Titre: Title:	An Adaptive Large Neighborhood Search Algorithm for Tactical Planning of a Single-Segment Corridor Network in a Many-to-one-to-many Transportation System
Auteur: Author:	Melahat Khodemaniyazdi
Date:	2022
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Khodemaniyazdi, M. (2022). An Adaptive Large Neighborhood Search Algorithm for Tactical Planning of a Single-Segment Corridor Network in a Many-to-one-to-many Transportation System [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/10707/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/10707/
Directeurs de recherche: Advisors:	Michel Gendreau, Téodor G. Crainic, & Walter Rei
Programme: Program:	Maîtrise recherche en génie industriel

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

An Adaptive Large Neighborhood Search Algorithm for Tactical Planning of a Single-Segment Corridor Network in a Many-to-one-to-many Transportation System

MELAHAT KHODEMANIYAZDI

Département de mathématiques et de génie industriel

présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*

Génie industriel

Novembre 2022

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Ce mémoire intitulé :

An Adaptive Large Neighborhood Search Algorithm for Tactical Planning of a Single-Segment Corridor Network in a Many-to-one-to-many Transportation System

présenté par **Melahat KHODEMANIYAZDI**

en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*

a été dûment accepté par le jury d'examen constitué de :

Louis-Martin ROUSSEAU, président

Michel GENDREAU, membre et directeur de recherche

Téodor G. CRAINIC, membre et codirecteur de recherche

Walter REI, membre et codirecteur de recherche

Antoine LEGRAIN, membre

DEDICATION

I dedicate my work to my parents.

ACKNOWLEDGEMENTS

First of all, I would like to thank Professor Michel Gendreau, Professor Teodor Gabriel Crainic and Professor Walter Rei, my research director and co-research directors, for their support, advice, and financial support. They offered me an interesting project, introduced me to the world of freight transportation systems, and allowed me to have unique experiences for development of Operations Research techniques for solving an optimization model in large-scale instances using heuristics and metaheuristics. I also thank my mother and father for everything they have done for me.

RÉSUMÉ

Le transport de marchandises est un processus physique de transport de marchandises et d'expéditions entre différents terminaux par voie maritime, aérienne ou terrestre entre différentes zones. Cette thèse porte sur un réseau Many-to-One-to-Many (M1M), un cadre décisionnel où un côté du système comprend de nombreux expéditeurs, producteurs, grossistes, distributeurs, prestataires de services logistiques, etc. -demandes de demande de transport efficace en termes de coûts et de temps de chargements de produits. De l'autre côté du système M1M, de nombreux transporteurs font des offres de capacité de transporteur pour le transport, demandant des chargements rentables. Dans ce contexte, les transporteurs peuvent être des fournisseurs de services de transport de divers modes, par exemple, les chemins de fer, les compagnies aériennes, la navigation fluviale et océanique, et différents types, par exemple, des transporteurs complets ou partiels. Le modèle proposé considère un réseau de corridor à segment unique comprenant de nombreuses demandes de demande des expéditeurs à transférer d'un terminal d'origine à un terminal de destination, qui ont tous deux une capacité d'entreposage à toutes les périodes. Pour résoudre le modèle proposé, une solution initiale est construite par un algorithme heuristique. Ensuite, cette solution est améliorée par une Recherche Adaptive Large Voisinage (ALNS) combinant une recherche locale utilisant huit opérateurs de suppression pour détruire une solution et quatre opérateurs d'insertion pour la réparer. Enfin, l'algorithme proposé est testé sur différents benchmarks et analysé avec soin pour évaluer les performances de chaque opérateur et de l'algorithme heuristique constructif. L'une des principales conclusions de la comparaison d'algorithmes est que l'ALNS proposé peut obtenir des solutions avec un écart d'optimalité de 11 % pour résoudre de très grands ensembles de données en moins d'une minute. Cependant, le solveur exact peut trouver les solutions en une heure environ pour résoudre les données mentionnées ou ne peut pas les obtenir dans les délais. Les solutions obtenues par ALNS montrent des écarts d'environ 10%, et ce fait signifie que l'ALNS n'a pas été très efficace pour résoudre notre problème.

ABSTRACT

Freight transportation is a physical process for the transportation of commodities and shipments among different terminals by sea, air, or land between different zones. This thesis focuses on a Many-to-One-to-Many (M1M) networks, a decision-making framework where one side of the system includes many shippers, producers, wholesalers, distributors, logistics service providers and so on, making shipper-demand requests for cost and time-efficient transportation of product loads. On the other side of the M1M system, many carriers make carrier-capacity offers for transportation, requesting profitable loads. In this context, carriers may be transportation service providers of diverse modes, e.g., railroads, airlines, river and ocean navigation, and different types, e.g., full or less-than-truckload carriers. The proposed model considers a single-segment corridor network including many shipper-demand requests to be transferred from an origin terminal to a destination terminal, both of which have a warehousing capacity in all time periods. To solve the proposed model, an initial solution is constructed by a heuristic algorithm. Then, this solution is improved by an Adaptive Large Neighborhood Search (ALNS) combining a local search using eight removal operators to destroy a solution and four insertion operators to repair it. Finally, the proposed algorithm is tested on different benchmarks and analyzed carefully to assess the performance of each operator and of the constructive heuristic algorithm. One main finding of the algorithm comparison is that the proposed ALNS can achieve solutions with an 11% optimality gap for solving very large data sets in less than one minute. However, the exact solver either can find the solutions in around one hour for solving the mentioned data or cannot get them in time limitation. The solutions obtained by ALNS shows gaps around 10% and this fact represents that the ALNS was not very efficient for addressing our problem.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

ALNS	Adaptive Large Neighborhood Search
IDSP	Intelligent Decision Support Platform
M1M	Many-to-One-to-Many
RPD	Relative Percentage Deviation
SA	Simulated Annealing
S/N	Signal to Noise
SSND	Scheduled Service Network Design

CHAPTER 1 INTRODUCTION

1.1 Motivation

Nowadays, there is a great deal of interest in the development of optimization models and intelligent solution algorithms to tackle real-world transportation and logistics problems [1]. To improve the performance of logistics activities, such studies are highly important. For example, in 2019, around 18 billion tons of commodities, as reported by U.S. Department of Transportation [2], with a value of 18.5$ trillion, were moved by logistics companies and the role of optimization models and intelligent solution algorithms is undeniable.

The business environment is very competitive for businesses especially for logistics companies. In this regard, the speed in designing, manufacturing, and distributing products, transportation, and logistics activities, forces them to always look for ways to improve operations, planning and coordination of supply chain members [3]. Planning and scheduling processes in the transportation and logistics network, can reduce the cost, and many researchers and industrialists made efforts to develop efficient planning tools for logistics and transportation activities [4]. Hence, the development of new and practical optimization models and solution algorithms can help companies to create decision support systems to improve their performance in operations and stay competitive. This study is based on the idea of Many-to-One-to-Many (M1M) systems proposed earlier by Crainic et al. [49] and Taherkhani et al. [10]. The M1M system includes carriers, shippers, and an Intelligent Decision Support Platform (IDSP) for handling different decision-making problems for a freight transportation system. While the supply side for the M1M system is the carriers that offer different services for transportation of shipments, the demand side is the shippers. We integrate the decisions from both sides of M1M system using an IDSP for making the tactical planning on a single-segment corridor network. The IDSP automates the planning and optimizing the M1M system to increase the profitability while satisfying the demands from both stakeholders. The inputs to the IDSP are the time-dependent shipper-demand requests and the carrier-capacity offers which are available at different time periods. After receiving the requests, the IDSP optimizes operations to accept or reject request as well as selected services, the assignment of shipments to these services, and create itineraries of accepted requests through the time-space network. With regards to the carriers, a transportation service has some characteristics including the origin, destination, capacity of vehicles and the carrier-predefined timetable as well as the fixed cost, variable costs, duration, and a time attribute as scheduling. In the proposed transportation system, we have a

capacity limitation for the execution arcs and the capacity of terminals. A service network design specifies the movements of shipments through a time-space network for the transportation of time-dependent shipper-demand requests by a single-leg service. The time-stamped service arcs are called service legs. Each service has a departure time from its origin and an arrival time to its destination.

