ENERGY EFFICIENT MULTI-PLACE ROBOT RENDEZVOUS PROBLEM WITH CAMPAIGN TIME RESTRICTIONS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

NAZLI DOLU HASTÜRK

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
OPERATIONAL RESEARCH

JULY 2020

Approval of the thesis:

ENERGY EFFICIENT MULTI-PLACE ROBOT RENDEZVOUS PROBLEM WITH CAMPAIGN TIME RESTRICTIONS

submitted by **NAZLI DOLU HASTÜRK** in partial fulfillment of the requirements for the degree of **Master of Science in Operational Research Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** __________

Assoc. Prof. Dr. Cem İyigün
Head of Department, **Operational Research** __________

Assist. Prof. Dr. Mustafa Kemal Tural
Supervisor, **Industrial Engineering, METU** __________

Examining Committee Members:

Prof. Dr. Meral Azizoğlu
Industrial Engineering, METU __________

Assist. Prof. Dr. Mustafa Kemal Tural
Industrial Engineering, METU __________

Assoc. Prof. Dr. Cem İyigün
Industrial Engineering, METU __________

Assist. Prof. Dr. Sakine Batun
Industrial Engineering, METU __________

Assist. Prof. Dr. Diclehan Tezcaner Öztürk
Industrial Engineering, Hacettepe University __________

Date: 21.07.2020

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Nazlı Dolu Hastürk

Signature :

ABSTRACT

ENERGY EFFICIENT MULTI-PLACE ROBOT RENDEZVOUS PROBLEM WITH CAMPAIGN TIME RESTRICTIONS

Dolu Hastürk, Nazlı
M.S., Department of Operational Research
Supervisor: Assist. Prof. Dr. Mustafa Kemal Tural

July 2020, 73 pages

We study the energy efficient multi-place robot rendezvous problem. In this problem, we aim to find a set of rendezvous places where a tanker robot meets with mobile worker robots for a recharging task by preserving a meeting order. The problem is examined under two different objective functions. The first objective function is to minimize the total time spent, i.e., campaign time to recharge all the robots. The second objective function is to minimize the total energy consumption of all the robots by taking a predetermined campaign time as a restriction. The energy consumption functions of both the mobile worker robots and the tanker robot used in this study are nonlinear and distances between locations are calculated by the Euclidean distances. This problem is NP-hard when we aim to find the optimal rendezvous places and the optimal meeting order simultaneously. In our solution approach, we first fix the meeting order and determine the optimal rendezvous places based on a given meeting order. To do so, we provide a second order cone programming formulation. Then, we utilize improvement heuristics to find a better meeting order to improve the objective function value. Mainly we work on 2-opt and 3-opt edge exchange improvement heuristics as well as their combination to search for a better meeting order. Furthermore, we implement speed-up techniques to decrease the solution times of the improvement heuristics. Finally, extensive computational experiments are conducted to compare the suggested improvement heuristic algorithms and speed-up techniques.

Keywords: Energy Efficiency, Second Order Cone Programming, Mobile Robots, Rendezvous Problem

ÖZ

ENERJİ VERİMLİLİĞİ ESASLI ZAMAN KISITLI VE ÇOK KONUMLU ROBOT BULUŞMA PROBLEMİ

Dolu Hastürk, Nazlı
Yüksek Lisans, Yöneylem Araştırması Bölümü
Tez Yöneticisi: Dr. Öğr. Üyesi. Mustafa Kemal Tural

Temmuz 2020 , 73 sayfa

Bu çalışmada enerji verimliliğine dayalı çok konumlu robot buluşma problemi üzerinde durulmuştur. Tanker robotun gezici işçi robotları şarj etmek üzere buluştuğu ve bu robotlarla buluşma sırasını koruduğu varsayılarak optimal buluşma konumlarından oluşan kümenin bulunması amaçlandı. Problem için iki farklı amaç fonksiyonu tanımlandı. Birincisi, harcanan toplam zamanı en azlamak ve ikincisi ise harcanan toplam enerji tüketimini zaman kısıtı doğrultusunda en azlamak olarak tanımlandı. Bu problemde hem gezici işçi robotların hem de tanker robotun enerji tüketimini hesaplamak için doğrusal olmayan fonksiyonlar kullanılmıştır. Ayrıca iki nokta arasındaki uzaklık ölçütü olarak Öklid uzaklığı kullanılmıştır. Problem optimal buluşma konumlarının ve optimal buluşma sırasının aynı anda bulunması olarak düşünüldüğünde NP-Zor (NP-hard) bir problemdir. Biz bu problemi iki bölümde inceledik. Birinci bölümde kararlaştırılan bir buluşma sırasına göre optimal buluşma konumlarının bulunması, ikinci bölümde ise daha iyi bir buluşma sırasının bulunması üzerine çalışıldı. Birinci bölüm için ikinci dereceden konik programlama formülasyonu önerildi. İkinci bölüm için ise 2-opt ve 3-opt sezgisel kenar değişimi algoritmaları ve bu iki algoritmanın kombinasyonu kullanıldı. Ek olarak çözüm zamanlarını geliştirmek için hızlandırma teknikleri uygulandı. Son olarak bu algoritmalar kapsamlı hesaplama çalışmaları doğrultusunda kıyaslandı.

Anahtar Kelimeler: Enerji Verimliliği, İkinci Dereceden Konik Programlama, Gezici Robotlar, Buluşma Problemi

To my family

ACKNOWLEDGMENTS

I would like to express my gratitude to my supervisor Assist. Prof. Dr. Mustafa Kemal Tural for sharing his invaluable experiences and guidance throughout this thesis.

Furthermore, I am thankful for examining committee members of my thesis Prof. Dr. Meral Azizoğlu, Assoc. Prof. Dr. Cem İyigün, Assist. Prof. Dr. Sakine Batun, and Assist. Prof. Dr. Diclehan Tezcaner Öztürk for their valuable feedbacks.

Moreover, I am grateful to have the most caring and supportive people in the world as my parents, Hafize Dolu and Mustafa Dolu. Their invaluable thoughts and perspectives are my sole guiding light in this life. Also, I would like to thank to my beloved husband, Umur Hastürk, with whom I started this journey and went through its ups and downs together. Thank you for being there all the time, your kindness and support means a lot to me.

In addition, I would like to thank to Tuğçe Yalçın and Serkan Yalçın for cheering me up when I needed the most during the difficult times. I owe my sincere thanks to my dear friends, especially to Yağmur Caner for being an amazing support and for her precious friendship, also to Tuna Berk Kaya for being the most motivative friend. Lastly, I would like to thank to my friends and colleagues who contributed to this thesis directly or indirectly.

This study was supported by the Scientific and Technological Research Council of Turkey (TUBITAK).

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

Robotics is a growing field in nowadays world. It is a combination of many engineering and science disciplines such as electrical and electronics, mechanical engineering, and computer sciences. Robotics is mostly used in industrial manufacturing as robot arms and robotic manipulators. According to International Federation of Robotics (IFR), industrial robotics market is valued at $ 16.5 billion [1]. In an assembly line, a robot arm is able to conduct repetitive work at high speed and high precision, and in the electronics industry, a robotic manipulator is able to place a component with extremely high precision so that a computer can be manufactured. However, the most significant inability of these industrial robots is the fact that they cannot move [2]. To overcome this inability, mobile robots are created. A mobile robot can travel inside the manufacturing plant. Mobile robots can be utilized alongside humans. Furthermore, when a mobile robot is also autonomous, it can work in hostile or hazardous environments where humans cannot travel through [3].

Developments in autonomous mobile robotics prove that there is a wide range of application areas for autonomous mobile robots. To be more specific, mobile robots can operate in the following missions:

- border security [4],
- military missions (battlefield surveillance, payload delivery) [5], [6],
- forest fires or disaster-hit areas (searching for survivors) [7], [8],
- oil spill monitoring [9],
- wild-life population monitoring [10].

In these examples, most of the time, a group of autonomous mobile robots aim to achieve a mission. In this thesis, we study the energy efficient multi-place robot rendezvous problem (abbreviated as EEMPR), which can also be observed and applied to these missions. Assume that there are multiple mobile worker robots operating in an area for a specific purpose. Note that, the term "mobile worker robot" will refer to an "autonomous mobile robot" along the thesis. In long-term missions, mobile worker robots are required to be recharged while maintaining the mission. In our study, there is another type of mobile robot called the tanker robot whose only task is to recharge the mobile worker robots. Tanker robot meets with mobile worker robots to perform a recharging task with a meeting order. Tanker robot can meet with mobile worker robots at different places. These meeting places are called "rendezvous places." We also allow that tanker robot can meet with more than one mobile worker robot at the same place. However, the meeting order is always preserved. In other words, tanker robot should recharge the mobile worker robots with the meeting order even though more than one mobile worker robot meet at the same rendezvous place. We assume that each robot has a certain energy level, a known initial and final location. The tanker robot has sufficient energy and does not require to be recharged during the mission. Furthermore, each mobile worker robot consumes energy based on a non-linear energy consumption function. There is a maximum battery recharge level, i.e., battery capacity, for each mobile worker robot. Also, robots cannot operate more than a predetermined speed based on their design parameters, but we allow them to adjust their own speeds.

The main aim of EEMPR is to find the optimal rendezvous places as well as optimal meeting order simultaneously. The Euclidean Traveling Salesman Problem (TSP), which is a special case of EEMPR, is known as NP-Hard. Therefore, EEMPR is also NP-Hard. We analyze EEMPR in two parts. First, we fix the meeting order and under a given meeting order, we provide a second order cone programming (SOCP) formulation. Later, we use improvement heuristics to find a better meeting order as means of the objective function value by solving the SOCP for each improved meeting order. In this context, we utilize the widely used 2-opt and 3-opt improvement heuristics. The problem is examined under two different objective functions. We first study the problem which minimizes the total time spent during the recharging, i.e., campaign time (abbreviated as EEMPR-T). Then, we analyze minimizing the total energy consumption of robots by taking the campaign time as a restriction (abbreviated as EEMPR-E).

The outline of this thesis is as follows. In Chapter 2, related literature review is provided. In Chapter 3, the notation is introduced and the problem description is given for the EEMPR when the meeting order is fixed. In Chapter 4, details of the solution methods proposed for EEMPR when the meeting order is fixed as well as improvement heuristics algorithms for developing a better meeting order are discussed. In Chapter 5, the computational experiments are provided which compares the algorithms in terms of solution quality and computational time. Finally, Chapter 6 concludes the study with some future research directions.

CHAPTER 2

LITERATURE REVIEW

The rendezvous problem is first introduced by Alpern in [11]. Later, it is discussed in detail by Alpern and Gal in [12]. The authors studied how two agents can meet when they are randomly placed in an area while minimizing the time required to rendezvous. They examine this problem on lines, circles, and polygons, etc. and draw some conclusions. In [13], Roy and Dudek study this rendezvous problem by combining it with the multi-agent robotics. The authors worked on a problem where two mobile worker robots are exploring an unknown environment and should meet at a predefined time at a single rendezvous place to share information. Later, Meghjani and Dudek studied rendezvous problem when more than two mobile worker robots are exploring an environment that is defined as a random graph [14].

The energy limitation is one of the most significant challenges a mobile robot encounters. Although the rendezvous problem is originated by aiming to minimize the total time spent while achieving rendezvous, another objective, which is minimizing the energy consumption of the robots is seen in many studies, see [15], [16].

In long-term missions, multiple mobile worker robots operating in an area are required to be recharged while maintaining the mission. The recharging can be made by taking the mobile worker robot offline and connecting it to a power unit by a human. However, this is not always possible when the mobile worker robots are utilized for hazardous sites where human intervention is impossible. For these cases, Silverman et al. presented a solution by creating a stationary recharging station to implement autonomous recharging [17]. This stationary recharging station is modeled so that the mobile worker robot can dock to the station to recharge. The authors provide a stationary recharging station design along with a docking station and a docking mechanism for the mobile worker robot. Another approach developed to solve the recharging problem that occurred in long-term missions is provided by Zebrowski et al. [18]. In this article, the authors proposed creating an exceptional mobile robot called the tanker robot whose only task is to recharge the mobile worker robots operating in an area. They provided the design details of the tanker robot in the article.

Overall, the rendezvous problem can be examined in two categories: single place and multi-place rendezvous. In the single place rendezvous problem, the aim is to find a single location where all the mobile worker robots meet. Recharging on a stationary station, maintenance activities, or collection can be the reasons for meeting at a single rendezvous place. In [16], Zebrowski et al., aim to minimize the energy consumption of the mobile worker robots while traveling towards the rendezvous place by assuming that the energy consumption of a mobile worker robot is linearly proportional with the distances they traveled. The authors proposed a heuristic method where each mobile worker robot iteratively computes where to head based on the initial locations of the other mobile worker robots. In [19], Lanthier et al. worked on finding a rendezvous place for recharging in a weighted graph, which minimizes the maximum of the mobile worker robot travel costs. The cost can be interpreted as distance, time or energy. They proposed a heuristic method for a given meeting order in which a mobile worker robot uses the locations of its predecessor, successor, and itself to go towards the single rendezvous place.

In the multi-place rendezvous problem, mobile worker robots meet with each other at different locations to share information or with a tanker robot to be recharged. In [20], Litus et al. studied a multi-place rendezvous problem where the mobile worker robots meet with the tanker robot at different places based on a meeting order. The aim is to minimize the total cost of travel of the robots where the individual travel costs are measured by weighted Euclidean distances. They proposed a heuristic algorithm which should be run by each mobile worker robot individually. The proposed method is able to find approximate solutions that are close to the global solution.

In this study, we consider EEMPR by utilizing the tanker approach. EEMPR has been first introduced by Litus et al. in [15]. The authors aim to find the set of rendezvous places where the mobile worker robots meet with the tanker robot by minimizing the energy consumed while traveling towards the rendezvous places. They propose three methods to find the rendezvous places under a given meeting order. First of all, they assume a discrete location case where the tanker robot can only meet with the mobile worker robots at a fixed set of locations with arbitrary travel costs. For this, they propose a solution method based on recurrence. Second, they consider continuous location case and offer two iterative methods, one is based on the Weiszfeld's algorithm, and the other is based on the Newton's algorithm. These methods find approximate solutions. Also, in the continuous case, they assume that the energy consumption is calculated based on weighted Euclidean distances traveled. For each algorithm, computational studies are made by using five mobile worker robots. They also prove that finding the optimal meeting order is NP-hard and improvement heuristics can be used to improve the meeting order but no further algorithmic details or computational studies are given. Our study differentiates from the other studies in the literature in the following ways. We propose a solution method that can find an optimal set of rendezvous places for a given meeting order by utilizing a non-linear energy consumption function. Also, robots cannot operate more than a predetermined speed based on their design parameters, but we allow them to choose their own speeds. We implement improvement heuristics to find a better meeting order, which improves the objective function value. In addition, we conduct detailed computational experiments on these algorithms. Furthermore, in our computational studies, it is realized that while utilizing improvement heuristics, computational times increase rapidly when the number of mobile worker robots increases. Hence, for cases where 30 or more mobile worker robots are used, we provide speed-up algorithms to decrease the solution times. Moreover, for EEMPR, we utilize two objective functions: to minimize the total time spent during the mission, i.e., campaign time, and to minimize the total energy consumption of the robots under a campaign time constraint.

CHAPTER 3

PROBLEM DESCRIPTION AND NOTATION

In this chapter, we provide the problem description and notation for both versions of EEMPR when the meeting order is taken as given.

We are given n mobile worker robots working in an area, and the environment is obstacle free. We consider that the tanker robot is meeting with each mobile worker robot $i \in I = \{1, 2, \ldots, n\}$ based on a given meeting order to perform a recharging task. The initial and the final locations of the mobile worker robots are known and indicated by a_i and b_i, respectively, for $i \in I = \{1, 2, \ldots, n\}$. Also, the initial and final location of the tanker robot are also known and represented by, a_T and b_T, respectively.

A mobile worker robot starts its movement from its initial location, meets with the tanker robot at a rendezvous place for recharging and then proceed to its final location. Tanker robot starts the movement from a known initial location, meets with each mobile worker robot at rendezvous places based on the meeting order. After meeting with the mobile worker robot having the last place in the meeting order, it goes to its final location. Each mobile worker robot has an initial energy level E_i^0 before starting its movement for the rendezvous place. In addition, tanker robot has an initial energy level T_0. It is assumed that the tanker robot has sufficient energy and does not require to be recharged during the mission. Each worker robot has a maximum energy storage level, i.e., battery capacity, stated with D_i. Also, robots cannot operate more than a predetermined speed based on their design parameters which is represented by v_i^{max} and v_T^{max} for mobile worker robots and the tanker robot, respectively. Furthermore, robots, i.e., mobile worker robots and the tanker robot, can have different energy consumption functions and the battery recharge functions. We measure the distance between two locations by the Euclidean distance.

In Figure 3.1, an illustrative example is presented. In Figure 3.1a, mobile worker robots and the tanker robot are located at their initial locations with a battery sign showing their initial energy levels. In the battery sign, we have minimum and target energy levels. We assume that the energy levels of the mobile worker robots cannot drop under the minimum energy level. Furthermore, when they finalize their movement at their final locations, the remaining energies should be at least at the target level to continue their operations. Note that, these minimum and target level assumptions are not valid for the tanker robot. In the illustrative example, the meeting order is $2-1-3-4$. In Figures 3.1a and 3.1f, the rendezvous places are marked based on the mobile worker robot number to show with which mobile worker robot the tanker robot is meeting. For instance, the rendezvous place $1', 3'$ means that mobile worker robots 1 and 3 are meeting with the tanker robot at the same location. Moreover, for simplicity, it is assumed that robots return back to their initial locations as their final locations. In Figure 3.1b, tanker robot meets with mobile worker robot 2 at rendezvous place $2'$. Mobile worker robots 1 and 3 has also moved towards the rendezvous place $1', 3'$, while mobile worker robot 4 is waiting at its initial location. Note that, all the robots which are moving consumed some energy which can be observed from the decrease in their remaining energy levels. In Figure 3.1c, mobile worker robot 2 has been recharged which can be observed from the increase in its remaining energy level, and moves towards its final location while the tanker robot meets with mobile worker robots 1 and 3. However, the meeting order should be preserved. So, In Figure 3.1c, mobile worker robot 1 is recharged while mobile worker robot 3 is waiting for its turn. Moreover, mobile worker robot 4 has started its movement towards its rendezvous place $4'$. In Figure 3.1d, mobile worker robot 2 has reached to its final location while mobile worker robot 1 is moving towards its final location and mobile worker robot 3 has been recharged. Also, the mobile worker robot 4 is getting closer to its rendezvous place $4'$. In Figure 3.1e, mobile worker robot 1 has reached to its final location while mobile worker robot 3 is moving towards its final location. Moreover, tanker robot has met with mobile worker robot 4. In the end, in Figure 3.1f, robots can be observed in their final locations with their final energy levels.

