535**	1835.4	3532	387303.0
	410	**3586**	444.6	3558	71975.0	**3586**	186.8	3584	1495.2	**3586**	222912.2
Set 130 ($\varrho = 1$)	50	**462**	37.0	**462**	11067.4	**462**	23.6	**462**	433.8	**462**	35991.0
	100	1089	134.4	1201	116078.4	1138	158.8	**1204**	1744.8	**1204**	150495.3
	150	1534	195.0	**1846**	175414.2	1777	473.8	1781	4720.0	1834	451869.0
	200	2082	259.6	**2465**	288745.5	2385	788.0	2400	6649.1	2424	544818.2
	250	2802	367.4	2967	326766.0	**3003**	852.4	3000	5252.3	**3003**	573457.8
	300	3379	456.7	3405	498728.4	3422	420.4	**3448**	3657.8	3414	275628.8
	350	**3609**	438.0	**3609**	388994.2	**3609**	83.4	**3609**	900.8	**3609**	75533.2
Set 130 ($\varrho = 2$)	50	543	41.6	548	10500.0	545	29.5	548	378.5	**567**	30201.4
	100	1330	150.1	**1368**	94527.1	1333	204.4	1352	2192.1	1357	150701.0
	150	1774	217.8	2115	207108.8	2087	475.8	**2118**	4112.8	**2118**	328684.4
	200	2441	304.8	2777	392091.4	2728	594.6	2730	3722.2	**2785**	340909.8
	250	3258	437.6	3296	398421.6	3297	433.4	3326	2777.4	**3343**	157301.4
	300	**3609**	444.6	**3609**	344742.6	3586	175.4	**3609**	787.0	**3609**	66542.7