The last characteristic of each service is the duration from departure time to the arrival time to represent the total travel time. The travel time is directly related to distance between the origin and destination terminals. Based on these time attributes of each transportation service, one decision is to create a set of scheduled services which can be used to transfer a shipper-demand request between two terminals. The scheduled services are divided into fast or regular services. As may be understood from their names, a fast service has the lowest travel time in comparison with the regular services.

Another side of M1M system is the shippers which are divided into two types, i.e., contract-based, and non-contract. Most studies in the literature, only consider one type of shipper-demand request known as contract-based requests. In this case, there is a long-term contract ensuring the required capacity for the carriers. In a real-world setting, there are also some irregular potential shippers in addition to the contract-based shippers; the requests of shippers are called non-contract-based requests; If both types of requests are accepted, their shipments are transferred from their origins to their destinations. Based on these facts, one challenging optimization decision is to select non-contract-based requests to increase the maximum profit.

In addition to the classification of shippers, the demand is divided into two types, i.e., standard, and urgent. Based on their names, the duration delivery for the urgent requests must be lower than for the standard ones. Therefore, each shipment from the side of shippers, may be a contract-based or non-contract and the demand may be classified as urgent or standard. The shipments from different shippers are consolidated and transferred to the vehicles for addressing the selection of shipper-

demand requests among contract and non-contract ones, and the choice itineraries of shipments and routes of carriers through a time-space network. For both types of requests, if the shipper-demand request is accepted, the shipment is picked up at its origin and delivered to its destination within the predefined time window by the selected scheduled services.

Based on the both types of requests, the proposed optimization model makes this decision to accept contract-based and non-contract requests where there is a large penalty for unaccepted contract-based requests. In this regard, another decision is to consider the opportunity cost for unaccepted requests based on revenue management. The last decision is to design the itinerary of each accepted shipment to maximize the revenue while avoiding incurring penalties related to the time windows of shipments. All these activities must be managed at a single-segment corridor network design problem.

1.2 Research questions

This thesis focuses on the development of a single-segment corridor network using an optimization model and a metaheuristic algorithm. The single-segment corridor network is defined by a set of shipper-demand requests which must be transferred from the origin to the destination where the selection of contract-based or non-contract requests, assignment of services and the planning of itineraries of shipments, are the main decision variables. Based on this fact, this research aims to address the following research questions:

- Using the decision variables from our optimization model, which one of requests should be accepted? Is it possible to accept all shipper-demand requests?
- Using the decision variables from our optimization model, for each accepted shipper-demand request, which transportation services should be assigned to them? What is the optimal itinerary from the origin to the destination?
- Using the proposed metaheuristic algorithm, how we can solve the proposed problem in large-scale transportation networks?
- Using the proposed metaheuristic algorithm, how we can destroy a solution and construct it efficiently?
- Using the proposed metaheuristic algorithm, how we can find a new optimal solution and escape from the local optimum solution?

1.3 Research objectives

To address the aforementioned research questions, this research has the following objectives:

- Adapting the optimization model of multi-stakeholder freight transportation system for the single-segment corridor network.

- Considering different contract-based or non-contract-based requests which should be transferred from the origin terminal to the destination terminal.
- Contributing to the concept of ad-hoc arcs or the extra capacity for unaccepted shipper-demand requests.
- Based on revenue management concepts, the objective is to maximize the net profit for all accepted requests, while considering the fixed costs, transportation costs and penalty costs if the schedule does not meet the time window.
- Proposing an efficient metaheuristic algorithm using adaptive large neighborhood search algorithm, simulated annealing, and the local search operator to solve the model.

1.4 Expected contribution

Although our concepts for the development of an optimization model for a multi-stakeholder freight transportation system considering different shippers, carriers and demand requests which can be a contract-based or non-contract-based request, have been introduced earlier, the main contribution of this research is to introduce an efficient metaheuristic algorithm based on adaptive large neighborhood search for addressing a single segment corridor network design problem in a many-to-one-to-many system. This algorithm uses a strong constructive heuristic algorithm and set of removal and insertion operators as well as a decision rule from the simulated annealing in addition to a local search operator as subloop. The proposed model defines for the first time the opportunity cost for unaccepted requests to consider the concept of revenue management comprehensively.

1.5 Summary of other chapters

This introduction chapter is followed by five chapters. In Chapter 2, we describe the problem context and define all elements of the problem statement including freight transportation, IDSP, planning levels, physical network, time-space network, many-to-one-to-many systems, shippers, and carriers. Chapter 3 aims to provide a literature review to define the basic models and solution algorithms in the literature as well as studies and research gaps. Chapter 4 establishes an optimization model to illustrate the problem definition mathematically. Chapter 5 introduces an efficient adaptive large neighborhood search algorithm. Chapter 6 does an extensive analysis to validate the solutions of our metaheuristic and compares it while analyzing the performance of its

components. In addition, some sensitivity analyses are performed on the key parameters of the applied optimization model to show its efficiency for real-world settings. Finally, Chapter 7 provides a conclusion for this paper with findings, limitations, and recommendations for the future works.

CHAPTER 2 PROBLEM SETTING AND DEFINITIONS

The proposed problem aims to model an effective tactical planning for transportation over a single-segment corridor M1M system. A decision-making framework, known as an IDSP, is implemented for this M1M where the demand side is a set of shippers, and the supply side is a set of carriers. The goal of IDSP is to coordinate these shippers and carriers while making the decisions for the selection of contract-based and non-contract requests, the assignment of shipments to the selected services and make itinerary of requests.

The proposed single-segment corridor network includes only one terminal as the origin and another terminal for the destination. The supply side of our single-segment corridor network is the set of carriers that offer the services to move the shipments from the origin to the destination over the time-space network. Another side of M1M is the shippers where each shipper-demand request which should be transferred from its origin to its destination is characterized by some time and economic attributes where the volume, origin, and destination and time window indicating a release time, for the pickup time and the due date for the delivery time can be considered. Furthermore, the shipper-demand requests are divided into two groups, i.e., standard, and urgent. The duration delivery of urgent requests is lower than for the standard ones.

In real-world setting, it might not be possible to satisfy all shipper-demand requests within their specified time window. This study allows requests to be shipped earlier or later than the time window with a penalty cost. If the created schedule does not meet the time window for a shipment, there is an early or late pick-up, or an early or late delivery time. A penalty cost is defined to address these situations accordingly. Based on revenue management concepts, this study aims to maximize the net profits from accepted shipper-demand requests by assigning them to the selected services.

In conclusion, the optimization model creates a tactical planning level for the demand side (shippers) and the supply side (carriers) with the aim of maximizing the profitability of the system. The demand side aims to select the shippers for non-contract requests to increase the net profit based on the remaining capacity and satisfy them. It is that the proposed model can accept or reject both contract-based and non-contract requests where there is a big penalty of unaccepted contract-based requests. This thesis aims to model a single-segment corridor network design where the possibility of extra capacity or ad-hoc arcs, is contributed for unaccepted requests. The goal is to select the individual services and to design the itineraries based on the services capacity, duration,

and other attributes. These decisions need some input parameters including the net profit, time window, volume, origin, and destination for both contract-based and non-contract requests.

In this chapter, we focused on the main elements of proposed problem including a freight transportation in Section 2.1. Then, we explain our M1M system generally with a focus on the proposed single-segment corridor network in Section 2.2. The shipper-demand requests have been explained in Section 2.3. The carrier-capacity offers are studied in Section 2.4. Finally, the planning levels are illustrated in Section 2.5.

2.1 Freight transportation

The proposed single-segment corridor network is a customized freight transportation system which is generally a process for the transportation of commodities and shipments among different terminals by sea, air, or land from different zones. In this process, many carriers and shippers should be coordinated to improve the economic attributes of freight transportation which is responsible for a substantial increase in air pollution that originates from fossil fuels. With the growing concern towards the environment and to sustain nature, various measures are taken to reduce harmful gas emissions [1-4]. Besides, the selection of transportation services considering different vehicles and air transport facilities has a significant factor to optimize the cost of the system [3-8]. Based on these difficulties, the optimization of transportation systems is one of the challenging issues in both developed and developing countries.