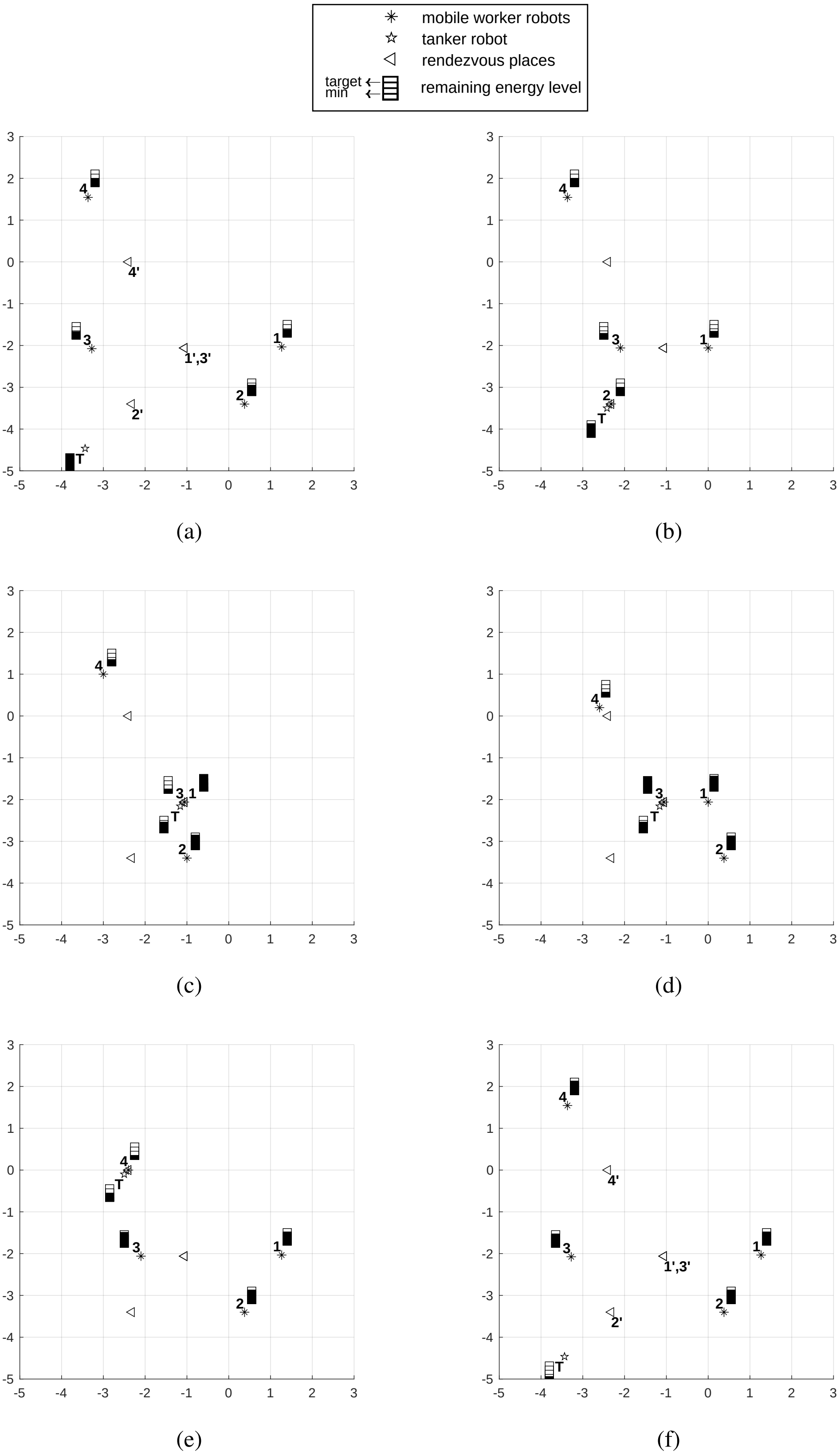


Figure 3.1: Illustrative example where the meeting order is $2-1-3-4$

For this study, we utilize the energy consumption function provided by Tokekar et al. in [21] to calculate the energy consumption of robots. In 3.1, E is the energy consumption in Joules for a robot which travels d meters with speed v $\mathrm{m/s}$ and α, β and, γ are function parameters.

$$E = \alpha d v + \beta d + \gamma \frac{d}{v} \tag{3.1}$$

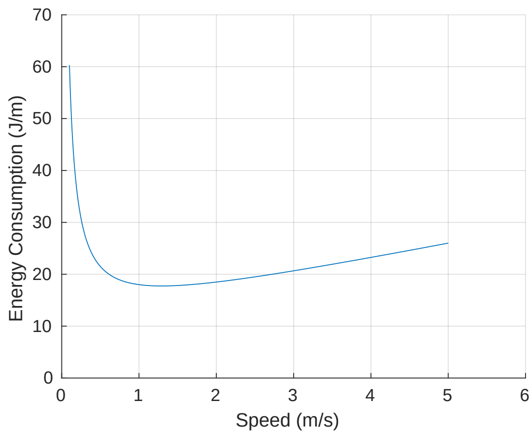


Figure 3.2: Energy consumption in Joule per meter as a function of the mobile robot speed

Figure 3.1 shows the energy consumption rate in Joules per meter with respect to speed in $\mathrm{m/s}$ according to Equation 3.1 and obtained by using the parameters as $\alpha = 3$, $\beta = 10$ and, $\gamma = 5$. In Figure 3.2, it can be observed that below 1.291 $\mathrm{m/s}$, energy consumption per unit distance for a robot increases with a decrease in speed. Furthermore, above 1.291 $\mathrm{m/s}$, an increase in speed increases the energy consumption per unit distance for a robot. Hence, 1.291 $\mathrm{m/s}$ is the optimal speed in which the energy consumption of a robot is the smallest. We represent function parameters as α_i, β_i, γ_i for mobile worker robots and α_T, β_T, γ_T for the tanker robot.

Moreover, the battery recharge function of a mobile worker robot is considered as in Equation 3.2 in which c indicates the total battery recharging time in seconds where the battery is recharged up to the energy level E^{final}. $E^{residual}$ represents the remaining energy right before the recharging activity starts and σ is the function parameter. We represent function parameter as σ_i for mobile worker robots.

$$c = (E^{final} - E^{residual})\sigma \tag{3.2}$$

For (EEMPR-E), we have mentioned that there is a campaign time restriction. We have taken this restriction from the decision maker and store it as t^{camp}. Also, we assume that mobile worker robots require some remaining energy when they go to their final locations. In order to indicate this, we define θ_i and say that $\theta_i D_i$ amount of energy should be left as a remaining energy level.

Note that, in the following mathematical formulations, some of the parameters of the mobile worker robots are required to be rearranged. We rename the mobile worker robots and assume that the meeting order is $1 - 2 - ... - n$. For instance, in the meeting order $2 - 1 - 3 - 4$, the mobile worker robot having position 1 in the meeting order is mobile worker robot 2. So, the initial location a_1 should indicate the initial location of mobile worker robot 2. Therefore, before giving parameters $a_i, b_i, E_i^0, D_i, \sigma_i, \theta_i, \alpha_i, \beta_i, \gamma_i$ to the mathematical formulation, we rearrange them according to the current meeting order.

Given these information, we formulated the mathematical model for our problem. Parameters and decision variables are summarized as follows:

Parameters:

- α_i, β_i, γ_i: energy consumption function parameters of mobile worker robot having position i in the meeting order
- α_T, β_T, γ_T: energy consumption function parameters of the tanker robot
- σ_i: battery recharge function parameter of mobile worker robot having position i in the meeting order
- v_i^{max}: the maximum speed $(\mathrm{m/s})$ with which the mobile worker robot having position i in the meeting order can function
- v_T^{max}: the maximum speed $(\mathrm{m/s})$ with which the tanker robot can function
- E_i^0: initial energy level of the mobile worker robot having position i in the meeting order
- D_i: the maximum energy storage level, i.e., battery capacity, of mobile worker robot having position i in the meeting order

- T_0: initial energy level of the tanker robot
- a_i: initial location of the mobile worker robot having position i in the meeting order
- b_i: final location of the mobile worker robot having position i in the meeting order
- a_T: initial location of the tanker robot
- b_T: final location of the tanker robot
- θ_i: coefficient to indicate remaining energy level requirement in the final location for the mobile worker robot having position i in the meeting order
- t^{camp}: campaign time that the user wants the rendezvous task to last

Decision variables:

- x_i: rendezvous place where the tanker robot meets with the mobile worker robot having position i in the meeting order
- t_i: time spent during the movement of mobile worker robot having position i in the meeting order from its initial location to its rendezvous place
- t_i^{end}: time spent during the movement of mobile worker robot having position i in the meeting order from rendezvous place to its final location
- $\bar{t}_i$: time spent during the movement of the tanker robot from rendezvous place with worker robot having position $i-1$ in the meeting order to rendezvous place with mobile worker robot having meeting order i
- d_i: distance traveled during the movement of mobile worker robot having position i in the meeting order from its initial location to its rendezvous place
- d_i^{end}: distance traveled during the movement of mobile worker robot having position i in the meeting order from rendezvous place to its final location
- $\bar{d}_i$: distance traveled during the movement of the tanker robot from rendezvous place with mobile worker robot having position $i-1$ in the meeting order to

rendezvous location with mobile worker robot having position i in the meeting order

- s_i: start time of the battery recharge mission of mobile worker robot having position i in the meeting order
- c_i: time spent during the battery recharge of mobile worker robot having position i in the meeting order
- s^{end}: total duration of the rendezvous task, i.e., campaign time
- E_i^m: remaining energy level of the mobile worker robot having position i in the meeting order after its movement from its initial location to the rendezvous place
- E_i^f: remaining energy level of the mobile worker robot having position i in the meeting order after its movement from the rendezvous place to its final location
- E_i^{max}: the energy level of the mobile worker robot having position i in the meeting order right after recharging
- T_i: remaining energy level of the tanker robot after its movement from rendezvous place with mobile worker robot having position $i-1$ in the meeting order to rendezvous place with mobile worker robot having position i in the meeting order

According to these, (EEMPR-E) can be mathematically formulated as follows:

$$\text{minimize} \quad \sum_{i \in I} (E_i^0 - E_i^m + E_i^{max} - E_i^f) + (T_0 - T_{n+1}) \qquad \text{(EEMPR-E)}$$

subject to

$$s_i \geq s_{i-1} + c_{i-1} + \bar{t}_i \qquad \forall i \in I \quad s_0, c_0 = 0 \qquad (3.3)$$

$$s_i \geq t_i \qquad \forall i \in I \qquad (3.4)$$

$$c_i \geq (E_i^{max} - E_i^m)\sigma_i \qquad \forall i \in I \qquad (3.5)$$

$$d_i \geq \|a_i - x_i\| \qquad \forall i \in I \qquad (3.6)$$

$$d_i^{end} \geq \|x_i - b_i\| \qquad \forall i \in I \qquad (3.7)$$

$$\overline{d_i} \geq \|x_i - x_{i-1}\| \qquad \forall i \in I \cup \{n+1\} \qquad (3.8)$$

$$x_0 = a_T \tag{3.9}$$

$$x_{n+1} = b_T \tag{3.10}$$

$$E_i^0 - E_i^m \geq \alpha_i \frac{{d_i}^2}{t_i} + \beta_i d_i + \gamma_i t_i \qquad \forall i \in I \tag{3.11}$$

$$E_i^{max} - E_i^f \geq \alpha_i \frac{(d_i^{end})^2}{t_i^{end}} + \beta_i d_i^{end} + \gamma_i t_i^{end} \qquad \forall i \in I \tag{3.12}$$

$$T_{i-1} - T_i \geq \alpha_T \frac{\overline{d}_i^2}{\overline{t}_i} + \beta_T \overline{d}_i + \gamma_T \overline{t}_i \qquad \forall i \in I \cup \{n+1\} \tag{3.13}$$

$$E_i^f \geq \theta_i D_i \qquad \forall i \in I \tag{3.14}$$

$$E_i^{max} \leq D_i \qquad \forall i \in I \tag{3.15}$$

$$E_i^{max} \geq E_i^m \qquad \forall i \in I \tag{3.16}$$

$$d_i \leq t_i v_i^{max} \qquad \forall i \in I \tag{3.17}$$

$$d_i^{end} \leq t_i^{end} v_i^{max} \qquad \forall i \in I \tag{3.18}$$

$$\overline{d}_i \leq \overline{t}_i v_T^{max} \qquad \forall i \in I \tag{3.19}$$

$$s^{end} \geq s_i + c_i + t_i^{end} \qquad \forall i \in I \tag{3.20}$$

$$s^{end} \geq s_n + c_n + \overline{t}_{n+1} \tag{3.21}$$

$$s_i,\, c_i,\, t_i,\, d_i,\, t_i^{end},\, d_i^{end}, E_i^m, E_i^f, E_i^{max} \geq 0 \qquad \forall i \in I \tag{3.22}$$

$$\overline{t}_i,\, T_i,\, \overline{d}_i \geq 0 \qquad \forall i \in I \cup \{n+1\} \tag{3.23}$$

$$s^{end} \geq 0 \tag{3.24}$$

$$s^{end} \leq t^{camp} \tag{3.25}$$

In the formulation, the objective function minimizes the total energy consumption of all the mobile worker robots and the tanker robot. The $\sum_{i \in I}(E_i^0 - E_i^m)$ part of the objective function represents the energy consumption of all the mobile worker robots while moving from their initial locations to their rendezvous places. The $\sum_{i \in I}(E_i^{max} - E_i^f)$ part is for calculating the energy consumptions of mobile worker robots while going from their rendezvous places towards their final locations. Also, $(T_0 - T_{n+1})$ presents the energy consumption of the tanker robot after all its movements since T_0 is the initial energy of the tanker robot while T_{n+1} is the remaining energy level in its final location. All these components together represents the objective function.

Furthermore, constraint 3.3 is to make sure that the recharging task for mobile worker robot having position i in meeting order cannot start before the recharging task of its predecessor is finished and the arrival of the tanker robot to the rendezvous place x_i. Constraint 3.4 ensures the recharging task of mobile worker robot having position i in meeting order cannot start before it arrives to the rendezvous place x_i.

Constraint 3.5 is defined based on the battery recharge function stated in Equation 3.2. Constraints 3.6, 3.7 and 3.8 represent that the distances are measured by Euclidean distances. Constraints 3.9 indicates that tanker robot departs from its initial location and 3.10 shows that it ends its movement at predefined final location.

Constraints 3.11, 3.12 and 3.13 are constructed based on Equation 3.1. For these constraints, the equation is adjusted by using $v = d/t$ equality as follows:

$$E = \alpha d\frac{d}{t} + \beta d + \gamma\frac{d}{\frac{d}{t}}$$

Constraint 3.11 represents the energy consumption of the mobile worker robot having position i in meeting order when moving towards the rendezvous place while constraint 3.12 indicates the energy consumption of the mobile worker robot having position i in meeting order when going from its rendezvous place to its final location. Constraint 3.13 indicates the energy consumption of the tanker robot during its activities.

After recharged, a mobile worker robot is assumed to maintain working in the field and constraint 3.14 makes sure that it preserves θ_i times of its maximum energy storage level when it goes to its final location. Constraint 3.15 indicates that after recharging, a mobile worker robot cannot have an energy level more than its battery capacity. Also, due to constraint 3.5, constraint 3.16 is required to correctly decide on E_i^{max}.

According to their design parameters, there is a maximum speed that robots can operate at. Constraints 3.17, 3.18 are to specify that mobile worker robot having position i in meeting order cannot move with a speed more than v_i^{max} m/s while constraint 3.19 indicates this restriction for the tanker robot.

Constraints 3.20 and 3.21 indicate that the campaign time is either bounded with the time when all the mobile worker robots are recharged and traveled to their final locations or to the time that the tanker robot is finished recharging the last mobile worker robot and traveled to its final location. Furthermore, constraint 3.25 reflects the campaign time restriction where t^{camp} is the predefined time in which the rendezvous task should be finalized within.

The minimization of the campaign time objective function, (EEMPR-T), can be formulated as follows by adjusting the mathematical formulation of (EEMPR-E):

$$\begin{aligned} &\text{minimize} \quad s^{end} && \text{(EEMPR-T)} \\ &\text{subject to} \\ &\qquad 3.3 - 3.24 \end{aligned}$$

Note that, the objective function of the mathematical formulation is changed. Now, it minimizes the campaign time which is represented by the decision variable s^{end}. In addition, constraint 3.25 is discarded out from the formulation, because while utilizing (EEMPR-T), we cannot have a campaign time restriction.

Moreover, when minimizing campaign time, if time is sufficient to finalize the rendezvous mission, robots may move slower or faster than the optimal speed which would cause an increase in the energy consumption value. In this case there may be lots of alternative optimal solutions. Hence, we can modify the objective function as follows:

$$\text{minimize} \quad s^{end} + \varepsilon \sum_{i \in I} (E_i^0 - E_i^m + E_i^{max} - E_i^f) + (T_0 - T_{n+1})$$

In this modification, we add the energy consumption multiplied with an ε value to the objective function. This may decrease the number of alternative solutions.

In both (EEMPR-E) and (EEMPR-T), we measure the distance between two points by the Euclidean distance. Also, the energy consumption function defined in Equation 3.1 is non-linear. Therefore, the problem we described appeared as non-linear. In order to overcome the non-linearity, we aimed to formulate both versions of the problem as second order cone programs.

In the next chapter, we first introduce second order cone programming (SOCP), then we describe an SOCP formulation for (EEMPR-E) and (EEMPR-T). Later, we examine the 2-opt and 3-opt improvement heuristics to improve the given meeting order.

CHAPTER 4

SOLUTION METHODS

4.1 SOCP Formulation

An SOCP problem is a convex optimization problem which has a linear objective function and some second order cone constraints as well as linear contraints [22]. Mathematically, an SOCP problem is a problem of the following form:

$$\begin{aligned} &\text{minimize} \;\; h^T x \\ &\text{subject to} \;\; \|A_f x + b_f\| \leq c_f^T x + e_f, \quad f = 1, 2, \ldots, m \end{aligned} \tag{4.1}$$

where $x \in \mathbb{R}^{\mathrm{n}}$ is the vector of decision variables, and $h \in \mathbb{R}^{\mathrm{n}}, \mathrm{A_f} \in \mathbb{R}^{\mathrm{n_f} \times \mathrm{n}}, \mathrm{b_f} \in \mathbb{R}^{\mathrm{n_f}}, \mathrm{c_f} \in \mathbb{R}^{\mathrm{n}}, \mathrm{e_f} \in \mathbb{R}$ are the problem parameters. The constraints in 4.1 are called as the second order cone constraints. SOCP problem can be solved in polynomial time. The reader is referred to [22] and [23] for more detailed information.

Note that, except the constraints 3.6, 3.7, 3.8 which compute the Euclidean distances and constraints 3.11, 3.12 and 3.13 which represent energy consumptions, all other constraints are linear for both (EEMPR-E) and (EEMPR-T). Although constraints indicating the Euclidean distances are non-linear, they are in the form of 4.1 and hence are SOCP constraints. Therefore, if we can convert energy consumption constraints into SOCP constraints, we can end up with an SOCP formulation for both versions of the problem.

So, we only examine constraints 3.11, 3.12 and 3.13 to show that these can be converted into SOCP constraints. For the conversion we need to introduce new decision variables. Let us consider constraint 3.11 as an example and split it into two constraints with the help of a new variable $f_i, \; \forall i \in I$. Then we will have,

(a) $f_i \leq E_i^0 - E_i^m - \beta_i d_i - \gamma_i t_i$,

(b) $\alpha_i \frac{{d_i}^2}{t_i} \leq f_i$, and

(c) $f_i \geq 0$.

Notice that now (a) and (c) are linear. We need to convert (b) to obtain an SOCP formulation. (b) is equivalent to

$$\alpha_i {d_i}^2 \leq f_i t_i$$

and this can be written as

$$\left\| \begin{pmatrix} \sqrt{\alpha_i} d_i \\ \frac{(t_i - f_i)}{2} \end{pmatrix} \right\| \leq \frac{(t_i + f_i)}{2}$$

which is an SOCP constraint. Hence, we discard constraint 3.11 out and add the following constraints to the mathematical formulation:

(3.11a) $f_i \leq E_i^0 - E_i^m - \beta_i d_i - \gamma_i t_i \qquad \forall i \in I$

(3.11b) $\left\| \begin{pmatrix} \sqrt{\alpha_i} d_i \\ \frac{(t_i - f_i)}{2} \end{pmatrix} \right\| \leq \frac{(t_i + f_i)}{2} \qquad \forall i \in I$

(3.11c) $f_i \geq 0 \qquad \forall i \in I.$

Constraints 3.12 and 3.13 can be rewritten in the same manner with the new decision variables g_i and h_i, respectively. Therefore, by discarding constraint 3.12 out and adding the following constraints,

(3.12a) $g_i \leq E_i^{max} - E_i^f - \beta_i d_i^{end} - \gamma_i t_i^{end} \qquad \forall i \in I$

(3.12b) $\left\| \begin{pmatrix} \sqrt{\alpha_i} d_i^{end} \\ \frac{(t_i^{end} - g_i)}{2} \end{pmatrix} \right\| \leq \frac{(t_i^{end} + g_i)}{2} \qquad \forall i \in I$

(3.12c) $g_i \geq 0 \qquad \forall i \in I$

and, by discarding constraint 3.13 out and adding the following constraints,

(3.13a) $h_i \leq T_{i-1} - T_i - \beta_T \overline{d}_i - \gamma_T \overline{t}_i \qquad \forall i \in I$

(3.13b) $\left\| \begin{pmatrix} \sqrt{\alpha_T} \overline{d}_i \\ \frac{(\overline{t}_i - h_i)}{2} \end{pmatrix} \right\| \leq \frac{(\overline{t}_i + h_i)}{2} \qquad \forall i \in I$

$$(3.13c) \quad h_i \geq 0 \qquad \forall i \in I$$

we end up with an SOCP formulation.

Now, we are able to solve EEMPR by an SOCP formulation for a given meeting order. If we would like to solve the problem to the optimal, we can implement the following optimizing procedure in Algorithm 1.

Algorithm 1 Optimizing procedure for EEMPR

1: Initialize a set P by determining all n! meeting orders.
2: Solve the SOCP formulation for each element of P, and store their objective function values in set P_{obj}.
3: Find min(P_{obj}).
4: Return min(P_{obj}) as the optimal objective function value and its corresponding meeting order as the optimal meeting order.

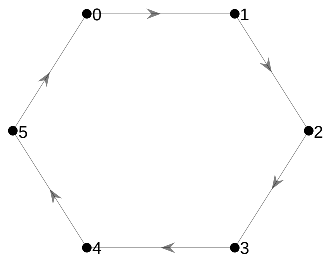
This optimizing procedure is still NP-hard. Hence, to be able to find a better meeting order as means of the objective function value, we utilize improvement heuristics.

4.2 Improvement Heuristics

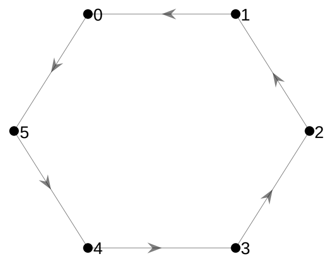
Improvement heuristics are utilized to search for an enhanced solution. Node insertion, edge insertion, k-opt edge exchange heuristics are some of the traveling salesman problem (TSP) improvement heuristics. The reader is referred to [24] for more information. In this study, we work on TSP 2-opt and 3-opt edge exchange heuristics to improve a meeting order. The idea of using TSP heuristics for EEMPR can be found in [15] but authors do not provide any algorithmic detail or perform any computational experiment.

As mentioned before, the meeting order $1-2-3-4-5$ indicates that the tanker robot is first meeting with mobile worker robot 1, then 2 so on and so forth. These meeting orders can represent tours as in TSP if we represent them as a complete tour. To do so, at the beginning of each meeting order, we add 0 to identify the tanker robot. Now, we can define a tour for the meeting order $1-2-3-4-5$ as in Figure 4.1a. Note that, although now we see $n+1$ many nodes in the representation, there are still

n many mobile worker robots in the meeting order since we do not count the tanker robot as a mobile worker robot. Realize that the term "meeting order" is used to show the meeting sequence of the mobile worker robots while the term "tour" is used when node 0 is added at the beginning of a meeting order.



(a) Tour 0-1-2-3-4-5 for order 1-2-3-4-5.



(b) Reverse tour 0-5-4-3-2-1 for order 1-2-3-4-5.

Figure 4.1: Example tour and reverse tour

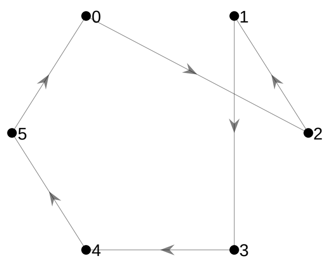
In EEMPR, it can be realized that a reverse meeting order would most probably end up with a different solution as in an asymmetric TSP, see [25]. Therefore, when we consider improving the meeting orders, we also examine the reverse tour of a current meeting order. For instance, for the meeting order example in Figure 4.1, SOCP formulations will also be solved for the reverse tour in Figure 4.1b.

4.2.1 2-opt Edge Exchange Heuristic

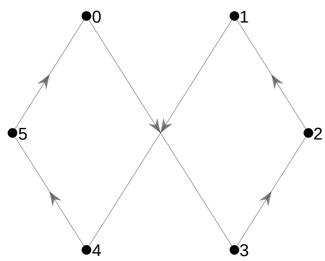
In 2-opt edge exchange heuristic, we break two edges of a tour which are not adjacent. Then we create two new edges to generate a new tour. This creation of the new tour is called as a 2-opt move. Note that, when two edges are broken, there is precisely one way to join them to create a legit tour. For instance, when we break edges $0 - 1$ and $2 - 3$ in the tour illustrated in Figure 4.1a, we can only connect these two edges as stated in 4.2a to generate a new tour. If we try to connect node 0 to node 3 rather than node 2, we end up with two sub tours as $0 - 3 - 4 - 5$ and $1 - 2$ which is not desired. Also, connecting node 0 to node 1 and node 2 to node 3 would end up with the initial tour we specified.

In a meeting order having n mobile worker robots, there are in total $\binom{n+1}{2} - (n + 1)$

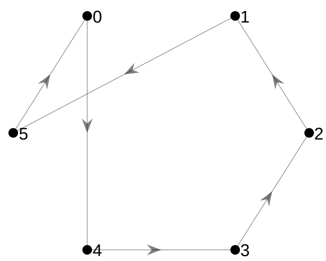
many 2-opt moves which can end up with legit tours. $\binom{n+1}{2}$ represents the number of different ways that two edges can be broken. Then we subtract $(n+1)$ since no two consecutive edges can be broken. For instance, when we choose $0-1$ as one of the edges for 2-opt move, we can only choose edges $2-3$, $3-4$ and $4-5$, see Figure 4.2.



(a) New tour: 0-2-1-3-4-5



(b) New tour: 0-3-2-1-4-5



(c) New tour: 0-4-3-2-1-5

Figure 4.2: The illustrations of the 2-opt moves

In order to implement 2-opt moves, we use the following 2-opt move function stated in Algorithm 2. The 2-opt move function takes a meeting order and the objective function value of the SOCP formulation as inputs. Then it checks all $\binom{n+1}{2}-(n+1)$ many 2-opt moves and their resulting meeting orders. Among these, the function returns back the meeting order which generates the maximum improvement. If there is no improvement, the function returns back the initial meeting order and the objective function value which were given as the input. Note that, to solve the SOCP formulation, the function also require parameters of the (EEMPR-E) or (EEMPR-T) as an input such as initial locations, initial energy levels etc. These are not indicated in Algorithm 2, but we assume that the required parameters are given to the function as inputs.

Algorithm 2 2-opt move function ($2opt$)

Inputs: a meeting order $p_{current}$ and its objective function value $obj_{current}$.
Initialize a set P by determining all possible meeting orders of $p_{current}$ by using 2-opt moves. Also, include their reverse orders.
Solve the SOCP formulation for each element of P, and store their objective function values in set P_{obj}.
if min$(P_{obj}) < obj_{current}$ **then**
  Find the meeting order, say $p_{neworder}$, in P whose objective function value equals min(P_{obj}).
  Update $p_{current} := p_{neworder}$.
  Update $obj_{current} :=$ min(P_{obj}).
end if
Return $p_{current}$ and $obj_{current}$.

Then, we demonstrate the 2-opt algorithm in Algorithm 3. In this algorithm, the 2-opt move function is executed with each improving meeting order until no improvement in the objective function value is observed.

For clarification purposes, a flowchart is provided in Figure 4.3 which represents how improvement heuristics work by taking 2-opt algorithm as an example.

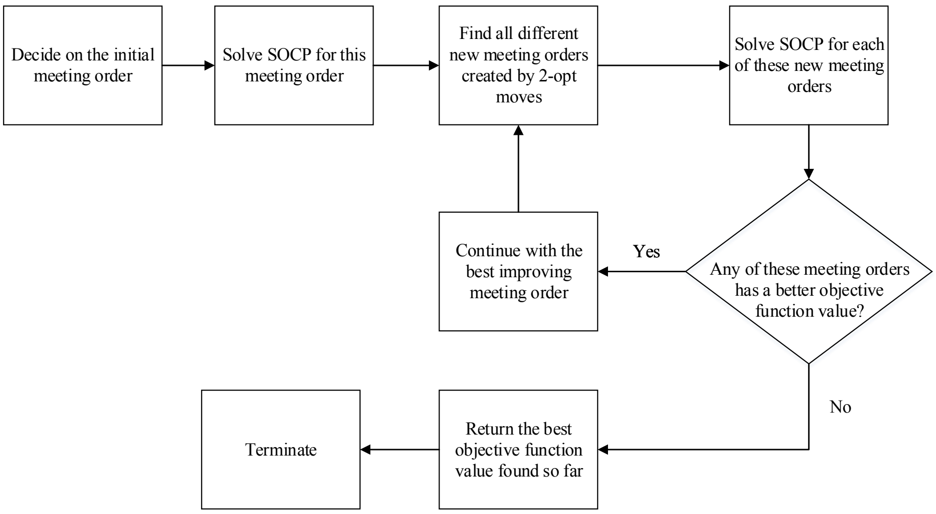


Figure 4.3: Flowchart for 2-opt algorithm

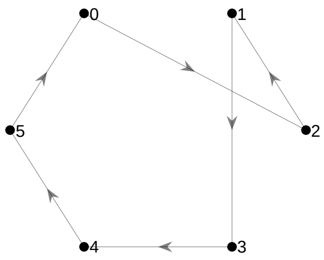
Algorithm 3 2-opt algorithm

1: Start with a meeting order $p_{initial}$.
2: Solve the SOCP formulation with $p_{initial}$ and calculate the objective function value as $obj_{initial}$.
3: Set $p_{current} \coloneqq p_{initial}$
4: Set $obj_{current} \coloneqq obj_{initial}$.
5: **repeat**
6: Call 2-opt move function stated in Algorithm 2 as $[p_{current}, obj_{current}] \coloneqq 2opt(p_{current}, obj_{current})$
7: **until** No improvement is observed.
8: Return $p_{current}$ and $obj_{current}$ as the best meeting order found and its objective function value, respectively.

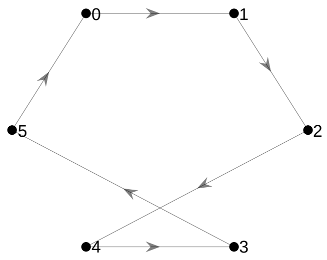
4.2.2 3-opt Edge Exchange Heuristic

In 3-opt edge exchange heuristic, we break three edges which are not adjacent to each other. Then we connect them in a way that we generate a new tour. Note that, when three non-adjacent edges are broken, there are seven different ways to join these edges to generate a new tour. For example, let us assume we break edges $0-1$, $2-3$ and $4-5$ in the tour illustrated in 4.1a. We can connect these edges in three different ways as in Figure 4.4. One can realize that these are actually 2-opt moves. In addition to these, we can also join them in another four different ways, represented in Figure 4.5 in which all three edges generated are new edges. We will call the latter four different ways as pure 3-opt moves. Hence, when these three 2-opt moves and four pure 3-opt moves combined are called 3-opt moves.

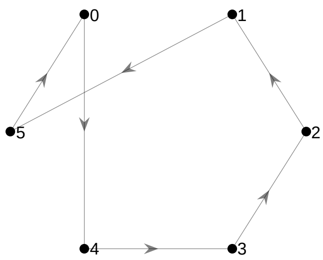
Although, we stated that if we want to implement a 3-opt move, no three consecutive edges can be chosen, when two of the edges are adjacent and the third one is not adjacent to these two, there is precisely one way to reconnect these edges to generate a legit tour, see [26]. For instance, in Figure 4.6, we illustrated possible tours when edge $0-1$ is chosen along with an adjacent and a non-adjacent edge. We will call this as 3-opt adjacent move. Hence, from now on the term pure 3-opt moves will also include 3-opt adjacent moves.



(a) New tour: 0-2-1-3-4-5



(b) New tour: 0-3-2-1-4-5



(c) New tour: 0-4-3-2-1-5

Figure 4.4: The illustrations of the 2-opt moves

In a meeting order having n mobile worker robots, there are in total $[\binom{n+1}{3} - (n+1) - (n+1)(n+1-4)](4) + (n+1)(n+1-4)$ pure 3-opt moves which can end up with legit tours. $\binom{n+1}{3}$ represents the number of different ways that three edges can be broken. Then, we subtract $(n+1)$ since no three consecutive edges can be broken. In addition, we subtract $(n+1)(n+1-4)$ since no two edges can be consecutive as well. As it was stated before, there are four different ways to connect these three non-adjacent edges, so we multiply this number with four. Moreover, we should also add 3-opt adjacent moves. In total, there are $(n+1)(n+1-4)$ ways to choose a two adjacent and one non-adjacent edge.

Note that, as mentioned before, 3-opt moves composed of 2-opt and pure 3-opt moves. In order to implement 3-opt moves, we use the following function stated in Algorithm 4. The function takes a meeting order and its objective function value as inputs. Then it checks all $\binom{n+1}{2} - (n+1) + [\binom{n+1}{3} - (n+1) - (n+1)(n+1-4)](4) + (n+1)(n+1-4)$ many 3-opt moves and their resulting meeting orders. Among these, the function returns back the meeting order which generates the maximum improvement. If there is no improvement, the function returns back the initial meeting order and its

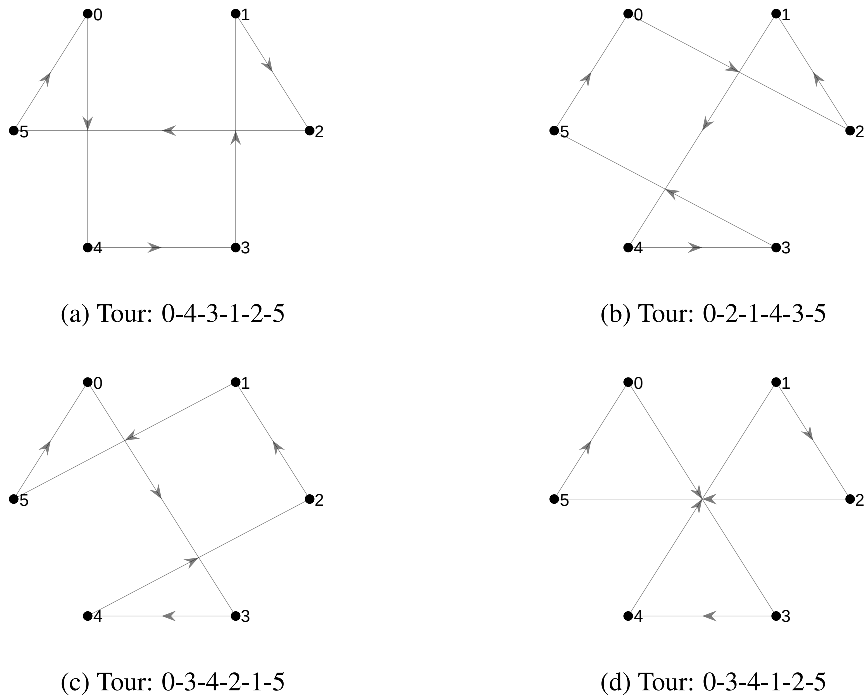


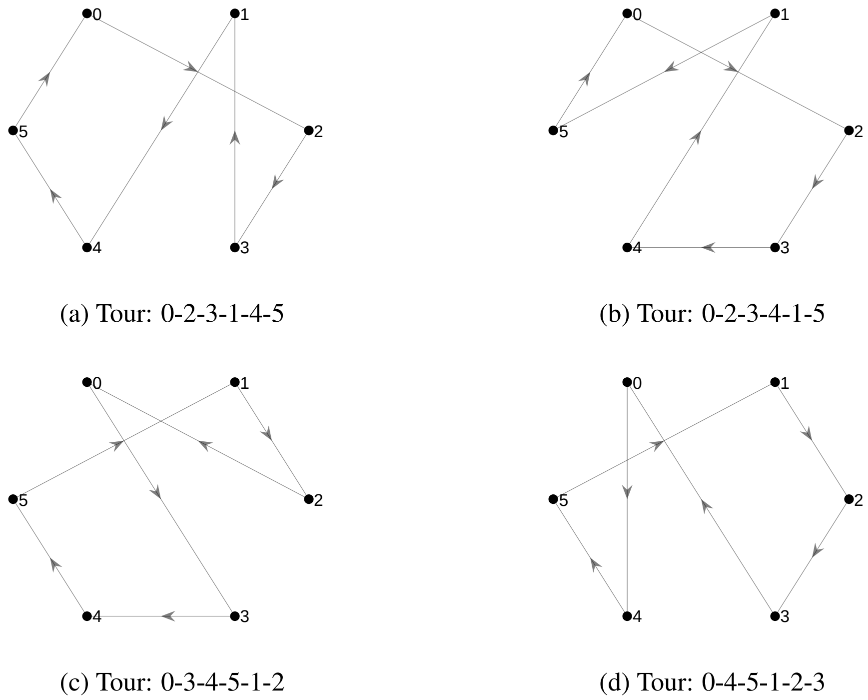
Figure 4.5: The illustrations of pure 3-opt moves

objective function value which were given as an input. Note that, to solve the SOCP formulation, the functions also require parameters of the (EEMPR-E) or (EEMPR-T) as an input such as initial locations, initial energy levels etc. These inputs are not indicated in the Algorithm 4, but we assume that the required parameters are given to the function as inputs.

Then, we display the 3-opt algorithm in Algorithm 5. In this algorithm, the 3-opt move function is executed with each improving meeting order until no improvement in the objective function value is observed.

4.2.3 Combination of 2-opt and 3-opt Edge Exchange Heuristics

The combined algorithm can be found in Algorithm 6. In this algorithm, we first utilize the 2-opt move function to make all the improvements which can be made by 2-opt moves until there is no improvement. Then, the output is given as an input to the 3-opt move function to check all the 3-opt moves until there is no improvement. Note that, when improvements made by 2-opt function is finalized at line 7, we should call



(a) Tour: 0-2-3-1-4-5

(b) Tour: 0-2-3-4-1-5

(c) Tour: 0-3-4-5-1-2

(d) Tour: 0-4-5-1-2-3

Figure 4.6: The illustrations of the 3-opt adjacent moves when $n = 5$ and edge $0 - 1$ is chosen along with one adjacent and one non-adjacent edge

a function which only checks pure 3-opt moves. At this step, if an improving move is found, the algorithm continues executing line 8. If no improvement is observed, then the algorithm stops by returning $p_{current}$ and $obj_{current}$ as outputs by going to line 11.

4.2.4 Speed-up Algorithms

When the number of mobile worker robots operating in a field increases, the time required to find an improved solution may increase undesirably. To overcome this issue, we analyzed speed-up techniques created for TSP edge exchange heuristics.

In a TSP 2-opt move, if both of the newly created edges increases in length, the length of the new tour cannot be decreased. Based on this observation, in [27], Bentley suggest the fixed-radius search to speed-up 2-opt edge exchange heuristic. The search is implemented by visiting the vertices of a tour. For a vertex v_a, consider both of its adjacent vertices as v_b in the given tour. If v_b is not already the nearest neighbor of v_a, then we search around v_a for v_c where $\|v_a - v_c\| \leq \|v_a - v_b\|$. To do so, we

Algorithm 4 3-opt move function ($3opt$)

Inputs: a meeting order $p_{current}$ and its objective function value $obj_{current}$.
Initialize a set P by determining all possible meeting orders of $p_{current}$ by using 3-opt moves. Also, include their reverse orders.
Solve the SOCP formulation for each element of P, and store their objective function values in set P_{obj}.
if min$(P_{obj}) < obj_{current}$ **then**
    Find the meeting order, say $p_{neworder}$, in P whose objective function value equals min(P_{obj}).
    Update $p_{current} \coloneqq p_{neworder}$.
    Update $obj_{current} \coloneqq$ min(P_{obj}).
end if
Return $p_{current}$ and $obj_{current}$.

define a radius r which is equal to $\|v_a - v_b\|$ and place a ball centered at v_a having radius r. The vertices within the ball are candidates to be chosen as v_c. Realize that, v_c has only one appropriate vertex to be deleted, say v_d, to achieve a 2-opt move. Hence, 2-opt move is generated by deleting edges (v_a, v_b) and (v_c, v_d), and adding edges (v_a, v_c) and (v_b, v_d). Bentley suggests that the first such improving 2-opt move is applied to the tour and the search continues from the new tour. However, in the 2-opt move function we defined in Algorithm 2, we solve the problem for all 2-opt moves and find the best improving move. Hence, while implementing fixed radius search, we continue with this approach. We solve the problem for every candidate vertex to find the one which improves the objective function value the best and set it as v_c rather than choosing the first improving 2-opt move. Furthermore, Bentley also provides the idea to extend the fixed-radius search to 3-opt edge exchange heuristic. For this, two searches are required to be applied. The first search is the 2-opt search defined above to find vertex v_c. Later, a second search is generated by centering a ball at v_c with radius r' which is equal to $\|v_a - v_b\| + \|v_c - v_d\| - \|v_a - v_c\|$. The vertices within the ball are candidates to be chosen as v_e. Realize that, v_e has only one appropriate vertex to be deleted, say v_f, to achieve a pure 3-opt move. For each v_e, we define the appropriate neighbor vertex v_f. Hence, a pure 3-opt move is applied to the current tour by deleting edges (v_a, v_b), (v_c, v_d) and (v_e, v_f), and adding edges

Algorithm 5 3-opt algorithm

1: Start with a meeting order $p_{initial}$.
2: Solve the SOCP formulation with $p_{initial}$ and calculate the objective function value as $obj_{initial}$.
3: Set $p_{current} \coloneqq p_{initial}$
4: Set $obj_{current} \coloneqq obj_{initial}$.
5: **repeat**
6: Call 3-opt move function stated in Algorithm 4 as $[p_{current}, obj_{current}] \coloneqq 3opt(p_{current}, obj_{current})$.
7: **until** No improvement is observed.
8: Return $p_{current}$ and $obj_{current}$ as the best meeting order found and its objective function value, respectively.

(v_a, v_c), (v_b, v_f) and (v_d, v_e).

The fixed radius search depends on the edge lengths between the vertices. However, in EEMPR, there is no specific edge length description. Hence, we consider the distances between the rendezvous places of the mobile worker robots. However, since at the start of the tour there is no rendezvous place specified yet, we take the initial location of the tanker robot into account. For instance, in the tour $0-1-2-3-4-5$ the edge length between nodes 0 and 1 is computed as $\|a_T - x_1\|$ and the edge length between nodes 1 and 2 is computed as $\|x_1 - x_2\|$ so on and so forth. For the edge length between nodes 5 and 0, the distance is computed based on the final location of the tanker and appeared as $\|x_5 - b_T\|$. One can realize that all these distances should be recalculated when the current tour is changed with an improving tour.