Optimization models for the freight transportation usually minimize the transportation costs while addressing the real-life constraints for transferring shipments from different origins to different destination by vehicles, trucks, ships, trains, or planes [9-12]. If a freight transportation system has more than one mode of transportation, it is called a multi-modal freight transportation system. The main difficulty is to manage all decisions in such systems while improving the quality of solutions and meeting the freight transportation's constraints [13-15]. In this regard, the role of a logistics company is undeniable to improve the performance of transportation operations [3-4]. In 2019, more than 17.9 billion tons of commodities with a value around $18.2 trillion were moved by logistics companies [5-6]. To get closer to real-world setting, a logistics company should work with different stakeholders and schedule the shipper-demand requests from shippers and offer the carrier-capacity planning to define the routing decisions regarding the time windows for sending a

set of commodities from origins to destinations [6]. Generally, the freight transportation network operates as follows:

- A shipper-demand request is picked up at a shipper's location.
- Then, this shipment is transported to one of the origin terminals.
- Lads brought at terminals may be either transferred (unloaded, cross-dock moved and loaded) to an outgoing vehicle or classified (re-classified, eventually unloaded, sorted, moved, and loaded into the outgoing vehicle).
- Next, the request is transported from its origin terminals to its destination terminals.
- At the destination terminal, the shipments are unloaded, freight is potentially separated and loaded into vehicles for efficient "local" delivery.

2.2 M1M system in a single-segment corridor network

Here, a general definition of the M1M system is explained and then it is customized for the proposed single-segment corridor network. The graphical presentation of the M1M system can be shown in Figure 2.1. It comprises three main elements including shippers, carriers, and a decision maker as the IDSP [13-17] illustrated in Figure 2.1

On the one side of the M1M system, many shippers, producers, wholesalers, distributers, logistics-service providers and so on, make the shipper-demand requests for cost and time-efficient transportation for product loads. Warehousing services will be included in later problem setting. Each shipment is to sort at a given shipper location and must travel to be delivered to a consignee location. Notice that IDSP is not explicitly represented in the problem setting since its elements are explained here.

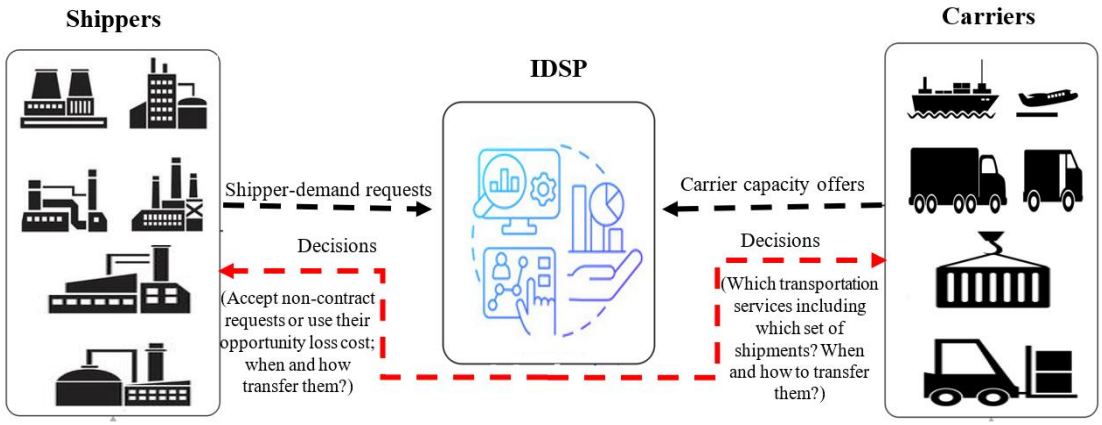


Figure 2-1 The M1M system structure, framework and decisions in the logistics company [10].

for the transportation, requesting profitable loads. In this context, carriers may be transportation service providers of diverse modes, e.g., railroads, airlines, river, and ocean navigation. In conclusion, the coordination of shippers and carriers in the M1M system, is made by the IDSP. The graphical presentation of this IDSP framework and our decisions is shown in Figure 2.1 to illustrate the connections for the IDSP with shippers and carriers to make the decisions while monitoring all transportation activities simultaneously. In addition, carriers are providing different transportation services with diverse modes and types like carriers of motors, maritime, rivers, airlines and railroads while offering different capacities for the transportation [16-18]. Each shipment starts from the origin terminal and ends to the destination terminal. Large shipper-demand requests usually sign long-term contracts and classified as the contract-based requests, and they must be accepted [19-20]. Carriers should have enough capacity for the transportation services to cover them. However, small shipper-demand requests usually do not sign a long-term contract but may use transportation services if they are accepted [21-23]. The IDSP must provide transportation services which are offered by the carriers for all accepted shipper-demand requests even they are non-contract requests.

Possibly, there is conflict of interest between shippers and carriers [23-24]. High-quality transportation services which are safety and able to meet the time window of shipper-demand requests. However, the carriers would like to increase their revenue to accept more requests with less transportation services [24-28]. The IDSP is an automated planning platform for the M1M system to satisfy all the shippers, carriers, and platform's interests. This system optimizes a

mathematical model to accept requests while defining the itineraries of shipments with regards to their predefined time window.

To customize the M1M system for the proposed problem, the introduced single-segment corridor network has efficiency and profitability which are resulted from sharing resources in an integrated IDSP. Having the best system performance yielding a four-win situation, the IDSP helps the carriers to earn additional revenue instead of moving air, the shippers to see their delivery costs decrease, the intermediary to earn on each transaction and the society to benefit from consolidation through less vehicles on the road and less pollution.

The proposed problem focuses on a single-segment corridor network where the shipments are transferred from the origin terminal to the destination terminal. It is a type of M1M system where the possibility of extra capacity or ad-hoc services exist. The proposed network includes two terminals which are the origin and the destination. There are several shippers as the demand side of M1M system where they have a set of shipments which should be transferred from one region to another region. Transportation activities takes place between two major terminals, where one of them is the origin and another one is the destination terminal to deliver the shipments. The pickup and distribution activities within these terminals are not explicitly considered in the problem setting. Carriers are the supply side of the M1M system that are the service providers to handle all transportation activities in the system. To manage both sides of M1M system and making the tactical decisions, the IDSP is developed to plan and optimize operations, improve profitably and simultaneously satisfy the needs of both categories of stakeholders.

2.3 Shipper-demand requests for transportation

In the proposed problem, we have defined the customers as the shippers who are on the demand side of our M1M system. Shipper-demand requests are characterized by their volume, origin, and destination. It is assumed that the demand of each shipper request is non-split and, accordingly, the accepted shipments are picked up from their origin and delivered to their destination as a whole. One main characteristic of accepted shipper-demand requests is the time window to indicate the release time, when the shipper makes the request available for the pickup and the due date which is the preferable delivery time for requests [28-30].

In the problem under study, the shippers are divided into two types, i.e., contract-based, and non-contract. Most studies in the literature, only consider one type of shipper-demand requests, known

as contract-based requests [31-33]. In this case, there is a long-term contract to let the IDSP ensure the required capacity for the carriers. In a real-world setting, there are some irregular potential shippers in addition to the contract-based shippers and in this respect, there is a short-notice between these shippers and the IDSP, therefore, these shippers are called as non-contract-based requests. If both types of requests are accepted, their shipments are transferred from their origins to their destinations. In addition to the classification of shippers, the demand is divided into two types, i.e., standard, and urgent. According to their names, the duration delivery in the urgent requests is lower than the standard ones. Therefore, each shipment from the side of shippers, may be a contract-based or non-contract and the demand may be classified as urgent or standard [32-35].

As occurred in real-world setting, it is not possible to satisfy all demand requests within their specified time window. In this regard, this study considers a soft time-window to allow carriers to move earlier or later than the time window with a penalty cost. Such information is provided by shippers to the IDSP. If the generated schedule does not meet the time window for a shipment, in this case, there is an early or late pick-up or an early or late delivery time. In the problem under study, the IDSP should pay the penalty costs to the shippers. Based on the revenue management concept, the IDSP tries to set a schedule to meet the time window for all accepted shipper-demand requests as much as possible. It is obvious that the penalty cost is lower than the net profit for each shipper-demand request [36-38].