Moreover, in [28], Hoos and Thomas state that combining the fixed radius search with candidate list and don't look bits approaches can increase the search speed even more. In candidate list approach, we do not examine all the candidate vertices while choosing v_c and v_e but only examine the ones in the candidate list. For instance, let us say that we set the candidate list length as 2 while choosing v_c. Then, to create the candidate list, we list the candidate vertices which fall into the ball (centered at v_a having radius r) in descending order of their proximity to the v_a. We solve the problem for only the first 2 vertices in the candidate list. If there are less candidate

Algorithm 6 Combined 2-opt and 3-opt algorithm

Start with a meeting order $p_{initial}$.
Solve the SOCP formulation with $p_{initial}$ and calculate the objective function value as $obj_{initial}$.
Set $p_{current} := p_{initial}$
Set $obj_{current} := obj_{initial}$.
repeat
  Call 2-opt move function stated in Algorithm 2 as $[p_{current}, obj_{current}] := 2opt(p_{current}, obj_{current})$.
until No improvement is observed.
repeat
  Call 3-opt move function stated in Algorithm 4 as $[p_{current}, obj_{current}] := 3opt(p_{current}, obj_{current})$.
until No improvement is observed.
Return $p_{current}$ and $obj_{current}$ as the best meeting order found by the combination of 2-opt and 3-opt algorithms and its objective function value, respectively.

vertices in the candidate list than the length of the list, then the search is terminated when the list is fully examined.

In addition, according to Hood and Thomas [28], don't look bits approach is based on the following observation. If no improving 2-opt or 3-opt move is found for a vertex v_a in a given search step of fixed radius search, there is a slight chance that an improving move will be found in future search steps, unless one of the edges incident to the v_a changes. To implement don't look bits approach, the authors suggest to assign a Don't Look Bit (DLB) to each vertex in the tour. At the start of the fixed radius search, all DLBs should be turned off, i.e., set to zero. Later, if no improving move is found for a vertex in a given fixed radius search step, then the DLB of this vertex is turned on, i.e., set to one, at the end of the search step. The vertices whose DLBs are turned on are not examined in future search steps unless one of their incident edges changes. If this change is observed, then the DLB of the corresponding vertex is turned off again at the end of the search step.

As an example, combined 2-opt and 3-opt speed-up algorithm is shown in Algorithm

Algorithm 7 A combined 2-opt and 3-opt speed-up algorithm

1: Take the candidate list length as an input and set it as n_{cl}.
2: Start with a meeting order $p_{initial}$.
3: Solve the SOCP formulation with $p_{initial}$ and calculate the objective function value as $obj_{initial}$.
4: Set $p_{current} := p_{initial}$.
5: Set $obj_{current} := obj_{initial}$.
6: Initialize a DLB for each vertex.
7: Turn off all DLBs.
8: **repeat**
9: Call 2-opt move speed-up function which is created by modifying Algorithm 2 as $[p_{current}, obj_{current}, DLBs] := 2opt_{speedup}(n_{cl}, p_{current}, obj_{current}, DLBs)$.
10: **until** No improvement is observed.
11: Turn off all DLBs.
12: **repeat**
13: Call 3-opt move speed-up function which is created by modifying Algorithm 4 as $[p_{current}, obj_{current}, DLBs] := 3opt_{speedup}(n_{cl}, p_{current}, obj_{current}, DLBs)$.
14: **until** No improvement is observed.
15: Return $p_{current}$ and $obj_{current}$ as the best meeting order found by the combined 2-opt and 3-opt speed-up algorithm and its objective function value, respectively.

7. To perform speed-up techniques, the 2-opt and 3-opt move functions are modified according to the above mentioned speed-up techniques. These functions are then called in the corresponding lines 9 and 13 of Algorithm 7 by also providing the decided candidate list length and DLBs as inputs. Realize that in this algorithm, a fixed radius search step indicates looking at all the 2-opt moves or 3-opt moves. In other words, one execution of line 9 or 13 is a fixed radius search step. Hence, an update for DLBs is made at the end of each execution of lines 9 or 13.

CHAPTER 5

COMPUTATIONAL STUDIES

In this chapter, computational studies are discussed. We created random problem instances to test the SOCP formulations and improvement heuristics as there is no available benchmark instances. We use GUROBI 9.0.2 with its default parameters through C++ API (Visual Studio 2019, v142) to solve the SOCP formulation and improvement heuristics. For the instance creation, MATLAB R2018a is utilized. The computations were performed with Intel Core i7-4770S CPU @3.10 GHz and 16.00 GB RAM.

We generated random problem instances. Instance generation is defined in Section 5.1. Later, in Section 5.2 we discuss preliminary experiments. In detail, preliminary experiments are analyzed for the penalty approach, improvement heuristics and speed-up algorithms in Section 5.2.1, Section 5.2.2, and Section 5.2.3, respectively. In the end, in Section 5.3, the results of extensive computational studies are provided.

5.1 Instance Generation

During our experiments, we realized that some of the problem instances are not feasible even though complete enumeration for the meeting order of the mobile worker robots is conducted. Due to this, we would like to generate instances in which we ensure feasibility. To do so, we thought backwards and assumed that we know each rendezvous place by randomly generating each x_i. Then, an initial location of the mobile worker robot i is randomly generated within a distance to x_i. The initial location of the tanker robot is also randomly generated. Later, we measure the energy consumption of the mobile worker robot i while moving towards the rendezvous place

x_i from its initial location. Also, the energy consumption of the tanker robot is calculated as if it is visiting each x_i based on the specified meeting order in the instance generation. In the end, we increase these energy consumption values at a rate to preserve the feasibility. Until now the idea behind the instance generation is briefly given. Now, we will provide more detail about how we implemented these ideas.

We assume that rendezvous places are created in a way that they form a circle. The circle is approximated with 100 points. It has a radius $\sqrt{n}$ and center the origin. The rendezvous places divide the circle into n equal parts, see Figure 5.1. The numbers represent which mobile worker robot will meet with tanker robot in this rendezvous place.

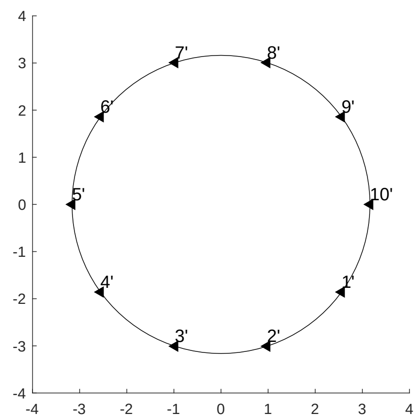


Figure 5.1: Rendezvous place generation to create a feasible instance when $n = 10$

Later, to generate initial locations, a_is, of the mobile worker robots, n smaller circles are created. We create an a_i, uniformly at random inside or on the circle having center as the rendezvous place i', see Figure 5.3a. These circles are approximated with 100 points. Note that, deciding on the radius of these smaller circles is significant for the randomness of the initial locations. To illustrate, when the radius is chosen as $\sqrt{n}/3$ for $n = 10$, we end up with Figure 5.2. Hence, after preliminary experiments, the radius of the smaller circles is taken as $\sqrt{n}/1.5$. In the end, when we remove the circles, we end up with the initial locations of the mobile worker robots, see Figure 5.3b.

In addition, the initial location of the tanker robot, a_T, is assigned as follows. After determining the initial locations of the mobile worker robots, the minimum and

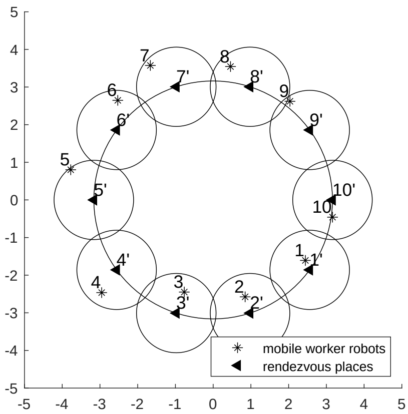
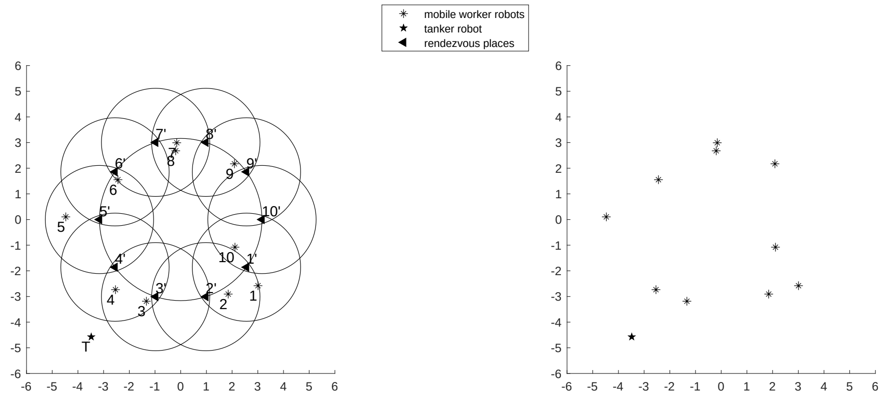


Figure 5.2: Initial location creation for worker robots when $n = 10$ with circles having radius $\sqrt{10}/3$

maximum x and y coordinate values of the initial locations, say x_{min}, y_{min} and x_{max}, y_{max} are observed. Later, x and y coordinate values of the initial location of the tanker robot are assumed to be randomly generated within the following ranges, respectively: $[x_{min} - \frac{x_{max} - x_{min}}{2}, x_{max} + \frac{x_{max} - x_{min}}{2}]$, $[y_{min} - \frac{y_{max} - y_{min}}{2}, y_{max} + \frac{y_{max} - y_{min}}{2}]$. In Figure 5.4, you can see that this range is specified by the continuous lines while dashed lines are drawn to indicate the minimum and maximum x and y coordinate values. The point having label T is indicating the initial location of the tanker robot which is randomly created in the specified range. Note that, probability of the initial location of the tanker robot is created within the area of dashed lines is 0.25 and within the area between the continuous lines and the dashed lines is 0.75.

For other parameter selections and the calculations, the following assumptions are made. All mobile worker robots are homogeneous. In other words, the energy consumption and the battery recharge function parameters are taken as the same as $\alpha = 3$, $\beta = 10$, $\gamma = 5$ and $\sigma = 0.005$ for each mobile worker robot. However, the tanker robot consumes two times as much energy as a mobile worker robot. Hence, the parameters are taken as 2α, 2β and 2γ in the energy consumption function of the tanker robot. Also, at rendezvous place i', mobile worker robot i recharges its battery up to its maximum energy storage level. This means that E^{max} is assumed to be equal to D. The maximum energy storage level D is taken as the same for mobile worker robots created in instances having the same n value. It may differ when n changes. The final locations of both the mobile worker robots and the tanker robot are assumed to be the



(a) Creation of initial location of mobile worker robots with smaller circles

(b) Generated initial locations of mobile worker robots

Figure 5.3: Initial location creation for worker robots when $n = 10$ with circles having radius $\sqrt{10}/1.5$

same as their initial locations, i.e., $a_i = b_i$ and hence $d_i = d_i^{end}$, $\forall i \in I$ and $a_T = b_T$. Both mobile worker robots and the tanker robot are assumed to be operating at the optimal speed, i.e., $v_{opt} = 1.291$, for the initial energy calculations. One unit is taken as $1\ km$ and mobile worker robots are working in a $10\ km \times 10\ km$ area. Mobile worker robot i and tanker robot meet at specified rendezvous place i', see Figure 5.3a.

In the instance creation, it is considered that the tanker robot first meets with the mobile worker robot whose initial location is the closest to the tanker robot. Then tanker robot continues moving in the clockwise direction to meet with the other mobile worker robots. For example, in Figure 5.3a, mobile worker robot 4 is the closest to the tanker robot. Hence, tanker robot meets with the mobile worker robots in the following meeting order $4 - 5 - 6 - 7 - 8 - 9 - 10 - 1 - 2 - 3$. In order to make sure that the instances are feasible, 2 times more energy is assumed to be loaded to the mobile worker robots while calculated feasible energy for the tanker robot is multiplied by 1.5. Therefore, we calculated the initial energy level of the mobile worker robot i as follows; $E_i^{initial} = 3(\alpha d_i v_{opt} + \beta d_i + \gamma \frac{d_i}{v_{opt}})$ where $d_i = 1000(\|a_i - i'\|)$. We multiply the distance with 1000 since a unit is assumed as $1\ km$. Furthermore, the initial energy of the tanker robot is calculated as follows, $T_0 = 1.5(2\alpha d_{sum} v_{opt} + 2\beta d_{sum} + 2\gamma \frac{d_{sum}}{v_{opt}}))$ where d_{sum} is the total distance trav-

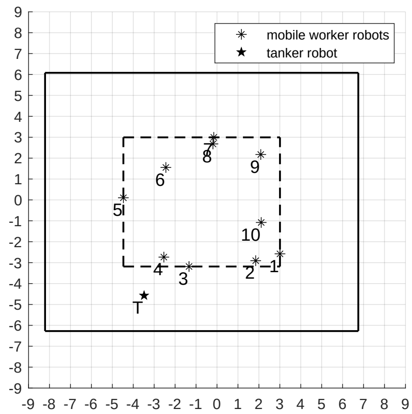


Figure 5.4: Initial location creation of the tanker robot when $n = 10$

eled by the tanker robot. If we consider the instance in Figure 5.3a, then $\overline{d}_{sum} = 1000(\|a_T - 4'\| + \|4' - 5'\| + ... + \|2' - 3'\| + \|3' - a_T\|)$.

In order to determine the maximum energy storage level of mobile worker robots, D, all instances are created for a specific n value and initial energy levels of the mobile worker robots of each of these instances are calculated. Later, D is set to the maximum initial energy level among all the mobile worker robots in all the instances created for a specific n value.

In utilization of the (EEMPR-E), it can be realized that if constraint 3.25 is not binding, then both mobile worker robots and the tanker robot travel at the optimal speed. Hence, generation of the campaign time parameter, t^{camp}, is significant. In order to calculate a t^{camp} value, we first roughly estimated s^{end} of the instances generated. To do so, we calculate t_i as the time that mobile worker robot having position i in the meeting order spends on traveling from its initial location, a_i, to the rendezvous place i'. Since the final location is the same as the initial location and $t_i = t_i^{end}$ equality is also utilized. Moreover, for the tanker robot, we calculated total travel time as follows:

$$s^{end} = max\{\max_{\{i \in I\}}\{s_i + c_i + t_i^{end}\}, s_n + c_n + \overline{t}_{n+1}\} \ \ where \tag{5.1}$$

$$s_i = \max_{\{i \in I\}}\{s_{i-1} + c_{i-1} + \overline{t}_i, t_i\} \ \ and \ \ s_0, c_0 = 0$$

Equation 5.1 can clearly be converted into Equation 5.2;

$$s^{end} = max\{\max_{\{i \in I\}}\{s_i + c_i + \frac{d_i}{v_{opt}}\}, s_n + c_n + \frac{\overline{d}_{n+1}}{v_{opt}}\} \ \ where \tag{5.2}$$

$$s_i = \max_{\{i \in I\}} \{s_{i-1} + c_{i-1} + \frac{\overline{d}_i}{v_{opt}}, \frac{d_i}{v_{opt}}\} \quad and \quad s_0, c_0 = 0$$

Equation 5.2 also requires an approximate value for battery recharge time, c_i, of the mobile worker robot having position i in the meeting order. To be able to approximate c_i, Equation 5.3 is used based on the battery recharge function specified as 3.2,

$$\begin{aligned} c_i &= (E^{max} - E_i^{residual})\sigma \quad where \\ E_i^{residual} &= E_i^{initial} - (\alpha d_i v_{opt} + \beta d_i + \gamma \frac{d_i}{v_{opt}}) \end{aligned} \tag{5.3}$$

Note that, $E_i^{residual}$ is calculated by subtracting the energy consumed while traveling towards the rendezvous place i' from the initial energy level $E_i^{initial}$. Then, we determine t^{camp} of an instance based on its approximated s^{end} value. If t^{camp} is taken as s^{end} then again most of the time constraint 3.25 would be binding. After preliminary experiments, it is decided that multiplying s^{end} value found by Equation 5.2 by 0.9 to have parameter t^{camp} would be limiting enough.

Furthermore, the instances we create may still not be feasible for some meeting orders. We can eliminate infeasibility by letting the model be able to increase the initial energy level of the tanker robot. In this case, even if mobile worker robots are restricted due to their low initial energy levels in some of the instances, tanker robot can meet with each of them at their initial locations. Therefore, a new constraint stated as 5.4 is constructed by defining two new decision variables, T_0^{new} and e.

$$T_0^{new} \leq T_0 + e \tag{5.4}$$

Note that, after adding constraint 5.4 to both (EEMPR-E) and (EEMPR-T), constraints which use T_0 should be updated with T_0^{new}, e.g., constraint 3.13. We will then penalize e with a big M in both of the objective functions. The modified objective functions of (EEMPR-T) and (EEMPR-E) can be seen in 5.5 and in 5.6, respectively.

$$s^{end} + M(e) \tag{5.5}$$

$$\sum_{i \in I}(E_i^0 - E_i^m + E_i^{max} - E_i^f) + (T_0 - T_{n+1}) + M(e + e^{camp}) \tag{5.6}$$

Furthermore, when we utilize (EEMPR-E), an infeasibility may be caused due to the campaign time restriction. So, another decision variable, e^{camp}, is defined and added to the campaign time restriction constraint, see 5.7. In other words, constraint 3.25 should be changed with 5.7 in (EEMPR-E). Then, e^{camp} is also penalized with a big M value in the objective function of (EEMPR-T), see 5.6.

$$s^{end} \leq t^{camp} + e^{camp} \tag{5.7}$$

5.2 Preliminary Experiments

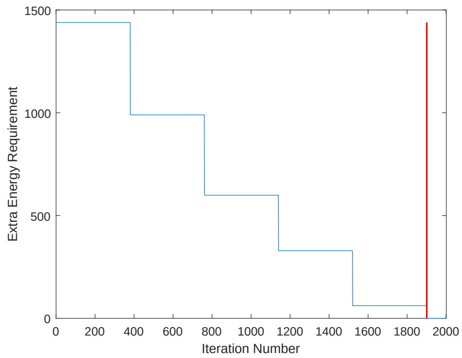
In this section, preliminary experiments are analyzed. In Section 5.2.1, Section 5.2.2, and Section 5.2.3, preliminary experiments are discussed for the penalty approach, improvement heuristics, and speed-up algorithms, respectively.

5.2.1 Preliminary Experiments for the Penalty Approach

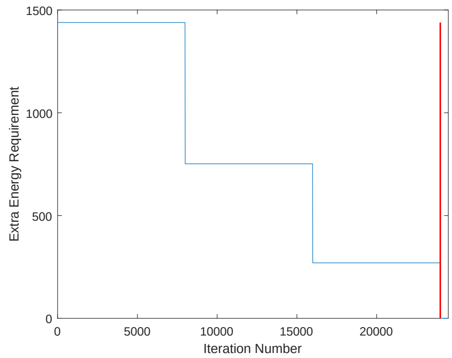
In the preliminary experiments, we have first examined the penalty approach discussed above. For this we utilize (EEMPR-E). Note that, in the instance we used to analyze penalty approach, t^{camp} is taken large enough to make e^{camp} equal to 0. So, we only see the effect of penalizing e.

We analyzed 2-opt, 3-opt and combined 2-opt and 3-opt algorithms which can be seen in Figures 5.5a, 5.5b and 5.5c, respectively. In these Figures, x-axis shows the iteration number. Each iteration number indicates one 2-opt or 3-opt move or its reverse order while y-axis refers to extra energy requirement, i.e., e, for the best meeting order found so far. In Chapter 4, it was discussed that the 2-opt and 3-opt functions return back the meeting order which generates the maximum improvement. Hence, each decrease observed in the extra energy requirement value caused by the best meeting order found by one execution of 2-opt or 3-opt function.

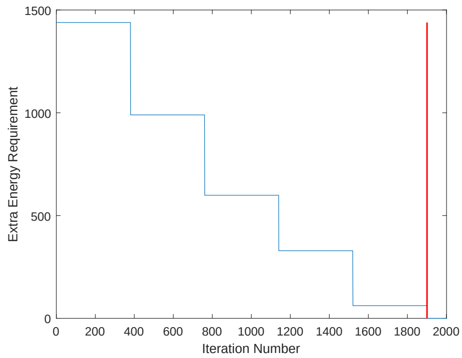
The vertical lines are drawn to the iteration number where the algorithms find a feasible solution for the first time. Iterations which are made before the vertical lines are infeasible since e has a value greater than 0. In Figure 5.6, the remaining parts of these improvements are shown. Note that, Figures 5.6a, 5.6b and 5.6c starts from the iteration number in which we observe the first feasible solution. Therefore, now y-axis shows the corresponding objective function values of the best meeting orders



(a) 2-opt algorithm



(b) 3-opt algorithm

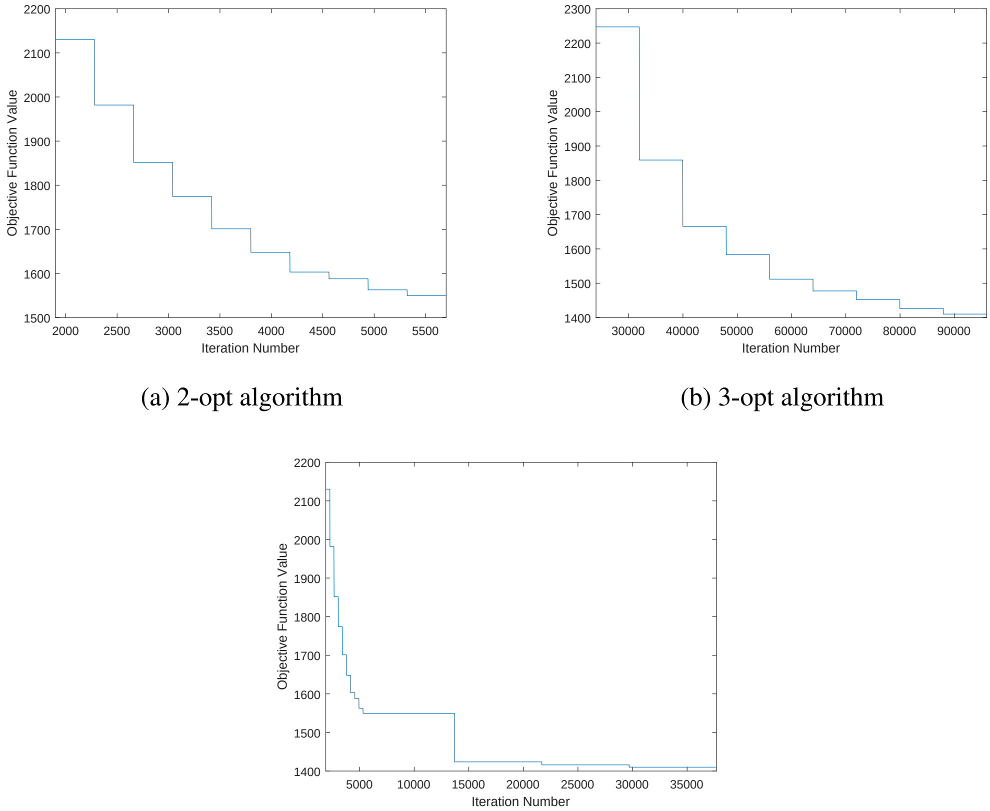


(c) Combined 2-opt and 3-opt algorithm

Figure 5.5: The illustrations of improvements in the extra energy requirements

found so far. One should realize that, as expected, improvement heuristics enhance a given meeting order as means of objective function value. Therefore, the given initial meeting order is improved and at the end we observe a better meeting order. This is observed throughout all the algorithms considered in computational experiments. Initial meeting orders given to algorithms and the final meeting orders found by algorithms are different than each other in every case for an instance.

Furthermore, for (EEMPR-E), we wanted to observe the effect of t^{camp} on energy consumption value and speed. To do so, we start a large enough t^{camp} value, i.e., $e^{camp} = 0$. Then we slowly decrease t^{camp} value to the point in which the instance cannot become feasible. At the times in which $e^{camp} = 0$, we observe that the mobile worker robots and the tanker robot operates with optimal speed which can be seen in Figure 5.7. In this figure, the dots represent different solutions for different t^{camp} values. When t^{camp} becomes tighter and tighter, the energy consumption value and



(a) 2-opt algorithm

(b) 3-opt algorithm

(c) Combined 2-opt and 3-opt algorithm

Figure 5.6: The illustrations of the improvements in the objective function values

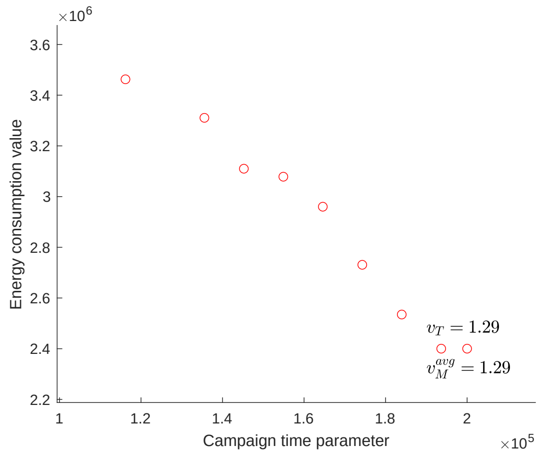
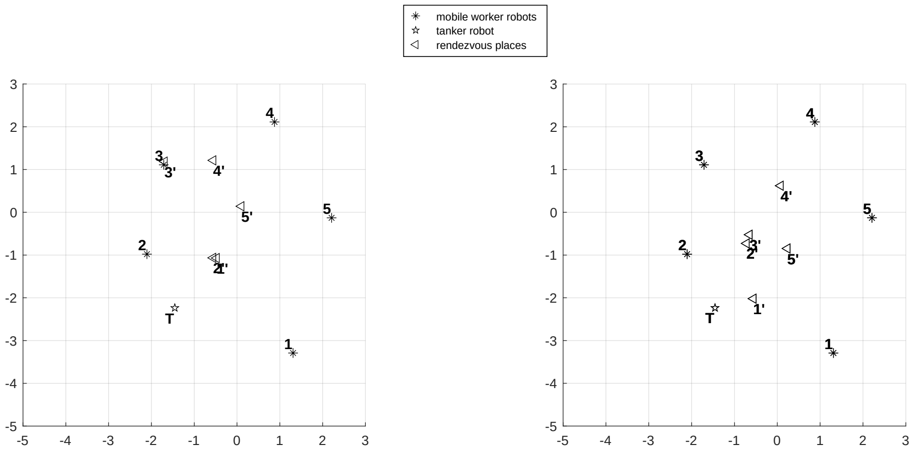


Figure 5.7: Effect of t^{camp} on speed and energy consumption value for (EEMPR-E) when $n = 30$ where $v_T = [4.87, 4.36, 3.68, 3.57, 3.18, 2.40, 1.74, 1.29, 1.29]$ and $v_M^{avg} = [4.18, 3.77, 3.22, 3.14, 2.81, 2.19, 1.66, 1.29, 1.29]$



(a) (b)

Figure 5.8: Different optimal rendezvous places when $n = 5$

the average speed of mobile worker robots, i.e., v_M^{avg}, as well as the speed of the tanker robot, i.e., v_T, increases. The arrays in the description of Figure 5.7 represents the v_T and v_M^{avg} values for the solutions represented in the figure accordingly. In order to illustrate the change of optimal rendezvous places, we use an instance when $n = 5$. Different optimal rendezvous places can be observed when t^{camp} is restrictive and not restrictive in Figures5.8a and 5.8b, respectively. Note that in Figure 5.8, mobile worker robots and the tanker robot are illustrated in their initial locations.

5.2.2 Preliminary Experiments for Improvement Heuristics

First of all, we generated 10 different instances randomly for $n = 8$. We solved these instances with the optimizing procedure given in Algorithm 1 in Chapter 4. We mainly utilize complete enumeration on meeting orders for both versions of the problem to find the optimal objective function values. The results can be seen in Table 5.1. Later, we utilize improvement heuristics algorithms with nine random meeting order starts and one Nearest Neighbor (NN) meeting order start for each instance. The Nearest Neighbor meeting order start is calculated based on the proximity of the mobile worker robots to the tanker robot by taking their initial locations into account. The mobile worker robot having the closest initial location to the initial location of

Table 5.1: Complete enumeration solutions for (EEMPR-T) and (EEMPR-E) when $n = 8$

Instance	EEMPR-T		EEMPR-E	
No	Best obj. value	Total time	Best obj. value	Total time
1	3740	354.09	780376	401.25
2	3535	1123.73	785957	742.48
3	3900	477.43	790847	1321.79
4	3247	958.93	747652	732.95
5	3336	345.41	728334	989.42
6	2532	559.33	603834	992.54
7	2997	571.65	648184	1361.08
8	3342	694.53	702906	642.15
9	4144	876.08	769030	693.11
10	2975	722.34	648158	1199.26

the tanker is set as the first in the meeting order, then by discarding this mobile worker robot out, the second closest mobile worker robot is found and put as the second in the meeting order and so on. The results of the preliminary experiments for improvement heuristics for $n = 8$ are displayed in Table 5.2 and 5.3 for (EEMPR-T) and (EEMPR-E), respectively. For each instance, the best objective function value found among all the random and NN meeting order starts of three improvement heuristic algorithms is displayed in the column titled Best Obj. Overall. Also, we analyze in how many different starts this best objective function value is observed by each of the algorithm in columns titled # Best. Furthermore, the best and average objective function values along with average solution times (in seconds) are shown for random meeting order starts while the objective function value and solution times (in seconds) are displayed for the NN meeting order start. When we make a comparison, optimal objective function values found by the optimizing procedure and the best objective function values found by improvement heuristics are appeared to be the same for each instance for $n = 8$. Hence, we continue utilizing our improvement heuristics and maintain our preliminary experiments.

In remaining part of the preliminary experiments for improvement heuristic algorithms, we consider three values for $n : 15, 20,$ and 30. For each n value, 3 instances

Table 5.2: Computational times (in seconds) and objective function values of the improvement heuristic algorithms for (EEMPR-T) when n = 8

Instance	2-opt							3-opt							2-opt + 3-opt								
	Random				NN			Random				NN			Random				NN				
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Best Obj. Value Overall	% Difference
1	9	3740	3740	2.29	1	3740	0.93	9	3740	3740	11.02	1	3740	6.01	9	3740	3740	5.00	1	3740	3.91	3740	0.00
2	9	3535	3535	3.65	1	3535	0.49	9	3535	3535	20.36	1	3535	3.16	9	3535	3535	6.61	1	3535	3.51	3535	0.00
3	9	3900	3900	2.32	1	3900	0.95	9	3900	3900	11.51	1	3900	6.00	9	3900	3900	5.30	1	3900	3.74	3900	0.00
4	9	3247	3247	2.97	1	3247	1.37	9	3247	3247	14.70	1	3247	9.05	9	3247	3247	5.86	1	3247	4.18	3247	0.00
5	9	3336	3336	2.80	1	3336	1.88	9	3336	3336	12.41	1	3336	6.06	9	3336	3336	5.78	1	3336	4.72	3336	0.00
6	7	2532	2538	2.61	1	2532	0.91	9	2532	2532	11.96	1	2532	5.84	9	2532	2532	6.11	1	2532	3.83	2532	0.00
7	6	2997	2998	2.40	1	2997	1.32	9	2997	2997	11.21	1	2997	5.89	9	2997	2997	6.39	1	2997	4.26	2997	0.00
8	9	3342	3342	2.80	1	3342	0.49	9	3342	3342	13.27	1	3342	3.08	9	3342	3342	5.82	1	3342	3.61	3342	0.00
9	9	4144	4144	2.68	1	4144	0.48	9	4144	4144	14.75	1	4144	3.03	9	4144	4144	5.53	1	4144	3.49	4144	0.00
10	9	2975	2975	2.34	1	2975	0.91	9	2975	2975	10.80	1	2975	6.19	9	2975	2975	5.17	1	2975	3.82	2975	0.00

Table 5.3: Computational times (in seconds) and objective function values of the improvement heuristic algorithms for (EEMPR-E) when n = 8

Instance	2-opt							3-opt							2-opt + 3-opt								
	Random				NN			Random				NN			Random				NN				
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Best Obj. Value Overall	% Difference
1	9	780376	780376	2.54	1	780376	1.26	9	780376	780376	12.23	1	780376	7.30	9	780376	780376	5.93	1	780376	5.33	780376	0.00
2	9	785957	785957	4.18	1	785957	0.68	9	785957	785957	22.47	1	785957	3.72	9	785957	785957	7.84	1	785957	4.75	785957	0.00
3	9	790847	790847	2.62	1	790847	1.19	9	790847	790847	13.30	1	790847	7.31	9	790847	790847	6.37	1	790847	5.19	790847	0.00
4	9	747652	747652	3.24	1	747652	0.57	9	747652	747652	17.22	1	747652	3.64	9	747652	747652	7.02	1	747652	4.59	747652	0.00
5	9	728334	728334	3.01	1	728334	2.24	9	728334	728334	12.97	1	728334	7.16	9	728334	728334	6.68	1	728334	6.17	728334	0.00
6	9	603834	603834	2.99	1	603834	0.58	9	603834	603834	13.24	1	603834	3.64	9	603834	603834	6.72	1	603834	4.54	603834	0.00
7	9	648184	648184	2.83	1	648184	1.75	9	648184	648184	12.59	1	648184	7.26	9	648184	648184	6.46	1	648184	5.88	648184	0.00
8	9	702906	702906	3.00	1	702906	0.58	9	702906	702906	15.65	1	702906	3.54	9	702906	702906	6.47	1	702906	4.56	702906	0.00
9	9	769030	769030	3.23	1	769030	0.58	9	769030	769030	18.85	1	769030	3.66	9	769030	769030	6.96	1	769030	4.67	769030	0.00
10	9	648158	648158	2.89	1	648158	0.61	9	648158	648158	13.63	1	648158	3.89	9	648158	648158	6.61	1	648158	4.65	648158	0.00

are generated randomly. The experiments are conducted by running the improvement heuristic algorithms with three random meeting order starts and one Nearest Neighbor (NN) meeting order start for each instance. We set a three-hour time limit for each different start of an instance. The results of the preliminary experiments for improvement heuristics are displayed in Table 5.4 and Table 5.5 where the former shows the (EEMPR-T) version of the problem while the latter displays the results of (EEMPR-E) version of the problem. For each size of n and each instance, the best objective function value found among all the random and NN meeting order starts of three improvement heuristic algorithms is displayed in the column titled Best Obj. Overall. Also, we analyze in how many different starts this best objective function value is observed by each of the algorithm in columns titled $\#$ Best. Furthermore, the best and average objective function values along with average solution times (in seconds) are shown for random meeting order starts while the objective function value and solution times (in seconds) are displayed for the NN meeting order start.

For $n = 15$ and 20, 3-opt algorithm shows good quality results as means of finding the best objective function value for both versions of the problem, but the solution times are the slowest when compared to the 2-opt and combined 2-opt and 3-opt algorithms. Furthermore, for the second instance in both versions of the problem, and for the first instance of (EEMPR-T) when n is 30, 3-opt algorithm is not able to find feasible solutions within three-hour time limit with random meeting order start. Furthermore, even if it can find a feasible solution, it is unable to provide the best objective function value for random meeting order start when $n = 30$. However, when NN meeting order start is used, 3-opt algorithm is able to find the best objective function value in all of the instances except for the second instance of (EEMPR-T) whose objective function value only deviates from the best objective function value by 0.01 percent. Although NN meeting order start for 3-opt algorithm has good quality results as means of finding the best objective function value, the solution times are on average 4000.16 seconds for (EEMPR-T) and 4379.65 for (EEMPR-E) which are the slowest among all three algorithms for NN meeting order start. Hence, we can conclude that 3-opt algorithm shows the poorest performance among all three algorithms for both random and NN meeting order starts.

The fastest algorithm appeared as the 2-opt algorithm for each size of n with both

Table 5.4: Preliminary experimentation results of (EEMPR-T) for improvement heuristic algorithms

n	Instance No	2-opt							3-opt							2-opt + 3-opt							
		Random				NN			Random				NN			Random				NN			
		# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Best Obj. Overall
15	1	2	6094	6096	56.19	1	6094	6.17	3	6094	6094	637.68	1	6094	84.55	3	6094	6094	118.80	1	6094	48.50	6094
	2	3	5750	5755	49.98	1	5750	8.87	3	5750	5750	586.26	1	5750	90.40	3	5750	5750	97.08	1	5750	57.16	5750
	3	3	5181	5181	62.59	1	5181	5.66	3	5181	5181	704.24	1	5181	92.85	3	5181	5181	104.52	1	5181	52.80	5181
20	1	1	6754	6809	188.66	1	6754	23.06	1	6754	6755	3568.32	1	6754	419.63	1	6754	6755	411.13	1	6754	169.39	6754
	2	1	6812	6830	183.82	1	6812	11.19	1	6812	6813	3619.77	1	6812	278.05	1	6812	6813	346.26	1	6812	157.88	6812
	3	2	6657	6657	170.17	1	6657	46.90	2	6657	6657	3373.28	1	6657	748.49	3	6657	6657	318.85	1	6657	198.72	6657
30	1	1	11059	11062	1330.95	1	11059	223.31	0	36397	73×10^9	10799.96	1	11059	5049.14	2	11059	11059	2854.92	1	11059	1015.93	11059
	2	1	8629	8630	1266.48	0	8630	86.96	0	24×10^9	95×10^9	10799.93	0	8630	2528.30	1	8629	8630	2422.91	0	8630	903.54	8629
	3	2	9959	9988	1418.37	1	9959	127.55	0	72×10^9	150×10^9	10799.90	1	9959	4423.04	3	9959	9959	4060.23	1	9959	989.12	9959

Table 5.5: Preliminary experimentation results of (EEMPR-E) for improvement heuristic algorithms

n	Instance No	2-opt							3-opt							2-opt + 3-opt							
		Random				NN			Random				NN			Random				NN			
		# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Best Obj. Value Overall
15	1	3	1220173	1220173	64.99	1	1220173	8.07	3	1220173	1220173	761.82	1	1220173	104.09	3	1220173	1220173	119.15	1	1220173	64.97	1220173
	2	2	1298212	1311938	54.23	0	1319713	19.83	3	1298212	1298212	692.23	1	1298212	153.88	3	1298212	1298212	153.25	1	1298212	131.49	1298212
	3	3	1045929	1045929	67.64	1	1045929	3.97	3	1045929	1045929	886.43	1	1045929	53.70	3	1045929	1045929	129.27	1	1045929	63.37	1045929
20	1	3	1434714	1434714	223.00	1	1434714	47.63	3	1434714	1434714	4042.43	1	1434714	716.93	3	1434714	1434714	507.63	1	1434714	246.31	1434714
	2	3	1408935	1408935	216.52	1	1408935	17.06	3	1408935	1408935	3921.26	1	1408935	346.27	3	1408935	1408935	515.23	1	1408935	213.60	1408935
	3	1	1507593	1508732	216.83	0	1512716	75.47	3	1507593	1507593	3840.07	1	1507593	1173.79	2	1507593	1507593	616.41	0	1507594	502.70	1507593
30	1	1	2400182	2423429	1269.74	0	2445884	174.58	0	3981181	60×10^9	10799.94	1	2400182	4478.91	3	2400182	2400182	4103.27	1	2400182	3404.54	2400182
	2	2	2022610	2044962	1313.18	0	2029407	120.64	0	15×10^9	75×10^9	10799.94	1	2022610	4271.23	3	2022610	2022610	3286.86	1	2022610	1924.88	2022610
	3	1	2039472	2058590	1430.67	1	2039472	162.85	0	3063536	23×10^9	10799.96	1	2039472	4388.81	3	2039472	2039472	3609.11	1	2039472	1036.76	2039472

random and NN meeting order starts, which can be predicted since the 2-opt algorithm examines the minimum number of moves among all three algorithms, as discussed in section 4.2.1. For (EEMPR-T), the 2-opt algorithm is able to provide the best objective function value in total for 16 times in random meeting order start and 8 times in the NN meeting order starts. For (EEMPR-E), it finds the best objective function value in total for 19 times in random meeting order start and 5 times in NN meeting order starts. Although combined 2-opt and 3-opt algorithm runs slower than the 2-opt algorithm, for (EEMPR-T) it is able to provide the best objective function value 20 times in random meeting order starts and 8 times in NN starts in total, and for (EEMPR-E) it can find the best objective function value 26 times in random meeting order starts and 8 times in NN meeting order starts. Overall, when compared to the 2-opt algorithm, the solution quality of the combined 2-opt and 3-opt algorithm is higher, but solution times on average are 2.5 times slower when the random meeting order start is utilized for both of the versions of the problem while it is on average 6.5 times and 12 times slower for NN meeting order start for (EEMPR-T) and (EEMPR-E), respectively. Therefore, to use the advantage of high quality solutions of combined 2-opt and 3-opt algorithm, we decide to apply TSP speed-up techniques to make the algorithm faster by also preserving its solution quality.

5.2.3 Preliminary Experiments for Speed-up Algorithms

To be able to improve solution times, fixed radius search combined with candidate list and don't look bits approaches are applied to the combined 2-opt and 3-opt algorithm as discussed in Section 4.2.4, Algorithm 7. This algorithm is now called speed-up algorithm. To be able to do so, first of all, the candidate list length should be decided. Note that, because both versions of the problem has given similar results in the preliminary experiments conducted for the improvement heuristics as means of solution times and number of best solutions observed, we carry out the following preliminary experiments for only (EEMPR-T). The same instances generated for $n : 15, 20$, and 30 in Section 5.2.2 are continued to be used. Furthermore, we did not utilize instances for $n = 50$ in preliminary experiments for improvement heuristics due to excessive computational times, but now three instances are generated randomly for $n = 50$ to be able to examine the quality of the solutions better for increased instance sizes.

The experiments are conducted by running the speed-up algorithm with three random meeting order starts and one NN meeting order start for each instance. The evaluations are made based on three different candidate list lengths which are 1, $0.2n$ and 10.