In conclusion, this study aims to maximize the revenue among all accepted shipper-demand requests in the IDSP. There is a total revenue as the net profit for each shipper-demand request. This revenue depends on some factors including the demand types (which are urgent or standard), product characteristics (e.g., weight volume of shipments) and the distance between the origin and the destination terminals. The objective function covers the opportunity cost where unaccepted non-contract requests are considered as a benefit based on the revenue management concept.

2.4 Carrier capacity offers

In the considered single-segment corridor network, the supply side is carriers offering different services for transportation of shipments. Each transportation service has some characteristics including its origin, destination, capacity, duration, and timetable attributes, i.e., arrival and departure times as well as economic attributes, e.g., fixed, and variable costs [39-42]. The main

difference of a single-segment corridor with other networks is that all the shipments are transferred from the origin terminal to the destination terminal by services that have only one leg as single-leg services.

In the proposed transportation system, we have three types of capacity limitations, i.e., the capacity of origin and destination terminals as well as transportation services' capacity [41-44]. A service network design specifies the movements of shipments through a time-space network for the transportation of time-dependent requests [45-46]. As such, each service is defined in a time-space network labelled by different time and economic attributes. In the proposed single-segment corridor, we have different arcs corresponding to different services connecting the origin and the destination in a time-space network. A time-stamped service leg is based on the carrier-predefined timetable to show departure and arrival times. Each service leg has a departure time from its origin and an arrival time to its destination. Another characteristic of each service is the duration from departure time to the arrival time to represent the total travel time [47-48].

By another point of view, the travel time in the single-segment corridor is directly related to the time for transferring from one origin to one destination. Based on the travel time of services, each service has a set of time attributes as named as a scheduled service. There are two types of scheduled services, i.e., fast, or regular services. What we can understand from their names is that a fast service is quicker than the regular ones while a fast service is more expensive than a regular one and these differences provide diverse economic solutions for the IDSP [49- 50].

The economic performance of carrier capacity offers is assessed by the total cost including two terms, i.e., a fixed cost for selecting a service and its capacity as well as a unit of transportation which is variable based on the volume of shipments offered by shippers. All these costs are paid from the IDSP to the carriers who offered all services. It should be noted that the origin and the

destination terminals have the possibility of warehousing to storage the shipments which are not handled yet by the services, or we should wait for their availability time based on the time window of shipments.

2.5 Tactical planning of M1M system

The plan of IDSP optimizes the operations and transportation activities of M1M system where in this study, a tactical planning aims to propose economic solutions for the medium-term decisions for accepted shipper-demand requests, the assignment of shipments to transportation services and

designing the itineraries of shipments. The tactical planning helps to simplify the operations of M1M system while improving its profitability. By another point of view, the tactical level planning is directly related to the time representation where multi-period networks are operations research tools and instruments to model as well as address the planning level. Academically, these information and decisions, are made up the planning level to make medium-term planning horizon in a single-segment corridor network.

To study the time representation in multi-period networks, humans at least in most societies, have a continuous and discrete versions representation of time such as seconds, minutes, hours, days, etc., which are using currently. A continuous illustration of time in operations research is based on the time definition of the human society (given the part of the world where the system under study performs). This variant can be converted to the discrete time points e.g., date, hour, minute and so on for a transferring a shipper-demand request using carrier-capacity offers in the proposed problem setting [51-52]. We instead adopt the classical discreet time representation, which is more amendable to optimization. In this case, the time is represented by a sequence ($T= \{1,2, \ldots, T\}$) [53-54]. Then, given a time instance $t \in T$, the associated (time) period t represents the duration from instant t to its successor instant $t + 1$. Periods are equal of duration in the considered time-space network. The decisions for the scheduling of services and routing of shipper-demand requests made at time t, and associated to period t, are generally implemented, that is, they are not to be changed at the mentioned time period t and are transmitted to the appropriate stakeholders and departments of IDSP for execution. The following periods from t+1 to period T belong to the look ahead component as the planning horizon. Most decisions of these periods are temporary in nature and rather serve to evaluate within the optimization model in a mathematical way.

Remark that we should estimate all the parameters of our optimization model in the tactical planning level. For example, we should estimate the volume, total revenue, opportunity cost and time window for both contract-based and non-contract requests which should be transferred from the origin terminal to the destination terminal in a single-segment corridor network.

In this study, a tactical level planning for the studied M1M system under study is to design the demand side (shippers) and supply side (carriers) with an objective function to maximize the profitability of the IDSP. For the demand side, the IDSP selects the requests even they are contract-based or non-contract ones to maximize the total profit with regards to the capacity and time limitations. For unaccepted contract-based requests, there is a big penalty cost. However, for

unaccepted non-contract requests, we may have an opportunity loss cost. The goal is to assign the accepted requests to the selected transportation services with regards to the departure and arrival times of services and their capacity, duration, and other properties. Based on these assignments, we design the itineraries of each accepted request considering their net profit, time window, volume, origin, and destination which are available in all periods for the IDSP.

To sum up, since all elements in the problem under consideration are time-related features, the proposed problem is time-dependent. In this case, the time-space networks are tools and instruments which can be used to model our proposed network. Hence, this study introduces a scheduled service network design model for making the tactical planning level of M1M system through time-space network where the decisions are made at different time periods.

CHAPTER 3 LITERATURE REVIEW

In this chapter, we focus on the relevant and recent studies for the M1M system. This chapter focuses on deterministic optimization models for service network design and multi-commodity network design problems. To optimize the plans for M1M systems, algorithms based on exact, heuristic, and metaheuristic solution approaches have been proposed.

Here, we first consider different metaheuristic algorithms existing in the literature and review exact methods where most of them were applied to multi-commodity network design problems (Section 3.1). Next, different contributions to the Service Network Design (SND) models and algorithms have been studied (Section 3.2). Finally, the main differences between our thesis and the existing literature review and the research gaps are identified (Section 3.3).

3.1 Multi-commodity network design studies

The multi-commodity network design problem is an extension to the fixed-charge network design problems where much simpler fixed-charge networks have been around since the early 1960's [7]. Most metaheuristic and exact algorithms have been applied to solve multi-commodity network design problem and its variations [53-56]. The literature contains several problems which are very close to the multi-commodity network design problem including but not limited to the single-commodity piecewise linear network design problem [57], multi-commodity piecewise convex network design problem [58], unsplittable multi-commodity piecewise linear network design problem [59], multi-commodity piecewise linear network design problem with integer flows [48]. A comprehensive review paper in 2000, Crainic [7] defined the service network design as a complicated case of multi-commodity network design for the selection and scheduling of services, specification of terminal operations, and routing of freight and reviewed service network design models and suggested innovative solutions for the network design. This review also confirms that heuristic and metaheuristic algorithms were at that time the best approaches to tackle large network design instances. Among these, Ghamlouche et al. [43] proposed an efficient tabu search algorithm that used specialized cycle-based neighborhoods. These cycle-based operators were also applied in a path-relinking algorithm in another study (Ghamlouche et al. [44]). Furthermore, a multi-level cooperative framework was also proposed for the multi-commodity network design problem by Crainic et al. [45]. Scatter search was first introduced by Crainic and Gendreau [46] in this research area. In another study, Pedersen et al. [54] defined a new optimization model for the service

network design problem considering asset positioning and utilization through constraints on asset availability at terminals. Their model was a generalization of the multi-commodity network design problem and both arc- and cycle-based formulations were presented. For solving it, an efficient metaheuristic based on tabu search was developed. Paraskevopoulos et al. [47] proposed an innovative evolutionary algorithm for solving a generic multi-commodity network design problem. Their metaheuristic uses a scatter search in its main loop as well as a local search algorithm and cycle-based neighborhoods to generate more high-quality solutions compared to existing methods. In addition to metaheuristics, several exact and heuristic methods were developed for solving service network design or multi-commodity network design problems. Gendron et al. [48] combined an iterative linear programming method and slope scaling heuristics for solving a multi-commodity capacitated network design problem. Chouman et al. [49] revised iterative branch-and-cut (B&C) based on local search strategies for solving the multi-commodity fixed charge network design problem. In another paper, Kazemzadeh et al. [52] suggested a node-based Lagrangian relaxation-based algorithm for solving the multi-commodity fixed-charge network design problem. Last but not least in this group of studies, Agarwal et al. [53] proposed valid inequalities to reformulate the base model for fixed-charge network design problems efficiently.