We report the preliminary experimentation results for (EEMPR-T) in Table 5.6 along with a comparison of percent differences of best objective function values found by speed-up algorithm to the previous best objective function values which is provided by the improvement heuristics algorithms in Table 5.4. Also, we calculate the ratio of time by dividing the previous solution times provided in Table 5.4 to the solution times of the speed-up algorithm. There are not any objective function values or solution times for $n = 50$, so we solve these instances by using fixed radius search algorithm without candidate list and don't look bits approaches. The previous best objective function values and solution times for instances of $n = 50$ are taken from these fixed radius search algorithm solutions, see Table A.1 in the Appendix A. Furthermore, in Table 5.6, for each instance, the best objective function value found among all the random and NN meeting order starts of speed-up algorithm is displayed in the column titled Best Obj. Overall. Also, we analyze in how many different starts this best objective function value is observed by each of the algorithm and stored in columns titled # Best. Moreover, the best and average objective function values along with average solution times (in seconds) are shown for random meeting order starts while the objective function value and solution times (in seconds) are displayed for the NN meeting order start.

One can observe that, the average solution times are the fastest when candidate list length is equal to 1, but the algorithm is unable to find feasible solutions when n is 30 and 50 for random meeting order starts. Even though feasible solutions can be found with NN meeting order start for $n = 50$, the best objective function values found are not favorable enough when the percent differences are examined. For the instances, when n is 15 and 20 the candidate list lengths $0.2n$ and 10 provides very similar results as means of percent differences. Also, when n is 30, percent differences are the same with each other for candidate list lengths $0.2n$ and 10 for each instance. Realize that $0.2n$ is equal to 10 when $n = 50$, so the experiment does not have three but two different candidate list length parameters for this instance size. We observe

Table 5.6: Preliminary experimentation results of (EEMPR-T) for candidate list length trials

n	CL Length	Instance No	Speed-up algorithm											
			Random				NN							
			# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Best Obj. Value Overall	Previous Best Obj. Value Overall	% Difference of Best Obj. Values Overall	Previous Times	Ratio of Times
15	1	1	0	9445	14272	5.99	1	6181	0.89	6181	6094	1.41	118.80	134.24
		2	0	7176	8616	3.39	1	5757	0.90	5757	5750	0.12	97.08	107.99
		3	0	8264	$29x10^8$	4.03	1	5183	0.37	5183	5181	0.04	104.52	285.57
	0.2n	1	0	6110	6360	14.91	1	6097	2.15	6097	6094	0.05	118.80	55.28
		2	0	5837	6085	11.32	1	5786	1.86	5786	5750	0.62	97.08	52.08
		3	0	7305	7709	15.13	1	5183	0.57	5183	5181	0.04	104.52	184.01
	10	1	0	6160	8145	23.42	1	6097	3.70	6097	6094	0.05	118.80	32.11
		2	1	5785	5801	19.16	0	5786	2.44	5785	5750	0.61	97.08	39.84
		3	1	5182	7622	21.81	0	5183	0.62	5182	5181	0.02	104.52	168.58
20	1	1	0	8802	$19x10^9$	11.91	1	6903	1.44	6903	6754	2.16	411.13	285.90
		2	0	24940	$50x10^7$	16.14	1	6812	1.30	6812	6812	0.00	346.26	265.74
		3	0	9805	15117	12.57	1	6794	1.17	6794	6657	2.02	318.85	272.06
	0.2n	1	0	7798	13906	59.36	1	6778	5.74	6778	6754	0.35	411.13	71.58
		2	0	7179	8106	55.91	1	6812	3.08	6812	6812	0.00	346.26	112.46
		3	0	6676	7311	43.04	1	6669	6.89	6669	6657	0.18	318.85	46.30
	10	1	1	6776	7684	82.93	0	6778	7.98	6776	6754	0.32	411.13	51.52
		2	0	6834	6947	94.60	1	6812	3.87	6812	6812	0.00	346.26	89.38
		3	1	6665	6718	84.57	1	6665	14.08	6665	6657	0.12	318.85	22.65
30	1	1	0	$10x10^{10}$	$11x10^{10}$	50.00	1	11200	3.76	11200	11059	1.26	2854.92	759.69
		2	0	$62x10^9$	$11x10^{10}$	49.15	1	8654	2.63	8654	8629	0.29	2422.91	923.01
		3	0	$11x10^{10}$	$12x10^{10}$	48.31	1	9962	2.98	9962	9959	0.03	4060.23	1364.32
	0.2n	1	0	11353	13102	493.72	1	11135	25.44	11135	11059	0.68	2854.92	112.24
		2	0	9126	9452	328.56	1	8630	12.63	8630	8629	0.01	2422.91	191.85
		3	0	10568	15032	464.55	1	9962	9.56	9962	9959	0.03	4060.23	424.93
	10	1	0	11211	12456	623.93	1	11135	34.01	11135	11059	0.68	2854.92	83.94
		2	0	8918	10322	476.01	1	8630	13.33	8630	8629	0.01	2422.91	181.83
		3	0	10733	12434	571.90	1	9962	14.00	9962	9959	0.03	4060.23	290.12
50	1	1	0	$36x10^{10}$	$40x10^{10}$	203.37	1	15715	12.80	15715	13623	13.31	8146.89	636.87
		2	0	$32x10^{10}$	$34x10^{10}$	219.07	1	15440	12.19	15440	13230	14.31	6375.72	523.16
		3	0	$28x10^{10}$	$34x10^{10}$	169.28	1	30695	23.80	30695	13490	56.05	8012.68	336.64
	0.2n	1	0	13762	14117	3719.00	1	13709	207.24	13709	13623	0.63	8146.89	39.31
		2	0	13264	13435	6559.33	1	13258	164.48	13258	13230	0.21	6375.72	38.76
		3	1	13632	15248	4217.33	0	19981	446.94	13632	13490	1.04	8012.68	17.93

Note: CL refers to candidate list.

high quality solutions as means of percent differences for instances of $n = 50$ when candidate list length is $0.2n$. Furthermore, we observe faster solution times in each instance size for $0.2n$ when compared to 10. Nevertheless, when candidate list length is set as $0.2n$, even though solution times for instance sizes $n = 15, 20, 30$ decreased in a desirable way by also preserving the solution qualities when compared to the improvement heuristics, for $n = 50$, solution times are still very slow for random meeting order starts, i.e., on average 4831.89 seconds. This average time is required to be multiplied by the number of random starts the practitioner would like to execute which expands the solution time even more. Therefore, we search for a development in the speed-up algorithm to decrease the solution times. Nevertheless, if a practitioner has no time limitation, we can suggest speed-up algorithm with candidate list length $0.2n$ since it provides good results within a moderate amount of time.

To be able to improve the solution times, we want to make use of the fast solution times observed when candidate list length is 1. So, we worked on improving the objective function values by modifying the speed-up algorithm when candidate list length is set as 1. First of all, we tried a dynamic don't look bits approach which allow to the algorithm to turn on DLBs not for only one step but for k many steps. It can be expected that with an increase in k, we will observe an increase in the solution times. So, we have set k as 2. In Table 5.7, the results of this modification is provided under modified speed-up algorithm 1. There is an improvement in some of the instances when a comparison between the best objective function values found by the modified speed-up algorithm 1 and the speed-up algorithm with candidate list length 1, is made by looking at their percent differences. Note that, the previous best objective function values are taken as the same in both algorithms. So, we can directly compare percent differences columns of the algorithms. For example, the third instance when $n = 20$, the first instance when $n = 30$ and the second and third instances when $n = 50$ give better percent differences. All the other instances provide the same results with speed-up algorithm with candidate list length 1. In modified speed-up algorithm 1, the solution times are approximately 2 times slower for both random and NN meeting order starts for larger instance sizes, i.e., $n = 30, 50$ while for $n = 15, 20$ most of the solution times are almost the same when compared with the speed-up algorithm with candidate list length is 1. Although these results are promising, we are not able to improve all the objective function values as intended. For example, first and the third instances of $n = 50$ still provide high percent difference values of 13.31 and 29.58, respectively.

Secondly, we test the idea that when there is no improving move found by the speed-up algorithm, the fixed radius search combined with candidate list approach is executed for only one search step by removing the don't look bits approach. If at this step, an improving move is found then don't look bits approach is activated and the speed-up algorithm continues its execution. If no improving move is found, the algorithm is finalized. The preliminary experimentation results of this modified speed-up algorithm 2 are shown in Table 5.8. The solution times are approximately 2 times slower for both random and NN meeting order starts for higher instance sizes, i.e., $n = 30, 50$ while for $n = 15, 20$ most of the solution times are almost the same

Table 5.7: Preliminary experimentation results of (EEMPR-T) for modified speed-up algorithm 1

n	Instance No	Modified speed-up algorithm 1									
		Random				NN					
		# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Best Obj. Value Overall	Previous Best Obj. Value Overall	% Difference of Best Obj. Values Overall
15	1	0	6294	8279	6.96	1	6181	0.78	6181	6094	1.41
	2	0	6239	8116	3.77	1	5757	1.26	5757	5750	0.12
	3	0	7169	7655	7.01	1	5183	0.31	5183	5181	0.04
20	1	0	12452	15201	23.12	1	6903	1.45	6903	6754	2.16
	2	0	7940	8792	24.21	1	6812	1.69	6812	6812	0.00
	3	0	7343	11566	16.71	1	6775	1.93	6775	6657	1.74
30	1	0	24013	27594	102.03	1	11150	7.44	11150	11059	0.82
	2	0	12033	17894	112.19	1	8654	3.25	8654	8629	0.29
	3	0	12218	17570	136.53	1	9962	3.18	9962	9959	0.03
50	1	0	29634	$17\text{x}10^9$	545.14	1	15715	21.49	15715	13623	13.31
	2	0	15662	$39\text{x}10^8$	505.41	1	13977	32.68	13977	13230	5.34
	3	1	19156	$72\text{x}10^7$	513.31	0	27529	51.93	19156	13490	29.58

when compared with the speed-up algorithm with candidate list length 1. However, we only achieve a very small improvement in the third instance of $n = 50$ which is not desirable. If we compare this instance to the same instance in modified speed-up algorithm 1, the percent difference gets even worse.

Table 5.8: Preliminary experimentation results of (EEMPR-T) for modified speed-up algorithm 2

n	Instance No	Modified speed-up algorithm 2									
		Random				NN					
		# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Best Obj. Value Overall	Previous Best Obj. Value Overall	% Difference of Best Obj. Values Overall
15	1	1	6121	6801	7.22	0	6181	1.80	6121	6094	0.44
	2	0	6511	6857	3.92	1	5757	1.56	5757	5750	0.12
	3	0	8264	8762	6.19	1	5183	0.67	5183	5181	0.04
20	1	0	12585	13337	14.95	1	6903	2.94	6903	6754	2.16
	2	0	8204	10368	19.24	1	6812	2.58	6812	6812	0.00
	3	0	7527	7658	13.99	1	6775	2.86	6775	6657	1.74
30	1	0	22548	27063	70.35	1	11150	9.50	11150	11059	0.82
	2	0	9890	99	86.31	1	8654	5.47	8654	8629	0.29
	3	0	10803	17339	89.47	1	9962	5.33	9962	9959	0.03
50	1	0	16488	$72\text{x}10^8$	490.13	1	15715	19.95	15715	13623	13.31
	2	0	21826	$63\text{x}10^8$	541.22	1	13977	35.03	13977	13230	5.34
	3	1	26416	$38\text{x}10^8$	423.96	0	27258	41.26	26416	13490	48.93

Thirdly, we continue to improve the modified speed-up algorithm 2 by extending the candidate list length to 2 during the improvement search step and afterwards. In other words, if there is no improving move found by the speed-up algorithm, the fixed radius search combined with candidate list approach is executed for only one

Table 5.9: Preliminary experimentation results of (EEMPR-T) for modified speed-up algorithm 3

n	Instance No	Modified speed-up algorithm 3									
		Random				NN					
		# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Best Obj. Value Overall	Previous Best Obj. Value Overall	% Difference of Best Obj. Values Overall
15	1	0	7430	9409	6.88	1	6097	2.78	6097	6094	0.05
	2	0	5828	7297	4.98	1	5786	2.33	5786	5750	0.62
	3	1	5181	6796	7.54	0	5183	1.05	5181	5181	0.00
20	1	0	7005	8921	25.22	1	6778	4.02	6778	6754	0.35
	2	0	7409	11157	27.01	1	6812	3.40	6812	6812	0.00
	3	0	6944	7520	25.03	1	6669	4.33	6669	6657	0.18
30	1	0	11158	13661	118.18	1	11135	12.04	11135	11059	0.68
	2	0	9099	10157	96.27	1	8654	7.63	8654	8629	0.29
	3	0	13368	15083	98.41	1	9962	6.11	9962	9959	0.03
50	1	1	13836	22532	696.00	0	15412	58.32	13836	13623	1.54
	2	1	13650	19869	815.17	0	13965	41.26	13650	13230	3.08
	3	1	13885	18903	772.86	0	18054	93.65	13885	13490	2.84

step by removing the don't look bits approach and extending the candidate list size to 2. If at this step, an improving move is found then don't look bits approach is activated by preserving the extended candidate list length and the algorithm continues execution. If no improving move is found, the algorithm is finalized. The results of the preliminary experiment for the modified speed-up algorithm 3 can be seen in Table 5.9. Except for the second instance of $n = 15$ which has a small amount of increase, all the other instances either gets better or stays the same as means of the percent differences when compared with the speed-up algorithm with candidate list length 1. Particularly, we observe an improvement in the percent differences of each instance for $n = 50$. Furthermore, solution times are now compatible for even larger instance sizes. For $n = 30$, one random meeting order start lasts on average 104.29 seconds and one NN meeting order starts lasts on average 8.59 seconds while for $n = 50$ these are 761.34 and 64.41, respectively. Therefore, we end up using the modified speed-up algorithm 3 for cases which requires faster results with a compensation for some loss on the best objective function value found.

Before moving forward, we wanted to solve the modified speed-up algorithm 3 for $n = 8$ and compare the results with the results of optimizing procedure. In Tables 5.10 and 5.11, the outputs of the modified speed-up algorithm 3 when $n = 8$ for (EEMPR-T) and (EEMPR-E) are given, respectively. The column called as Previous Best Obj. Value stores the objective function values of the optimizing procedure and

Table 5.10: Computational times (in seconds) and objective function values of the modified speed-up algorithm 3 for (EEMPR-T) when n = 8

Instance	Random				NN			Best Obj.	Previous Best	% Difference of
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Value Overall	Obj. Value	Obj. Values
1	0	3921	4456	1	1	3740	0	3740	3740	0.00
2	1	3535	5815	1	1	3535	0	3535	3535	0.00
3	1	3900	4061	1	0	3957	0	3900	3900	0.00
4	1	3247	3828	1	0	3268	0	3247	3247	0.00
5	1	3336	3852	1	0	3880	1	3336	3336	0.00
6	1	2532	2827	1	1	2532	1	2532	2532	0.00
7	1	2997	3058	1	0	3001	0	2997	2997	0.00
8	0	3415	3858	1	1	3342	0	3342	3342	0.00
9	1	4144	5114	1	1	4144	1	4144	4144	0.00
10	1	2975	3227	1	0	2986	0	2975	2975	0.00

Table 5.11: Computational times (in seconds) and objective function values of the modified speed-up algorithm 3 for (EEMPR-E) when n = 8

Instance	Random				NN			Best Obj.	Previous Best	% Difference of
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Value Overall	Obj. Value	Obj. Values
1	1	780376	860279	1	1	780376	1	780376	780376	0.00
2	1	785957	937505	1	1	785957	0	785957	785957	0.00
3	1	790847	941447	0	0	835811	0	790847	790847	0.00
4	0	837395	912550	1	1	747652	0	747652	747652	0.00
5	1	728334	818114	1	0	979036	0	728334	728334	0.00
6	0	603977	680154	1	1	603834	1	603834	603834	0.00
7	1	648184	680368	1	0	662683	0	648184	648184	0.00
8	1	702906	816244	1	1	702906	0	702906	702906	0.00
9	0	769316	922764	1	1	769030	1	769030	769030	0.00
10	1	648158	732551	1	1	648158	1	648158	648158	0.00

can also be found in Table 5.1. The column called as % Difference of Obj. Values is calculated by comparing the Best Obj. Value Overall column and Previous Best Obj. Value column. Therefore, it can be observed that the modified speed-up algorithm 3 is able to find the optimal solutions for $n = 8$.

5.3 Computational Experiments

For the computational experiments, all mobile worker robots considered as homogeneous. In other words, the energy consumption and the battery recharge function parameters are taken as the same as $\alpha = 3$, $\beta = 10$, $\gamma = 5$ and $\sigma = 0.005$ for each mobile worker robot. The tanker robot is assumed to consume two times as much energy as a mobile worker robot. Hence, parameters are taken as 2α, 2β and 2γ in the energy consumption function of the tanker robot. Furthermore, we assume that mobile worker robots are recharged up to their maximum energy storage levels, D. This means that E^{max} is taken as a parameter instead of a decision variable and assumed to be equal to D. Also, the final locations of both the mobile worker robots and the tanker robot are assumed to be the same as their initial locations. The mobile worker robots must have 80% of their energy left when they return back to their initial locations. Hence θ used in constraint 3.14 is chosen as 0.8. The maximum speed for both mobile worker robots and the tanker robot is taken as $v^{max} = 5$ m/s. Last but not least, the big M value is taken as $100,000$.

In these experiments, we consider 6 values for n : $5, 8, 15, 20, 30$, and 50. For each n value, 10 instances are generated. We have solved each instance for 10 times. In 9 of them randomly generated initial meeting orders are used while in 1 of them a NN meeting order start is utilized. We have limited the computational times for three hours for each different start of an instance. In the random initial meeting order starts, we demonstrate the best objective function value of 9 random starts, their average objective function value and average computational times (in seconds) per instance. Also, among these 9 random starts, in how many of them the overall best objective function value is observed is shown in the # Best column. Then, we present the objective function value found by NN meeting order start and their computational times (in seconds). Again, we keep the information whether the overall best objective function value is observed by NN meeting order start or not in the column # Best. Furthermore, we represent the percent difference between the best objective value overall and the best objective function value found by all of the algorithms so far in % Difference column. This column is not present in the tables when $n = 50$ since we do not solve these instances by the previous algorithms due to excessive computational times.

We solved instances where $n = 5, 8, 15, 20, 30$ by using 2-opt, 3-opt and combined 2-opt and 3-opt algorithms for both (EEMPR-T) and (EEMPR-E). Improvement heuristics algorithms are not solved for $n = 50$ due to excessive computational times. In addition, for instances where $n = 30, 50$, we utilized speed-up algorithm with candidate list length $0.2n$ and modified speed-up algorithm 3 for both (EEMPR-E) and (EEMPR-T). These algorithms are not solved for $n = 20$ since we already observe

Table 5.12: Computational times (in seconds) and objective function values of the improvement heuristic algorithms for (EEMPR-T) when n = 20

	2-opt							3-opt							2-opt + 3-opt								
	Random				NN			Random				NN			Random				NN				
	# Best	Best Obj.	Avg. Obj.	Avg. Time	# Best	Obj.	Time	# Best	Best Obj.	Avg. Obj.	Avg. Time	# Best	Obj.	Time	# Best	Best Obj.	Avg. Obj.	Avg. Time	# Best	Obj.	Time	Best Obj. Overall	% Difference
1	3	6754	6809	188.66	1	6754	23.06	3	6754	6755	3568.32	1	6754	419.63	4	6754	6755	411.13	1	6754	169.39	6754	0.00
2	1	6812	6830	183.82	1	6812	11.19	2	6812	6813	3619.77	1	6812	278.05	1	6812	6813	346.26	1	6812	157.88	6812	0.00
3	8	6657	6657	170.17	1	6657	46.90	5	6657	6657	3373.28	1	6657	748.49	8	6657	6657	318.85	1	6657	198.72	6657	0.00
4	4	7851	7852	191.05	0	7854	31.94	4	7851	7852	3856.13	0	7852	300.17	4	7851	7852	355.77	0	7852	325.10	7851	0.00
5	6	5874	5927	207.61	1	5874	31.16	8	5874	5874	4028.84	1	5874	442.13	9	5874	5874	416.02	1	5874	183.45	5874	0.00
6	6	5943	5969	213.58	1	5943	45.14	9	5943	5943	4547.13	1	5943	734.39	9	5943	5943	416.78	1	5943	198.31	5943	0.00
7	1	7501	7502	228.27	1	7501	12.67	5	7501	7501	4546.33	1	7501	303.94	1	7501	7502	384.41	1	7501	163.81	7501	0.00
8	7	6394	6394	203.31	1	6394	53.57	5	6394	6394	3821.77	0	6395	706.37	7	6394	6394	360.69	1	6394	208.40	6394	0.00
9	9	6684	6684	233.24	1	6684	32.76	9	6684	6684	3450.55	1	6684	669.73	9	6684	6684	398.32	1	6684	190.14	6684	0.00
10	9	5594	5594	161.94	1	5594	30.43	9	5594	5594	2478.08	1	5594	432.08	9	5594	5594	324.99	1	5594	181.02	5594	0.00

Table 5.13: Computational times (in seconds) and objective function values of the improvement heuristic algorithms for (EEMPR-E) when n = 20

	2-opt							3-opt							2-opt + 3-opt								
	Random				NN			Random				NN			Random				NN				
	# Best	Best Obj.	Avg. Obj.	Avg. Time	# Best	Obj.	Time	# Best	Best Obj.	Avg. Obj.	Avg. Time	# Best	Obj.	Time	# Best	Best Obj.	Avg. Obj.	Avg. Time	# Best	Obj.	Time	Best Obj. Overall	% Difference
1	9	1434714	1434714	223.00	1	1434714	47.63	9	1434714	1434714	4042.43	1	1434714	716.93	9	1434714	1434714	507.63	1	1434714	246.31	1434714	0.00
2	9	1408935	1408935	216.52	1	1408935	17.06	9	1408935	1408935	3921.26	1	1408935	346.27	9	1408935	1408935	515.23	1	1408935	213.60	1408935	0.00
3	6	1507593	1508732	216.83	0	1512716	75.47	8	1507593	1507593	3840.07	1	1507593	1173.79	7	1507593	1507593	616.41	0	1507594	502.70	1507593	0.00
4	6	1788926	1789692	227.19	1	1788926	101.85	9	1788926	1788926	4140.67	1	1788926	773.40	9	1788926	1788926	660.87	1	1788926	307.60	1788926	0.00
5	8	1394394	1416888	226.14	1	1394394	24.94	9	1394394	1394394	4312.17	1	1394394	354.18	9	1394394	1394394	591.90	1	1394394	212.73	1394394	0.00
6	9	1309436	1309436	236.45	1	1309436	44.86	9	1309436	1309436	4716.84	1	1309436	545.50	9	1309436	1309436	571.01	1	1309436	240.58	1309436	0.00
7	9	1565578	1565578	253.52	1	1565578	24.38	9	1565578	1565578	4884.76	1	1565578	363.46	9	1565578	1565578	597.09	1	1565578	219.45	1565578	0.00
8	6	1413559	1417596	224.43	0	1421122	64.49	9	1413559	1413559	4557.96	1	1413559	994.18	9	1413559	1413559	683.18	1	1413559	494.38	1413559	0.00
9	9	1421640	1421640	255.46	1	1421640	49.43	9	1421640	1421640	5359.11	1	1421640	946.82	9	1421640	1421640	546.10	1	1421640	247.19	1421640	0.00
10	9	1239329	1239329	198.13	1	1239329	38.82	9	1239329	1239329	3254.68	1	1239329	710.26	9	1239329	1239329	433.85	1	1239329	276.87	1239329	0.00

fast solution times in improvement heuristic algorithms for these instances.

The results of the heuristics algorithms solved for both (EEMPR-T) and (EEMPR-E) versions of the problem for $n = 5$ and 15 can be found in Tables A.2, A.3, A.4, and A.5 in Appendix A. The results for $n = 8$ can be found in the previously given Tables 5.2 and 5.3.

The results of the heuristics algorithms for $n = 20$ can be seen in Tables 5.12 and 5.13 for (EEMPR-T) and (EEMPR-E) versions of the problem, respectively. For (EEMPR-T) version of the problem, if there is no time restriction for the computational studies, combined 2-opt and 3-opt algorithm should be executed since it is able to find the best objective function value in more starts than the 2-opt algorithm. Although for some of the instances 3-opt algorithm can arrive at the best objective function value solutions in more random starts, the solution times are very slow when compared to the combined 2-opt and 3-opt algorithm. For combined 2-opt and 3-opt algorithm, on average 373.32 seconds and 197.62 seconds are required for random and NN meeting order starts, respectively. When 9 random and 1 NN meeting order starts are executed, computations last approximately 1 hour. For (EEMPR-E) version of the problem, except for instance 3, combined 2-opt and 3-opt algorithm is able to find the best objective function value in all random and NN meeting order starts with solution times on average 572.33 and 296.14 seconds, respectively. When 9 random and 1 NN meeting order starts are executed, computations last approximately 1.5 hours. 3-opt algorithm is not desirable because of the excessive solution times. For both versions of the problem, if faster solution times are required, 2-opt algorithm can be executed with a small amount of decrease in the probability of being able to find the best objective function value, especially for (EEMPR-T) version of the problem. If 9 random and 1 NN meeting order start is executed for 2-opt algorithm, approximately 30 and 35 minutes are required for (EEMPR-T) and (EEMPR-E), respectively. These computational time requirements can be decreased by using parallel computers.

The results of the heuristics algorithms for $n = 30$ can be seen in Tables 5.14 and 5.15 for (EEMPR-T) and (EEMPR-E) versions of the problem, respectively. The combined 2-opt and 3-opt algorithm provides the best results as means of the total number of best objective function values observed in starts. However, the solu-

Table 5.14: Computational times (in seconds) and objective function values of the improvement heuristic algorithms for (EEMPR-T) when n = 30

	2-opt							3-opt							2-opt + 3-opt								
	Random				NN			Random				NN			Random				NN				
	# Best	Best Obj.	Avg. Obj.	Avg. Time	# Best	Obj.	Time	# Best	Best Obj.	Avg. Obj.	Avg. Time	# Best	Obj.	Time	# Best	Best Obj.	Avg. Obj.	Avg. Time	# Best	Obj.	Time	Best Obj. Overall	% Difference
1	2	11059	11062	1330.95	1	11059	223.31	0	36397	$73x10^9$	10799.96	1	11059	5049.14	5	11059	11059	2854.92	1	11059	1015.93	11059	0.00
2	1	8629	8630	1266.48	0	8630	86.96	0	$24x10^9$	$95x10^9$	10799.93	0	8630	2528.30	1	8629	8630	2422.91	0	8630	903.54	8629	0.00
3	5	9959	9988	1418.37	1	9959	127.55	0	$72x10^9$	$152x10^9$	10799.90	1	9959	4423.04	9	9959	9959	4060.23	1	9959	989.12	9959	0.00
4	6	8055	8066	1244.18	1	8055	130.26	0	$24x10^9$	$68x10^9$	10799.92	1	8055	2557.05	9	8055	8055	2230.13	1	8055	977.26	8055	0.00
5	2	8920	8922	1383.82	0	8923	220.35	0	$66x10^9$	$118x10^9$	10799.94	1	8920	7605.50	5	8920	8920	2720.88	1	8920	1987.91	8920	0.00
6	5	8687	8691	1309.65	1	8687	384.50	0	$35x10^9$	$95x10^9$	10799.94	1	8687	9710.59	8	8687	8687	2624.04	1	8687	1299.35	8687	0.00
7	3	9566	9567	1179.80	0	9567	150.73	0	$28x10^9$	$79x10^9$	10799.92	1	9566	5653.37	4	9566	9567	2503.58	1	9566	2973.44	9566	0.00
8	9	8873	8873	1440.61	1	8873	21.98	0	$56x10^9$	$122x10^9$	10799.93	1	8873	969.52	9	8873	8873	2646.25	1	8873	999.92	8873	0.00
9	2	9550	9588	1166.19	1	9550	129.87	0	24222	$40x10^9$	10799.95	1	9550	4807.05	2	9550	9555	3627.81	1	9550	1070.91	9550	0.00
10	0	7566	7566	1088.32	1	7565	175.89	0	14983	$25x10^9$	10799.95	1	7565	5963.23	0	7566	7566	2882.39	1	7565	1117.34	7565	0.00

Table 5.15: Computational times (in seconds) and objective function values of the improvement heuristic algorithms for (EEMPR-E) when n = 30

	2-opt							3-opt							2-opt + 3-opt								
	Random				NN			Random				NN			Random				NN				
	# Best	Best Obj.	Avg. Obj.	Avg. Time	# Best	Obj.	Time	# Best	Best Obj.	Avg. Obj.	Avg. Time	# Best	Obj.	Time	# Best	Best Obj.	Avg. Obj.	Avg. Time	# Best	Obj.	Time	Best Obj. Overall	% Difference
1	1	2400182	2423429	1269.74	0	2445884	174.58	0	3981181	$60x10^9$	10799.94	1	2400182	4478.91	9	2400182	2400182	4103.27	1	2400182	3404.54	2400182	0.00
2	6	2022610	2044962	1313.18	0	2029407	120.64	0	$15x10^9$	$75x10^9$	10799.94	1	2022610	4271.23	9	2022610	2022610	3286.86	1	2022610	1924.88	2022610	0.00
3	6	2039472	2058590	1430.67	1	2039472	162.85	0	$31x10^9$	$99x10^9$	10799.93	1	2039472	4388.81	9	2039472	2039472	3609.11	1	2039472	1036.76	2039472	0.00
4	7	1772701	1788810	1291.90	0	1780026	116.80	0	3171001	$32x10^9$	10799.95	1	1772701	3787.24	9	1772701	1772701	3512.92	1	1772701	2000.15	1772701	0.00
5	9	1927845	1927845	1503.28	1	1927845	306.68	0	3836305	$61x10^9$	10799.95	1	1927845	7872.53	9	1927845	1927845	3378.51	1	1927845	1245.79	1927845	0.00
6	1	2075064	2092955	1450.98	1	2075064	369.91	0	3518465	$49x10^9$	10799.96	1	2075064	9023.36	9	2075064	2075064	5337.78	1	2075064	1266.82	2075064	0.00
7	3	2255929	2272103	1347.80	0	2264819	191.49	0	4016026	$34x10^9$	10799.94	1	2255929	5455.20	7	2255929	2255961	6166.04	1	2255929	2056.04	2255929	0.00
8	7	1738229	1786268	1524.44	1	1738229	93.85	0	$10x10^9$	$84x10^9$	10799.97	1	1738229	4096.14	9	1738229	1738229	2545.14	1	1738229	983.59	1738229	0.00
9	3	2130996	2208257	1462.57	1	2130996	198.47	0	3687081	$41x10^9$	10799.93	1	2130996	5418.75	9	2130996	2130996	4146.44	1	2130996	1173.11	2130996	0.00
10	7	1838697	1848088	1442.84	1	1838697	305.22	0	3063536	$23x10^9$	10799.96	1	1838697	7495.80	9	1838697	1838697	2432.13	1	1838697	1315.01	1838697	0.00

tion times are very high. If 9 random and 1 NN meeting order starts are executed, then a practitioner needs approximately 7.5 hours and 10 hours for (EEMPR-T) and (EEMPR-E) versions, respectively, which are not desirable.

The results of the speed-up algorithm with candidate list length $0.2n$ for $n = 30$ can be observed in Tables 5.16 and 5.17 for (EEMPR-T) and (EEMPR-E) versions of the problem, respectively. On average, the random and NN meeting order starts lasts for 421.11 and 23.13 seconds while for (EEMPR-E) 390.94 and 24.90 seconds, respectively. Therefore, in total, 9 random an 1 NN meeting order start requires approximately 1 hour for both of the versions of the problem. Furthermore, the results of the speed-up algorithm with candidate list length $0.2n$ for $n = 50$ can be observed in Tables 5.18 and 5.19 for (EEMPR-T) and (EEMPR-E) versions of the problem, respectively. On average, the random and NN meeting order starts lasts for 4607.86 and 314.98 seconds while for (EEMPR-E) 5191.08 and 305.11 seconds, respectively. Therefore, in total, 9 random an 1 NN meeting order start requires 11.5 and 13 hours for (EEMPR-T) and (EEMPR-E), respectively. To conclude, if a practitioner has time for computations, we suggest to utilize speed-up algorithm for cases where more than 30 mobile worker robots are used in a field.

Nevertheless, if a practitioner has time restriction due to the nature of the mission, we suggest them to utilize modified speed-up algorithm 3 since it requires less amount of time. To be more specific, the results of the modified speed-up algorithm 3 for $n = 30$ can be observed in Tables 5.20 and 5.21 for (EEMPR-T) and (EEMPR-E) versions of the problem, respectively. On average, the random and NN meeting order starts lasts for 101.35 and 10.52 seconds while for (EEMPR-E) 104.05 and 14.98 seconds, respectively. Therefore, in total, 9 random an 1 NN meeting order start requires approximately 15 minutes for both of the versions of the problem. Moreover, the results of the modified speed-up algorithm 3 when $n = 50$ are displayed in Tables 5.22 and 5.23 for (EEMPR-T) and (EEMPR-E) versions of the problem, respectively. On average, the random and NN meeting order starts lasts for 783.56 and 79.25 seconds while for (EEMPR-E) 697.33 and 63.93 seconds, respectively. Therefore, in total, 9 random an 1 NN meeting order start requires approximately 2 hours for both of the versions of the problem. These computational time requirements can be decreased even more by using parallel computers.

Table 5.16: Computational times (in seconds) and objective function values of the speed-up algorithm with candidate list length 0.2n for (EEMPR-T) when n = 30

Instance	Random				NN			Best Obj.	Previous Best	% Difference of
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Value Overall	Obj. Value	Obj. Values
1	1	11084	11322	466.51	0	11135	25.16	11084	11059	0.23
2	1	8630	8831	418.72	1	8630	11.78	8630	8629	0.01
3	0	9973	10780	445.86	1	9962	8.96	9962	9959	0.03
4	1	8074	8235	397.74	0	8082	16.85	8074	8055	0.24
5	1	8931	9064	425.87	0	9763	20.51	8931	8920	0.12
6	1	8700	8965	438.58	0	9050	95.55	8700	8687	0.15
7	0	9576	9619	410.86	1	9574	13.70	9574	9566	0.08
8	0	8875	9232	439.11	1	8873	9.14	8873	8873	0.00
9	0	9561	9822	366.69	1	9556	8.76	9556	9550	0.06
10	1	7588	7726	401.16	0	7813	20.86	7588	7565	0.30

Table 5.17: Computational times (in seconds) and objective function values of the speed-up algorithm with candidate list length 0.2n for (EEMPR-E) when n = 30

Instance	Random				NN			Best Obj.	Previous Best	% Difference of
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Value Overall	Obj. Value	Obj. Values
1	0	2698674	2952352	389.92	1	2485201	30.99	2485201	2400182	3.42
2	0	2169053	2543873	331.04	1	2059071	13.34	2059071	2022610	1.77
3	0	2156231	2646105	407.77	1	2074426	11.05	2074426	2039472	1.68
4	0	1864085	2203654	332.51	1	1786594	15.24	1786594	1772701	0.78
5	1	2093609	2549324	358.21	0	2285873	17.72	2093609	1927845	7.92
6	0	2356946	2548176	434.76	1	2196636	91.38	2196636	2075064	5.53
7	0	2360163	2675634	401.54	1	2330462	7.74	2330462	2255929	3.20
8	0	1968429	2236950	481.02	1	1740512	11.51	1740512	1738229	0.13
9	0	2392080	2581148	352.87	1	2172974	17.88	2172974	2130996	1.93
10	0	2054180	2284515	419.70	1	1939110	32.11	1939110	1838697	5.18

Furthermore, you can find the summaries of the algorithms in Tables 5.24 and 5.25, for (EEMPR-T) and (EEMPR-E), respectively. The average percent differences for objective function values are calculated based on the best objective function value for each instance found by all of the algorithms solved so far. When $n = 20$ and 30, the best objective function values are observed in improvement heuristics algorithms. However, when $n = 50$, we used the best objective function values seen in speed-up heuristics since we do not solve the heuristics algorithms for these instances. Also, for NN, we do not have any average or worst values for the objective function values since we only solve the algorithm with only one start which is an NN meeting order start. The values for random starts are the averages of all the instances based on different 9 random meeting order starts. An example calculation of these values can be given as follows. We observe the best objective function values found by all of

Table 5.18: Computational times (in seconds) and objective function values of the speed-up algorithm with candidate list length 0.2n for (EEMPR-T) when n = 50

Instance	Random				NN			Best Obj.
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Value Overall
1	0	13763	15224	4423.53	1	13709	203.16	13709
2	0	13275	13889	5171.04	1	13258	163.55	13258
3	1	13633	15281	4041.65	0	19981	431.57	13633
4	0	14822	15597	4719.05	1	14817	509.62	14817
5	0	15902	17305	4605.73	1	15843	459.21	15843
6	1	13468	13749	5195.85	0	13496	106.82	13468
7	1	13389	13890	4970.54	0	15561	310.55	13389
8	1	15699	19700	4787.47	0	22343	318.62	15699
9	1	15018	17485	4372.93	0	15169	446.06	15018
10	0	13363	14478	3790.77	1	13300	200.63	13300

Table 5.19: Computational times (in seconds) and objective function values of the speed-up algorithm with candidate list length 0.2n for (EEMPR-E) when n = 50

Instance	Random				NN			Best Obj.
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Value Overall
1	0	3518574	3810386	4826.33	1	3305553	123.11	3305553
2	0	3550945	3846592	4634.18	1	3456016	81.04	3456016
3	1	3423864	3810586	4974.63	0	3568597	337.46	3423864
4	1	3360792	3770216	5041.85	0	3554196	510.13	3360792
5	1	3640956	4000643	4582.40	0	3892224	383.03	3640956
6	0	3439439	3956339	5119.82	1	3278941	172.15	3278941
7	1	3180859	3375706	6074.98	0	3692085	511.32	3180859
8	1	4001039	4254164	5826.06	0	4546485	388.46	4001039
9	0	3944798	4373212	6251.40	1	3745553	255.01	3745553
10	1	3611398	4038793	4579.17	0	3925866	289.39	3611398

the algorithms for (EEMPR-T) when $n = 30$ in Table 5.14 in Best Obj. Overall column. For each instance, we compare these values one by one with the average objective function values found by the modified speed-up algorithm 3 seen in Table 5.20 in column Avg. Obj. Value. The comparison is made to see the % differences by calculating how much improvement is required to achieve the best objective function value. In the end, we take the average of all these % differences.

The values observed as 100.000 means that the instances are infeasible and require 100% improvement to become feasible. Moreover, in some of the instances, we ob-

Table 5.20: Computational times (in seconds) and objective function values of the modified speed-up algorithm 3 for (EEMPR-T) when n = 30

Instance	Random				NN			Best Obj.	Previous Best	% Difference of
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Value Overall	Obj. Value	Obj. Values
1	0	11284	17866	103.72	1	11135	11.26	11135	11059	0.68
2	1	8648	11426	106.29	0	8654	7.09	8648	8629	0.22
3	0	10070	17820	107.30	1	9962	5.98	9962	9959	0.03
4	1	8069	10204	92.19	0	8141	9.37	8069	8055	0.17
5	1	8958	11375	98.18	0	15522	10.07	8958	8920	0.42
6	1	8725	12644	112.25	0	19513	25.07	8725	8687	0.44
7	1	9584	14419	98.79	0	9621	7.54	9584	9566	0.19
8	0	9188	12095	111.50	1	8873	4.89	8873	8873	0.00
9	1	9589	11202	94.78	0	10752	8.11	9589	9550	0.41
10	1	7612	11742	88.47	0	7813	15.78	7612	7565	0.62

Table 5.21: Computational times (in seconds) and objective function values of the modified speed-up algorithm 3 for (EEMPR-E) when n = 30

Instance	Random				NN			Best Obj.	Previous Best	% Difference of
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Value Overall	Obj. Value	Obj. Values
1	0	2584709	3013017	98.28	1	2534891	18.47	2534891	2400182	5.31
2	0	2174729	2556566	111.22	1	2065868	8.58	2065868	2022610	2.09
3	0	2510292	2905690	122.51	1	2074426	6.23	2074426	2039472	1.68
4	0	2021662	2487332	86.77	1	1786594	9.22	1786594	1772701	0.78
5	1	2129074	2695540	103.82	0	2316571	8.68	2129074	1927845	9.45
6	1	2265556	$17\text{x}10^7$	122.89	0	2539601	48.87	2265556	2075064	8.41
7	0	2415421	2907891	93.99	1	2365036	7.32	2365036	2255929	4.61
8	0	1844294	2635520	110.76	1	1740512	6.42	1740512	1738229	0.13
9	1	2378329	2707315	94.32	0	2438792	11.74	2378329	2130996	10.40
10	0	1957516	2416234	95.98	1	1908355	24.28	1908355	1838697	3.65

serve both feasibility and infeasibility in best, average or worst objective function values. To calculate the average percent differences, we also take these infeasible solutions as 100% while calculating the average. For instance, in Table 5.24, when $n = 30$, we observe 87.969 as average % difference of bests for 3-opt algorithm, if we exclude the infeasible solutions observed in the instances, we would end up seeing this value as 59.899.