3.2 Service network design studies

Solving SND models are computationally challenging as they are more complex than the traditional multi-commodity network design problems [53-54]. One of the computational challenges associated with solving instances of SND models inspired by real-world operations is that the time-space networks on which these instances are growing very large increasingly [55-56]. As a result, the numbers of variables that model supply side and demand side simultaneously through this network in those instances are very large as well, leaving mathematical programs that are too large to be solved in a reasonable time [57-59]. The network reduction techniques proposed in Kim et al. [60] in 1999 applied the specifics of that logistics context in an attempt to mitigate this issue.

Later in 2002, Armacost et al. [61] studied another SND using an integer programming model where their initial formulation was not able to model package flows directly. In this regard, they extended their model based solely on design variables representing aircraft routes, while supporting sufficient capacity to transport all package demands, even though those demands are not explicitly

modeled. They called this specific design of variables as a composite variable, encoding the selection of multiple aircraft routes.

In 2009, Jarrah et al. [62] formulated the SND in the context of the less-than-truckload freight transportation industry. They considered the single path per shipment policy desired by carriers and developed a new mixed integer programming model. Specifically, since the paths, for shipments delivered to the same terminal, must induce a directed in-tree rooted at that terminal, the problem can be formulated with variables that represent flows on such trees. Their proposed solution approach generates destination in-trees in a column generation-fashion in the context of a heuristic scheme. In 2013, for a similar study, Erera et al. [63] proposed a new SND in the context of a matheuristic scheme which at each iteration chooses a destination terminal and then solved an integer program to route freight designed for that terminal, holding fixed the routes for freight destined for other terminals.

Service network design models can be applied to a multi-modal transportation system where different transportation modes, like trains, trucks, ships, and airplanes can be used for transferring the shipments. They can provide different alternatives for the capacity and costs of transportation and different timetables for the departure and arrival times of services. SteadieSeifi et al. [19] in 2014 reviewed all the relevant articles for multi-modal transportation for all the planning levels from strategic, tactical, and operational decision-making levels.

Another computational challenge often encountered when solving either SNDs or Scheduled SNDs (SSNDs) inspired by real-world operations, is that the number of shipments to be transported can be very large. This in turn can yield a large number of shipment flow variables and a mathematical program that is too large to be solved in reasonable run-times. One solution approach is a Benders decomposition-based method, wherein design decisions are made by a master problem based on estimates of the shipment's routing costs [64]. These estimates are reflected in a constraint set in the master problem that is iteratively added to as new designs are discovered. The downside of this type of approach is that these estimates are typically very poor in the early stages of the algorithm. That is why heuristics and metaheuristics are still the best alternatives to address this challenge of solving SNDs or SSNDs.

With the development of a local search-based metaheuristic algorithm, van Riessen et al. [24] considered synchro-modality in the area of service network design where a product development based on criteria of price and lead time, is contributed and the transportation services are considered

to transfer these new products. They applied their model to European Gateway Services where the objective was to maximize the revenue from transferring products from their origin to their destination. They defined a local search-based metaheuristic algorithm to final the global solution iteratively in the search space. To make the service network design more complex, Christiansen et al. [16] defined a new version of service network design as the liner shipping network design including a set of demands which have a specific origin terminal, destination terminal, and time limitations and a set of weekly services as well as a set of vessels with variable capacity. Their model defines the itinerary of each demand while maximizing the revenue of transported demands. Their solution algorithm was based on a two-stage heuristic to first design the suitable services and then select a subset of services to satisfy the demands.

The design of a single-segment corridor network is a special case of SND, which was first proposed by Crainic et al. [8] in 2021 in the context of planning at the operational level. In this paper, the authors consider a multi-period bin packing problem for the consolidation of shipments that must move from their origin terminal to their destination terminal. Constructive heuristic algorithms were applied for solving the resulting model. As far as we know, there is no other study that has contributed to the single-segment corridor network design problem similar to this study for tactical level planning.

More recently in 2022, Belgian et al. [25] integrated the revenue management and service network design model for the tactical level planning where intermodal consolidation-based freight transportation carriers have been considered in their optimization model. The objective function was the maximization of the net profit of carriers when there are a large number of customers distributed in different zones. They analyzed the distribution of demand nodes, network topology, and the quality of services in their results to show the impact of this integration on the optimality of solutions and values of decision variables.

Finally, Taherkhani et al. [10] in 2022 studied planning at the tactical level of a multi-stakeholder system and proposed an SSND model over a time-space network. Their model is the maximization of profit, accounting for revenues, transportation costs, warehousing costs, penalties, etc. One result of this paper was the implementation of the tactical planning level for an M1M system while integrating the demand and supply sides using an IDSP. They analyzed their SSND on a hyper corridor network where many terminals and zones existed in the time-space network and the main difficulty was to schedule services to satisfy the shipper-demand requests as much as possible.

3.3 Research gaps and contributions

To be more precise, the optimization problem in this thesis is a variation of the base model proposed in Taherkhani et al. [10]. First, their optimization model was based on revenue management to maximize the profits from accepted shipper-demand requests. In our optimization model, we have considered the same idea to maximize the profit minus the total cost for both sides of carriers and shippers within a single-segment corridor network including one pair of origin-destination terminals. Most notably, the proposed model considers the decision variable of extra capacity as ad-hoc services for unaccepted shipper-demand requests which are revenue as the opportunity loss cost in the proposed model. The original model did not contribute to the opportunity loss cost concept.

From the side of carriers in [10], the authors assumed that the transportation services could be bundled or individual. Another difference between our optimization model with their model is to consider only individual transportation services. In addition, the mentioned paper also considered the possibility of warehousing space to store the products temporarily for accepted shipper-demand requests. In this thesis, we have also the warehousing capacity in a different way where there is no decision variable to establish the warehouse centers. In this thesis, the terminals also play the role of warehouse centers. In another difference of our thesis refers to the services; while Takerhani et al. [10] consider both single-leg and multi-leg services because they study more complex network structures, our work focuses on single-leg services.

Regarding shipper-demand requests, the main difference is that in the model presented by Taherkhani et al. [10], it was assumed that all contract-based requests must be accepted. However, in this research, the proposed model can accept or reject both types of requests. In this regard, there is an opportunity cost for unaccepted non-contract requests and there is a big penalty cost for unaccepted contract-based requests. At last, but not least, the original model recognizes the possibility of split and unsplit shipment-flow versions of the problem, while the problem under study is not able to split them. It means that they are picked up from their origin and delivered to their destination as a whole.

Although many heuristics and metaheuristics have been applied to the service network design and multi-commodity network design problem [43-49], ALNS has not been applied yet. In a recent review paper, Mara et al. [55] in 2022 studied different applications of ALNS especially for routing

optimization. 252 articles from 2006 to 2021 were reviewed, but none of these papers considered the service network design.

To fill this research gap, finally, this study proposes an innovative solution method based on the ALNS where eight removal and four insertion operators are developed to improve them by a local search algorithm and SA for escaping from local solutions. As far as we know, there are no removal and insertion operators in the area of SND and our suggested operators are novel and can be improved by future studies in this field.

CHAPTER 4 MODELLING

In this chapter, the main scope is to model the proposed single-segment corridor network design problem in an M1M system under the tactical planning level. In this regard, a comprehensive problem definition with regards to key assumptions, is defined. Notations are then introduced to define sets, parameters, and decision variables. Finally, the main objective of this chapter is to offer an optimization model while formulating the proposed single-segment corridor network using the concept of extra capacity or ad-hoc arcs for all alternative requests through the network.