The average percent differences of times are calculated based on the times of 2-opt and 3-opt algorithms of random and NN meeting order starts. Since these are the base values, we observe 0.000 values in Avg. % Difference of Times rows of 2-opt and 3-opt algorithms for each value of n. Note that since we do not solve the heuristics algorithms for $n = 50$ instances, the average percent difference of times cannot be represented. An example calculation for Avg. % Difference of Times row can be

Table 5.22: Computational times (in seconds) and objective function values of the modified speed-up algorithm 3 for (EEMPR-T) when n = 50

Instance	Random				NN			Best Obj.
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Value Overall
1	1	13836	22532	696.00	0	15412	58.32	13836
2	1	13650	19869	815.17	0	13965	41.26	13650
3	1	13885	18903	772.86	0	18054	93.65	13885
4	0	17434	25803	697.61	1	16759	89.53	16759
5	1	16631	19494	671.80	0	22903	51.23	16631
6	1	13515	20449	747.05	0	14426	40.11	13515
7	1	13648	19323	912.73	0	20906	127.70	13648
8	1	16226	22232	802.20	0	22634	82.42	16226
9	0	16524	$90\text{x}10^{7}$	961.85	1	15428	132.49	15428
10	1	13559	19136	758.31	0	14277	75.77	13559

Table 5.23: Computational times (in seconds) and objective function values of the modified speed-up algorithm 3 for (EEMPR-E) when n = 50

Instance	Random				NN			Best Obj.
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Value Overall
1	0	3725646	4725548	730.75	1	3555053	56.73	3555053
2	0	3770751	4525174	713.44	1	3442646	47.64	3442646
3	1	3869511	4222883	709.36	0	3890121	99.65	3869511
4	1	4071201	$27\text{x}10^{8}$	637.48	0	4377633	58.38	4071201
5	1	3866733	4433459	613.49	0	4124602	56.90	3866733
6	1	3727663	4257557	697.80	0	3799870	47.33	3727663
7	1	3336292	3879094	764.91	0	3448967	103.43	3336292
8	1	4161240	4601310	689.03	0	4978486	69.68	4161240
9	1	4192975	$14\text{x}10^{8}$	810.09	0	4517024	55.43	4192975
10	1	3870281	4505301	606.92	0	4208711	44.10	3870281

given as follows. The average solution time of all 9 random starts are calculated for instances when $n = 20$. Then for each instance, these averages are compared with the average times of 2-opt and 3-opt algorithm which were calculated in the same manner. Then the average of all these percent differences for 10 instances are calculated and observed as -88.389.

Table 5.24: Summary table for (EEMPR-T)

EEMPR-T											
n	% Difference	2-opt		3-opt		2-opt and 3-opt		Speed-up alg. CL = 0.2n		Modified speed-up alg. 3	
		Random	NN	Random	NN	Random	NN	Random	NN	Random	NN
20	Avg. % Difference of Best Objectives	0.000	0.003	0.000	0.002	0.000	0.001	-	-	-	-
	Avg. % Difference of Average Objective	0.242	-	0.005	-	0.005	-	-	-	-	-
	Avg. % Difference of Worst Ojectives	1.284	-	0.010	-	0.008	-	-	-	-	-
	Avg. % Difference of Times	-88.389	-519.875	89.989	60.751	0.000	0.000	-	-	-	-
30	Avg. % Difference of Best Objectives	0.001	0.005	87.969	0.001	0.001	0.001	0.143	1.702	0.649	11.500
	Avg. % Difference of Average Objective	0.094	-	100.000	-	0.009	-	2.810	-	29.124	-
	Avg. % Difference of Worst Ojectives	0.552	-	100.000	-	0.015	-	22.658	-	56.695	-
	Avg. % Difference of Times	-122.734	-707.479	73.543	72.934	0.000	0.000	-578.520	-5665.570	-2719.360	-12581.501
50	Avg. % Difference of Best Objectives	-	-	-	-	-	-	0.139	7.667	1.050	13.733
	Avg. % Difference of Average Objective	-	-	-	-	-	-	8.693	-	36.562	-
	Avg. % Difference of Worst Ojectives	-	-	-	-	-	-	28.790	-	53.549	-
	Avg. % Difference of Times	-	-	-	-	-	-	-	-	-	-

Table 5.25: Summary table for (EEMPR-E)

EEMPR-E											
n	% Difference	2-opt		3-opt		2-opt and 3-opt		Speed-up alg. CL = 0.2n		Modified speed-up alg. 3	
		Random	NN	Random	NN	Random	NN	Random	NN	Random	NN
20	Avg. % Difference of Best Objectives	0.000	0.087	0.000	0.000	0.000	0.000	-	-	-	-
	Avg. % Difference of Average Objective	0.199	-	0.000	-	0.000	-	-	-	-	-
	Avg. % Difference of Worst Ojectives	1.415	-	0.000	-	0.000	-	-	-	-	-
	Avg. % Difference of Times	-151.270	-505.712	86.699	57.235	0.000	0.000	-	-	-	-
30	Avg. % Difference of Best Objectives	0.000	0.301	60.056	0.000	0.000	0.000	8.552	3.929	9.189	6.596
	Avg. % Difference of Average Objective	1.214	-	100.000	-	0.000	-	19.956	-	32.873	-
	Avg. % Difference of Worst Ojectives	5.050	-	100.000	-	0.000	-	30.384	-	44.106	-
	Avg. % Difference of Times	-174.397	-704.063	64.335	70.852	0.000	0.000	-630.892	-5256.064	-2646.016	-8801.507
50	Avg. % Difference of Best Objectives	-	-	-	-	-	-	1.882	5.019	9.269	12.635
	Avg. % Difference of Average Objective	-	-	-	-	-	-	10.753	-	36.426	-
	Avg. % Difference of Worst Ojectives	-	-	-	-	-	-	20.843	-	45.968	-
	Avg. % Difference of Times	-	-	-	-	-	-	-	-	-	-

CHAPTER 6

CONCLUSION

In this thesis, we worked on energy efficient multi-place robot rendezvous problem, i.e., EEMPR, by considering campaign time restrictions. We examine two different objective functions. The first one aims to minimize the total energy consumption of the robots, i.e., EEMPR-E while the second one is to minimize the campaign time, i.e., EEMPR-T. We proposed a second order cone programming formulation to find the optimal set of rendezvous places for a given meeting order. We also implement improvement heuristics to find a better meeting order which improves the objective function value. Furthermore, in our computational studies it is realized that while utilizing improvement heuristics, computational times increase rapidly when the number of mobile worker robots increases. Hence, for cases where 30 or more mobile worker robots are used, we provide speed-up algorithms to decrease the solution times. We suggest to the reader that if there is no time restriction for the computational studies, combined 2-opt and 3-opt algorithm can be utilized for small size instances. However, if the computational time is limited, then 2-opt algorithm can be executed. For the large size instances, if time is not restrictive, the speed-up algorithm which is an extension of combined 2-opt and 3-opt algorithm with fixed radius search, candidate list and don't look bits approaches should be used by taking the candidate list length as $0.2n$. On the other hand, when there is a time restriction for the computational studies, modified speed-up algorithm 3 should be used which is an extension of the speed-up algorithm.

Overall, the contributions of this study can be listed as follows. We can say that this is the first study that

- takes campaign time restriction into account for minimizing energy consumption objective function,

- can handle a non-linear energy consumption function,
- allows robots to adjust their speeds,
- proposes an SOCP formulation which can handle non-linearity,
- considers TSP speed-up techniques for favorable solution times to be implemented for higher number of mobile worker robots,
- performs extensive computational experiments on the implementation of TSP improvement heuristics on EEMPR.

In this study, we assume multi-place robot rendezvous problem. Single-place robot rendezvous problem version by considering energy efficiency and campaign time restrictions can be considered in future studies. Also, the case where there are multiple tanker robots can be studied. In this case, there will also be assignment of mobile worker robots to the tanker robots. Last but not least, different variations of improvement heuristics can be performed. For instance, rather than looking at all the moves and find the best improvement in the heuristics algorithms, implementing the first improving move to the meeting order can be applied.

APPENDIX A

APPENDIX

Table A.1: Fixed radius search algorithm results for (EEMPR-T) when n = 50

Instance	Random				NN			Best Obj.
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Value Overall
1	1	13623	13904	8146.89	0	13715	310.70	13623
2	1	13230	13358	6375.72	0	13262	387.26	13230
3	1	13490	13533	8012.68	0	14070	3287.32	13490

Table A.2: Computational times (in seconds) and objective function values of the improvement heuristic algorithms for (EEMPR-T) when n = 5

Instance	2-opt							3-opt							2-opt + 3-opt								
	Random				NN			Random				NN			Random				NN				
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Best Obj. Value Overall	% Difference
1	9	1428	1428	0.34	1	1428	0.19	9	1428	1428	0.77	1	1428	0.43	9	1428	1428	0.66	1	1428	0.56	1428	0.00
2	9	2417	2417	0.35	1	2417	0.13	9	2417	2417	0.85	1	2417	0.46	9	2417	2417	0.67	1	2417	0.56	2417	0.00
3	7	1565	1565	0.29	1	1565	0.23	9	1565	1565	0.79	1	1565	0.79	9	1565	1565	0.65	1	1565	0.63	1565	0.00
4	9	2558	2558	0.35	0	2568	0.15	9	2558	2558	0.75	0	2568	0.37	9	2558	2558	0.63	0	2568	0.52	2558	0.00
5	9	1856	1856	0.26	1	1856	0.41	9	1856	1856	0.68	1	1856	0.72	9	1856	1856	0.55	1	1856	0.75	1856	0.00
6	9	2851	2851	0.35	1	2851	0.13	9	2851	2851	0.72	1	2851	0.31	9	2851	2851	0.61	1	2851	0.49	2851	0.00
7	9	2409	2409	0.35	1	2409	0.24	9	2409	2409	0.76	1	2409	0.68	9	2409	2409	0.66	1	2409	0.63	2409	0.00
8	9	3269	3269	0.32	1	3269	0.21	9	3269	3269	0.69	1	3269	0.69	9	3269	3269	0.62	1	3269	0.64	3269	0.00
9	9	1943	1943	0.28	1	1943	0.13	9	1943	1943	0.65	1	1943	0.33	9	1943	1943	0.56	1	1943	0.49	1943	0.00
10	9	1961	1961	0.36	1	1961	0.23	9	1961	1961	0.79	1	1961	0.67	9	1961	1961	0.64	1	1961	0.57	1961	0.00

Table A.3: Computational times (in seconds) and objective function values of the improvement heuristic algorithms for (EEMPR-E) when n = 5

Instance	2-opt							3-opt							2-opt + 3-opt								
	Random				NN			Random				NN			Random				NN				
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Best Obj. Value Overall	% Difference
1	9	346122	346122	0.39	1	346122	0.41	9	346122	346122	0.95	1	346122	0.91	9	346122	346122	0.81	1	346122	0.92	346122	0.00
2	9	530033	530033	0.38	1	530033	0.16	9	530033	530033	0.96	1	530033	0.44	9	530033	530033	0.77	1	530033	0.63	530033	0.00
3	9	384451	384451	0.38	1	384451	0.36	9	384451	384451	0.89	1	384451	0.88	9	384451	384451	0.74	1	384451	0.68	384451	0.00
4	9	523817	523817	0.38	1	523817	0.17	9	523817	523817	0.85	1	523817	0.44	9	523817	523817	0.76	1	523817	0.61	523817	0.00
5	9	396671	396671	0.37	1	396671	0.47	9	396671	396671	0.87	1	396671	0.81	9	396671	396671	0.73	1	396671	1.01	396671	0.00
6	9	609573	609573	0.35	1	609573	0.17	9	609573	609573	0.79	1	609573	0.36	9	609573	609573	0.69	1	609573	0.61	609573	0.00
7	9	544407	544407	0.38	1	544407	0.29	9	544407	544407	0.79	1	544407	0.70	9	544407	544407	0.70	1	544407	0.82	544407	0.00
8	9	647297	647297	0.39	1	647297	0.28	9	647297	647297	0.84	1	647297	0.74	9	647297	647297	0.73	1	647297	0.88	647297	0.00
9	9	396223	396223	0.40	1	396223	0.16	9	396223	396223	0.84	1	396223	0.38	9	396223	396223	0.75	1	396223	0.63	396223	0.00
10	9	437072	437072	0.37	1	437072	0.17	9	437072	437072	0.85	1	437072	0.36	9	437072	437072	0.71	1	437072	0.60	437072	0.00

Table A.4: Computational times (in seconds) and objective function values of the improvement heuristic algorithms for (EEMPR-T) when n = 15

Instance	2-opt							3-opt							2-opt + 3-opt								
	Random				NN			Random				NN			Random				NN				
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Best Obj. Value Overall	% Difference
1	5	6094	6096	56.19	1	6094	6.17	9	6094	6094	637.68	1	6094	84.55	9	6094	6094	118.80	1	6094	48.50	6094	0.00
2	8	5750	5755	49.98	1	5750	8.87	9	5750	5750	586.26	1	5750	90.40	9	5750	5750	97.08	1	5750	57.16	5750	0.00
3	9	5181	5181	62.59	1	5181	5.66	9	5181	5181	704.24	1	5181	92.85	9	5181	5181	104.52	1	5181	52.80	5181	0.00
4	9	5077	5077	56.30	1	5077	11.36	9	5077	5077	670.73	1	5077	139.23	9	5077	5077	99.42	1	5077	59.92	5077	0.00
5	1	4872	4878	41.56	0	4877	10.88	9	4872	4872	524.95	1	4872	180.15	9	4872	4872	136.64	1	4872	105.80	4872	0.00
6	3	5306	5309	50.91	0	5316	29.40	9	5306	5306	702.95	1	5306	348.04	7	5306	5306	115.59	1	5306	128.44	5306	0.00
7	7	5401	5403	54.84	1	5401	5.76	7	5401	5401	687.86	1	5401	92.82	8	5401	5401	108.38	1	5401	53.66	5401	0.00
8	1	4742	4770	52.70	0	4743	5.41	0	4743	4743	698.93	0	4743	94.39	1	4742	4743	112.76	0	4743	54.14	4742	0.00
9	8	4971	4976	48.05	1	4971	10.35	9	4971	4971	672.60	1	4971	141.13	9	4971	4971	102.67	1	4971	59.56	4971	0.00
10	9	4885	4885	49.41	1	4885	10.54	9	4885	4885	656.72	1	4885	141.68	9	4885	4885	94.49	1	4885	60.07	4885	0.00

Table A.5: Computational times (in seconds) and objective function values of the improvement heuristic algorithms for (EEMPR-E) when n = 15

Instance	2-opt							3-opt							2-opt + 3-opt								
	Random				NN			Random				NN			Random				NN				
No	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	# Best	Best Obj. Value	Avg. Obj. Value	Avg. Time	# Best	Obj. Value	Time	Best Obj. Value Overall	% Difference
1	9	1220173	1220173	64.99	1	1220173	8.07	9	1220173	1220173	761.82	1	1220173	104.09	9	1220173	1220173	119.15	1	1220173	64.97	1220173	0.00
2	4	1298212	1311938	54.23	0	1319713	19.83	9	1298212	1298212	692.23	1	1298212	153.88	9	1298212	1298212	153.25	1	1298212	131.49	1298212	0.00
3	9	1045929	1045929	67.64	1	1045929	3.97	9	1045929	1045929	886.43	1	1045929	53.70	9	1045929	1045929	129.27	1	1045929	63.37	1045929	0.00
4	9	1262617	1262617	64.17	1	1262617	11.20	9	1262617	1262617	722.42	1	1262617	104.04	9	1262617	1262617	125.17	1	1262617	68.70	1262617	0.00
5	7	1242703	1255349	47.58	1	1242703	15.19	9	1242703	1242703	649.67	1	1242703	215.90	9	1242703	1242703	157.51	1	1242703	74.42	1242703	0.00
6	9	1136721	1136721	59.48	1	1136721	31.86	9	1136721	1136721	730.51	1	1136721	313.40	9	1136721	1136721	123.49	1	1136721	86.96	1136721	0.00
7	6	1232549	1254075	61.53	1	1232549	3.45	9	1232549	1232549	765.40	1	1232549	50.11	9	1232549	1232549	159.60	1	1232549	59.57	1232549	0.00
8	9	927141	927141	63.19	1	927141	11.16	9	927141	927141	804.26	1	927141	160.44	9	927141	927141	135.25	1	927141	70.09	927141	0.00
9	9	1077743	1077743	57.21	1	1077743	15.64	9	1077743	1077743	684.72	1	1077743	167.92	9	1077743	1077743	133.28	1	1077743	73.17	1077743	0.00
10	9	1024586	1024586	57.24	1	1024586	11.64	9	1024586	1024586	696.66	1	1024586	171.98	9	1024586	1024586	138.50	1	1024586	73.32	1024586	0.00

REFERENCES

[1] Ifr, "Industrial robots: Robot investment reaches record 16.5 billion usd," Sep 2019.

[2] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to autonomous mobile robots*. MIT press, 2011.

[3] L. Armesto, V. Girbés, M. Vincze, S. Olufs, and P. Munoz-Benavent, "Mobile robot obstacle avoidance based on quasi-holonomic smooth paths," in *Conference Towards Autonomous Robotic Systems*, pp. 244–255, Springer, 2012.

[4] N. Mathew, S. L. Smith, and S. L. Waslander, "A graph-based approach to multi-robot rendezvous for recharging in persistent tasks," in *2013 IEEE International Conference on Robotics and Automation*, pp. 3497–3502, IEEE, 2013.

[5] W. Guanghua, L. Deyi, G. Wenyan, and J. Peng, "Study on formation control of multi-robot systems," in *2013 Third International Conference on Intelligent System Design and Engineering Applications*, pp. 1335–1339, IEEE, 2013.

[6] A. Gautam and S. Mohan, "A review of research in multi-robot systems," in *2012 IEEE 7th International Conference on Industrial and Information Systems (ICIIS)*, pp. 1–5, IEEE, 2012.

[7] F. E. Schneider and D. Wildermuth, "Using robots for firefighters and first responders: Scenario specification and exemplary system description," in *2017 18th International Carpathian Control Conference (ICCC)*, pp. 216–221, IEEE, 2017.

[8] R. R. Murphy, "Trial by fire [rescue robots]," *IEEE Robotics & Automation Magazine*, vol. 11, no. 3, pp. 50–61, 2004.

[9] A. Shukla and H. Karki, "Application of robotics in offshore oil and gas industry—a review part ii," *Robotics and Autonomous Systems*, vol. 75, pp. 508–524, 2016.

[10] 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.

[11] S. Alpern, "The rendezvous search problem," *SIAM Journal on Control and Optimization*, vol. 33, no. 3, pp. 673–683, 1995.

[12] S. Alpern and S. Gal, *The theory of search games and rendezvous*. Kluwer Academic Publishers, Norwell, 2003.

[13] N. Roy and G. Dudek, "Collaborative Robot Exploration and Rendezvous: Algorithms, Performance Bounds and Observations," *Autonomous Robots*, vol. 11, no. 2, pp. 117–136, 2001.

[14] M. Meghjani and G. Dudek, "Multi-robot exploration and rendezvous on graphs," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5270–5276, IEEE, 2012.

[15] Y. Litus, R. T. Vaughan, and P. Zebrowski, "The Frugal Feeding Problem: Energy-efficient, multi-robot, multi-place rendezvous," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 27–32, IEEE, 2007.

[16] P. Zebrowski, Y. Litus, and R. T. Vaughan, "Energy Efficient Robot Rendezvous," in *Fourth Canadian Conference on Computer and Robot Vision (CRV'07)*, pp. 139–148, IEEE, 2007.

[17] M. C. Silverman, D. Nies, B. Jung, and G. S. Sukhatme, "Staying alive: A docking station for autonomous robot recharging," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 1, pp. 1050–1055, IEEE, 2002.

[18] P. Zebrowski and R. T. Vaughan, "Recharging robot teams: A tanker approach," in *ICAR'05. Proceedings., 12th International Conference on Advanced Robotics, 2005.*, pp. 803–810, IEEE, 2005.

[19] M. A. Lanthier, D. Nussbaum, and T.-J. Wang, "Calculating the meeting point of scattered robots on weighted terrain surfaces," in *Proceedings of the 2005 Australasian symposium on Theory of computing-Volume 41*, pp. 107–118, 2005.

[20] Y. Litus, P. Zebrowski, and R. T. Vaughan, "A distributed heuristic for energy-efficient multirobot multiplace rendezvous," *IEEE Transactions on Robotics*, vol. 25, no. 1, pp. 130–135, 2008.

[21] P. Tokekar, N. Karnad, and V. Isler, "Energy-optimal velocity profiles for car-like robots," in *2011 IEEE International Conference on Robotics and Automation*, pp. 1457–1462, IEEE, 2011.

[22] F. Alizadeh and D. Goldfarb, "Second-order cone programming," *Mathematical programming*, vol. 95, no. 1, pp. 3–51, 2003.

[23] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, "Applications of second-order cone programming," *Linear algebra and its applications*, vol. 284, no. 1-3, pp. 193–228, 1998.

[24] G. Reinelt, *The traveling salesman: computational solutions for TSP applications*. Springer-Verlag, 1994.

[25] T. Tachibana and M. Adachi, "A method for solving asymmetric traveling salesman problems using block shift operation and chaotic neurodynamics," *IEEJ Transactions on Electronics, Information and Systems*, vol. 132, no. 5, pp. 774–781, 2012.

[26] G. Sierksma, "Hamiltonicity and the 3-opt procedure for the traveling salesman problem," *Applicationes Mathematicae*, vol. 22, pp. 351–358, 1994.

[27] J. J. Bentley, "Fast algorithms for geometric traveling salesman problems," *ORSA Journal on computing*, vol. 4, no. 4, pp. 387–411, 1992.

[28] H. H. Hoos and T. Stützle, *Stochastic local search: Foundations and applications*. Elsevier, 2004.