4.1 Problem definition

The proposed network optimization model is defined for a Scheduled Service Network Design (SSND) on a single-segment network to make tactical planning of M1M system to be performed over planning horizon where the possibility of extra capacity or ad-hoc services is contributed. The proposed network includes two terminals which are the origin and the destination where there exists a segment to connect them directly. Figure 4.1 defines a graphical example of a single-segment corridor network where the demands are aggregated at the origin terminal and after delivering the shipments to the destination terminal, they are distributed to their final destination.
This study formulates the single-segment corridor network to address the tactical planning of M1M system where it has one supply side (carriers), one demand side (shippers) and one IDSP to make the tactical decisions and integrate both supply and demand sides. There are several shippers on the demand side of the M1M system where they have a set of shipments that should be transferred from one origin to one destination. Transportation activities using services offered by the supply side take place between the origin and destination terminals. It should be noted that the pickup and delivery activities within these terminals are not explicitly considered in the problem definition. To manage both sides of the M1M system and implement a tactical planning level, the IDSP is developed to plan and optimize operations, improve profitably and simultaneously satisfy the needs of both categories of these stakeholders. This tactical planning is performed for a certain schedule length (e.g., one week), dividing into time periods of equal length (e.g., one day). This schedule length is then repeated over a certain medium-term planning horizon. Given the set of all shipper-demand requests (i.e., the demand) and carriers (i.e., the supply), the objective is to

consider both accepted and non-accepted requests, the selection of scheduled services, and the assignment of shipments to these services while finding the maximum overall profit entire schedule length. The main novelty of the proposed model is to consider the opportunity loss cost which is another revenue for unaccepted requests.

To have more details for the attributes of demand side of M1M system, there are two different types of shipper-demand requests in this problem, i.e., contract-based, and non-contract requests. Each request is characterized by many attributes including the volume, the revenue, the time window involving of release time for the pickup and due date for the preferred delivery time, the origin, and the destination where shipments must be unsplit. The goal is to choose both types of shipper-demand requests by our optimization model to make more profit for IDSP and satisfy them with regards to their time window. In this regard, the proposed model can accept or reject both types of requests where there is a big penalty cost for unaccepted contract-based requests and there is an opportunity cost for unaccepted requests.

As such, for the supply side, the goal is to select the transportation services to define the demand itineraries. Services are offered by carriers to satisfy the shipper-demand requests based on the available capacity of these selected services and their timetable in the most cost-efficient way. Every service is available at a certain point at the original terminal where the IDSP can use such services for a certain period time to travel the requests. As services have different attributes with each other including capacity, fixed and transportation costs, travel time, departure time, arrival time through origin and destination, i.e., there is only one origin-destination pair in the single-segment corridor network.

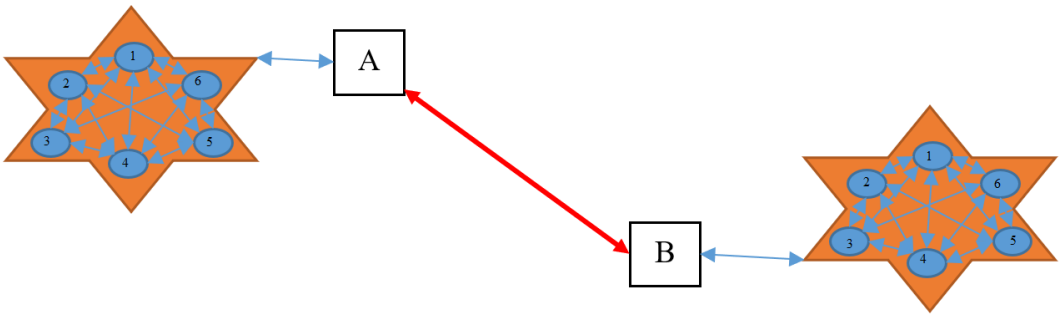


Figure 4-1: Single-segment corridor network

Based on the above problem definition, the proposed model covers the following assumptions:

- All transportation services offered by carriers are single-leg also called a single-segment corridor composed of a single long-haul segment and feeder segments in which each of the feeders are linked to both origin and destination terminals and there is one or more zones around each node where the shipments should be merged or distributed. That is to say that there is only one origin and one destination in the SSND.
- Carriers can provide transportation services to the IDSP only individually to match the demand side with the supply side for moving the shipper-demand requests.
- There is an opportunity loss cost as revenue for unaccepted shipper-demand requests.
- Carriers offer only transportation services with capacity limitations.
- There is the warehousing capacity for the origin and destination terminals.
- All shipments in this single-segment corridor network are unsplit.

4.2 Mathematical modelling

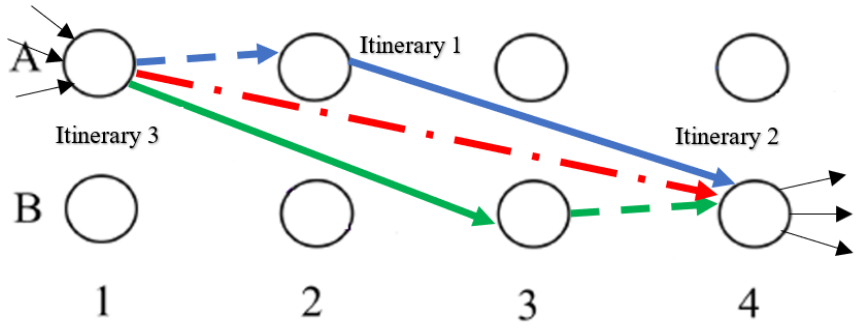
The proposed network is modelled by a set of nodes as terminals and a set of arcs which have time attributes as time-stamped arcs to transfer the shipments between these nodes. In this network, we have execution arcs to move the shipments and ad-hoc arcs where they have the extra capacity to artificially transfer unaccepted shipments. These ad-hoc arcs directly connect a shipment from its origin terminal to its destination. There is a very big penalty for contract-based requests while the opportunity loss cost is considered for the non-contract requests.

The capacity limitation is the main constraint of the proposed optimization model to make this SSND to be NP-hard [8] where there are three capacity limitations for the proposed model. First of all, as mentioned earlier, the transportation services have a limitation of capacity. Second, the stored shipments at the original terminal are limited by the warehousing capacity. Third, the destination terminal is limited by the capacity which is fixed for all the time periods. There is also a variable cost of holding for these shipments which are stored at the origin and destination terminals. Other main costs of the proposed model include two different fixed and variable costs relying on the either using services or not using them.

The total cost is the sum of transportation cost, i.e., fixed, and variable costs of services, holding cost of shipments and penalty cost if we have early and late pick-up and delivery of shipments the

request in both sides at its origin and destination. In addition to these costs, the proposed model maximizes the total revenue of accepted requests and the opportunity cost of non-accepted requests.

To illustrate how to model our single-segment corridor network graphically, see Figure 4.2 where there are two terminals the origin and the destination as well as a time-space network including four periods. The first and third itineraries are execution arcs in the time-space network. The second itinerary is an ad-hoc arc to define the opportunity loss cost for this unaccepted request instead of its net profit. As an example, we have one non-contract request and there are three alternatives as itineraries, where there are two potential transportation services as the execution arcs including green and blue services to move a shipment from the origin to the destination in a time-space network where T=4. The blue dash-line in itinerary 1, holds the shipment from t=1 to t=2 at the warehousing space of the origin terminal and then, transfer it to terminal B in time t=4. The green services known as itinerary 3, transfer the shipments from t=1 to t=3 and hold them from t=3 to t=4 at the warehousing space of the destination terminal. The last alternative is the use of ad-hoc arc (here, the red service) to transfer the shipments directly from the origin to the destination. This non-contract shipper request is not satisfied by this service.



$$Itinerarry\ 1 = \begin{Bmatrix} A^E = (A_2 \rightarrow B_4) \\ A^D = \{\emptyset\} \end{Bmatrix} \quad Itinerary\ 2 = \begin{Bmatrix} A^E = \{\emptyset\} \\ A^D = (A_1 \rightarrow B_4) \end{Bmatrix} \quad Itinerary\ 3 = \begin{Bmatrix} A^E = (A_1 \rightarrow B_3) \\ A^D = \{\emptyset\} \end{Bmatrix}$$

Figure 4-2: A graphical solution for the single-segment corridor network, G in which T=6

4.2.1 Notations

To establish an integer programming model for the proposed single-segment corridor network problem, the following notations should be used:

Sets:

$\mathcal{K}^C$	Set of contract-based shipper-demand requests where $k \in \mathcal{K}^C$,
$\mathcal{K}^{NC}$	Set of non-contract shipper-demand requests where $k \in \mathcal{K}^{NC}$,
$\mathcal{K}$	Set of all shipper-demand requests where $\mathcal{K} = \mathcal{K}^C \cup \mathcal{K}^{NC}$,
T	Set of time periods where $t \in T$,
$\mathcal{A}^E$	Set of execution arcs or transportation services, where $a \in \mathcal{A}^E$,
$\mathcal{A}^D$	Set of ad-hoc arcs or the extra capacity, where $a \in \mathcal{A}^D$,
$\mathcal{A}$	Set of all arcs in the time-space network, where $\mathcal{A} = \mathcal{A}^D \cup \mathcal{A}^E$,

Parameters:

$o_{(k)}$	Origin terminal for the shipment *k* represented by $n \in N$,
$d_{(k)}$	Destination terminal for the shipment *k* represented by $n \in N$,
ρ_k	Total revenue of a shipper-demand request *k*,
w_k	Volume of a shipper-demand request *k*,
$(\alpha_{(k)}, \beta_{(k)})$	Desired time period for picking up and dropping off a request *k* from the origin $o_{(k)}$ and at the destination $d_{(k)}$,
$\alpha_{(a)}$	Departure time of a transportation service *a* from the origin terminal,
$\beta_{(a)}$	Arrival time of a transportation service *a* to the destination terminal,
f_a	Fixed cost of selecting a transportation service *a*,
c_a^k	Total variable cost of a transportation service *a* for a shipper-demand request *k*,
$\bar{c}^k_{(o_{(k)},t)}$	Total variable cost of holding a shipment *k* at the origin terminal $o_{(k)}$ at time *t*,
$\bar{c}^k_{(d_{(k)},t)}$	Total variable cost of holding a shipment *k* at the destination terminal $d_{(k)}$ at time *t*,
$u^{MH}_{(o_{(k)},t)}$	Capacity of the origin terminal $o_{(k)}$ in time *t*,
$u^{MH}_{(d_{(k)},t)}$	Capacity of the destination terminal $d_{(k)}$ in time *t*,
oc_a^k	Opportunity loss cost for an unaccepted request *k* using an ad-hoc arc $a \in A^D$,
$\psi^k_{(o_{(k)},t)}$	Penalty cost for an earliness or lateness pickup of shipment *k* from the origin terminal $o_{(k)}$ at time t,
$\psi^k_{(d_{(k)},t)}$	Penalty cost for an earliness or lateness delivery of shipment *k* from the origin terminal $d_{(k)}$ at time t,
u_a	Capacity of a transportation service $a \in A^E$,

Decision variables:

x_a^k	1, if a shipper-demand request $k \in K$ is travelling on arc $a \in \mathcal{A}^E$, 0, otherwise.
y_a	1, if a transportation service $a \in \mathcal{A}^E$ is selected. 0, otherwise.
s_a^k	1, if an unaccepted request $k \in K$ is assigned to an ad-hoc arc $a \in \mathcal{A}^D$, 0, otherwise.
$r^k_{(o_{(k)},t)}$	1, if a shipper-demand request $k \in K$ is kept from its origin $o_{(k)}$ at time $t \in T$, 0, otherwise.

4.2.2 Formulation

The proposed optimization model for a single-segment corridor network design problem considering the possibility of extra capacity is developed as follows:

$$\max \sum_{k\in\mathcal{K}} \rho_k \sum_{a\in\mathcal{A}^E} x_a^k + \sum_{k\in\mathcal{K}} \sum_{a\in\mathcal{A}^D} oc_a^k s_a^k - \left(\sum_{a\in\mathcal{A}^E} f_a y_a + \sum_{k\in\mathcal{K}} \sum_{a\in\mathcal{A}^E} c_a^k . x_a^k + \sum_{k\in\mathcal{K}} \sum_{t\in\mathcal{T}} \left[\bar{c}^k_{(o(k),t)} r^k_{(o(k),t)} + \bar{c}^k_{(d(k),t)} r^k_{(d(k),t)} \right] + \sum_{k\in\mathcal{K}} \sum_{t\in\mathcal{T}} \left[\psi^k_{(o(k),t)} \sum_{a\in\mathcal{A}^E} \alpha_{(a)} x_a^k + \psi^k_{(d(k),t)} \sum_{a\in\mathcal{A}^E} \beta_{(a)} x_a^k \right] \right) \quad (4\text{-}1)$$

s.t

$$\sum_{a\in\mathcal{A}^E} x_a^k + s_a^k = 1 \qquad k\in\mathcal{K}, a \in \mathcal{A}^D \quad (4\text{-}2)$$

$$\sum_{k\in K} w_k x_a^k \le u_a y_a \qquad a \in \mathcal{A}^E \quad (4\text{-}3)$$

$$\sum_{k\in K} w_k r^k_{(o(k),t)} \le u^{MH}_{(o(k),t)} \qquad o_{(k)} \in N, t \in T \quad (4\text{-}4)$$

$$\sum_{k\in K} w_k r^k_{(d(k),t)} \le u^{MH}_{(d(k),t)} \qquad d_{(k)} \in N, t \in T \quad (4\text{-}5)$$

$$|\mathcal{T}| r^k_{(o(k),t)} \ge \sum_{a\in\mathcal{A}^E} x_a^k (\alpha_{(a)} - t) \qquad k\in\mathcal{K}, t\in\mathcal{T}, t \ge \alpha_{(k)} \quad (4\text{-}6)$$

$$|\mathcal{T}| r^k_{(d(k),t)} \ge \sum_{a\in\mathcal{A}^E} x_a^k (t - \beta_{(a)}) \qquad k\in\mathcal{K}, t\in\mathcal{T}, t \le \beta_{(k)} \quad (4\text{-}7)$$

$$x_a^k = \{0,1\} \qquad k\in\mathcal{K}, a\in\mathcal{A}^E \quad (4\text{-}8)$$

$$y_a = \{0,1\} \qquad a\in\mathcal{A}^E \quad (4\text{-}9)$$

$$s_a^k = \{0,1\} \qquad k\in\mathcal{K}, a\in\mathcal{A}^D \quad (4\text{-}10)$$

$$r^k_{(o(k),t)} = \{0,1\} \qquad o_{(k)} \in N, t\in T, k\in\mathcal{K} \quad (4\text{-}11)$$

$$r^k_{(d(k),t)} = \{0,1\} \qquad d_{(k)} \in N, t\in T, k\in\mathcal{K} \quad (4\text{-}12)$$

The objective function is given in Eq. (4-1), while constraints are given in Eqs. (4-2) to (4-12). The objective function consists of six expressions. The first is the net profit for each shipment based on the total volume. The second term is the total cost of the opportunity loss for all shipper-demand requests which are not accepted. The third term is a fixed fee for choosing transportation services. The fourth term is the total shipping cost. The fifth term is the total holding cost at origin and/or destination. The last one is the cost of penalty for pickup and delivery.

The proposed model is limited by a set of constraints where we start with the constraint set (4-2) to confirm that each shipment must be shipped by an ad-hoc service if this request is not accepted. Constraints (4-3) to (4-5) confirm that there is a capacity constraint for transportation services, the terminal of origin, and the terminal of destination. The constraint set (4-3) means that the total flow of shipments cannot exceed the capacity of the selected service. Thus, constraints (4-4) and (4-5) mean that the total number of shipments that can be kept at a terminal cannot exceed its capacity. Constraints (4-6) and (4-7) confirm that if the time of departure and arrival time of the transport services are not favorable by the time window, the consignment must be stored at the terminals of origin and destination, respectively. The decision variables are defined in relationships (4-8) to (4-12).

CHAPTER 5 SOLUTION ALGORITHM

This thesis proposes an Adaptive Large Neighborhood Search (ALNS) using a local search and Simulated Annealing (SA) as our problem-solving method. Originally, Large Neighborhood Search (LNS) was first introduced by Shaw [1] inspired by the principle of removal and insertion [2] in which the main idea of LNS is to destroy the current solution and then reconstruct it exploratory. Based on these heuristics, the algorithm can explore and exploit a new solution. Exploring the optimal solutions found by the algorithm depends directly on the number of removal-insertion operators. Furthermore, exploiting new solutions depends on the quality of operators and local search coherence. The main difference between ALNS and the original LNS is having an adaptive strategy to consider and update the weight of each operator. This study focuses on exploring new solutions and then exploiting optimal ones using removal-insertion operators, local search-based strategies such as SA, and evolutionary mechanisms such as roulette wheel selection. To reconstruct this solution, one of the advantages of our heuristic methods is that they have less computational time while optimizing the improved ALNS which is combined with SA and a local search heuristic.

This hybrid ALSN-SA-based algorithm considering local search uses a selection method to choose one pair of removal-insertion operators based on their weights and then, probability is computed by the normalized weights. Finally, an evolutionary mechanism, such as the selection of a roulette wheel [3] or a tournament operator [4], can be used to select each pair of removal and construction operators in each iteration. It should be noted that in the first iteration, the weight of all operators and their probability is the same. Based on the recorded performance of the operators, after each iteration, their weights are updated and adjusted repeatedly, and their probabilities are calculated. For example, after several repetitions, consider selecting a roulette wheel in which there are two operators with probabilities of 0.7 and 0.3, respectively. Then, the cumulative weight for each operator is calculated as [0, 0.7] and [0.7, 1]. The roulette wheel selects a random number between zero and one (e.g., 0.46). In this case, the first operator is assumed to be 0.46 in the range [0, 0.7]. The main contributions of our solution are the development of various creative and efficient removal and insertion operators. The most important part of removing is our eight operators which eliminating some useless services or requests while deleting even one solution is considered. Although alternatives to random selection for available services exist, costly and underused items

are involved with related requests. Useless requests focus on low-profit request-based removal, low-volume request-based removal, cluster- based removal operators, and random-based requests. A hybrid destruction method relates to both inefficient services and less useful requests as a removal method. This destruction operator takes advantage of two efficient operators for selecting low-utility service and eliminating a bunch of non-contract requests from the shipper. We also remove the services randomly, low-utilization and high-cost as the criteria for removing them. After the removal phase, there are four repair operators, one of them will offer a new service with lower cost if the current open services do not have enough capacity to add a new request. The rest focuses on submitting a request among unaccepted them based on the criteria of maximum net profit or maximum volume of requests while meeting capacity constraints, as well as a random request-service operator. The final step is to have a local search operator for this new solution found by the pair of the insertion-removal operators, at each iteration, we have sub-iteration for the local search operator. In this regard, small changes to remove one or two requests and add them for escaping from local optimum, are made. After this phase, the algorithm checks the termination criterion.

In the following, we first explain the solution representation and search space for our metaheuristic algorithm (Section 5.1). The full algorithmic framework for the proposed hybrid metaheuristic is explained later (Section 5.2). A constructive heuristic algorithm for creating an initial solution is developed (Section 5.3). Next, the neighborhood procedures including our removal (Section 5.4) and insertion (Section 5.5) operators, are defined, and illustrated mathematically. Finally, the local search operator as a sub-loop of our metaheuristic is presented (Section 5.6).

5.1 Solution representation and search space

In the proposed metaheuristic algorithm, the search space includes all the feasible solutions $s \in S$ where the global optimum (s^*) is expected to be found. In the developed metaheuristic, the search space covers different alternatives of assignment variables ($i.e., x_a^k$) to show that each accepted request has been assigned to which transportation service. The proposed algorithm explores and exploits new neighborhoods from the search space defined by the main design variable ($i.e., x_a^k$). Neighborhoods are found by removal and insertion operators in the search space at each iteration ($l \in \{1, 2, \dots, MaxIt\}$). The search space in the developed metaheuristic

should be designed by execution ($a \in A^E$) and ad-hoc arcs ($a \in A^D$) to pick the shipments up from their origin and deliver them to their destination while satisfying the desired time window and considering the capacity of services and terminals. In addition to x_a^k, other variables are defined by this variable as will be explained in the following.

To present the solution while having less computational time, we add a new auxiliary variable depending on x_a^k. In this regard, the selection of requests is defined to make our solution reasonable as follows:

$$\bar{z}_k = \begin{Bmatrix} 1 & If \sum_{a \in \mathcal{A}^E} x_a^k = 1, \forall k \in K \\ 0 & Otherwise \end{Bmatrix} \quad (5\text{-}1)$$

where z_k defines the status of accepted or rejected requests. In addition, the search space does not include $y_a = \bar{y}_a$ as there is an implicit relationship between y_a and x_a^k for any feasible solution. Note that $\bar{y}_a$ shows the value for the decision variable y_a. Therefore, the search space focuses on x_a^k and we can get the value of $y_a = \bar{y}_a$ using the following relation:

$$\bar{y}_a = \begin{Bmatrix} 1 & if \sum_{k \in K} x_a^k \geq 1, \ \forall a \in A^E \\ 0 & Otherwise \end{Bmatrix} \quad (5\text{-}2)$$

Based on the triple vectors of x_a^k, y_a and z_k, the following conditions should be met to address a feasible solution:

$$z_k = \{0,1\} \qquad \forall k \in K \quad (5\text{-}3)$$

$$x_a^k \leq z_k \qquad \forall a \in A, k \in K \quad (5\text{-}4)$$

$$x_a^k \leq y_a \qquad \forall a \in A, k \in K \quad (5\text{-}5)$$

$[\{x_a^k | x_a^k = 1, \forall a \in A, k \in K\}$ defines a valid itinerary including one or more services to pick up the shipment (w_k) from $o_{(k)}$ and deliver to $d_{(k)}$ in the time-space network] (5-6)

To show an example of the values z_k, y_a and x_a^k, Fig. 5.1(a) shows a feasible solution for $z_k = \bar{z}_k$ where there are five contract-based and non-contract requests in this example. A feasible solution for $y_a = \bar{y}_a$ is shown in Fig. 5.1(b) where there are six transportation services and four of them are used. Finally, a feasible solution for $x_a^k = \bar{x}_a^k$ is displayed in Fig. 5.1(c).

	K^C(Set of contract − based requests)					K^{NC}(Set of non − contract requests)				
	z_1	z_2	z_3	z_4	z_5	z_6	z_7	z_8	z_9	z_{10}
z_k ➔	1	1	1	1	1	0	1	1	0	1

(a) The selection of requests

	A^E(Set of execution arcs or transportation services)					
	y_1	y_2	y_3	y_4	y_5	y_6
y_a ➔	1	0	1	1	0	1

(b) The selection of transportation services

	K^C			K^{NC}		
	z_1	…	z_5	z_6	…	z_{10}
y_1	1	…	0	0	…	0
x_a^k ➔ y_2	0	…	0	0	…	0
…	…	…	…	…	…	…
y_6	0	…	0	0	…	1

(c) The assignment of accepted requests to the transportation services

Figure 5-1: The search space of design vectors for a random feasible solution

In conclusion, our solution includes three main binary variables (z_k, y_a and x_a^k) and three dependent variables ($r^k_{(o_{(k)},t)}$ and $r^k_{(d_{(k)},t)}, s_a^k$), where these dependent variables for holding the shipments at the origin and destination terminals and ad-hoc arcs are computed by replacing the main variables after solving ($z_k = \bar{z}_k, y_a = \bar{y}_a, x_a^k = \bar{x}_a^k$) based on the constraints in the mathematical model. We define the extra capacity or ad-hoc arcs (A^D) for both contract-based (K^C) and non-contract (K^{NC}) shipper-demand requests. These arcs are made up $a \in A^D$ to create a direct connection for all pairs of origin-destination of requests. The search space for all possible solutions is to design x_a^k where a set of transportation services should be selected for each shipment or an ad-hoc arc (s_a^k) should be replaced to define the opportunity loss cost for unaccepted requests. The computation of a solution representation confirms its feasibility where the capacity constraints (4-3) to (4-5) have been met by the removal and insertion operators. In the proposed metaheuristic algorithm, a positive value of the objective function means that all contract-based requests have been accepted. However, a negative value means that one or more contract-based requests are rejected.

5.2 Full algorithmic framework

The hybrid ALNS-SA-based algorithm starts with an initial solution from our constructive heuristic algorithm which will be illustrated in Section 5.3 as the Step 0 where we call this initial solution as S_0 which is the current best solution (S_{best}). In the first iteration, the weights of all eight removal operators (rw_i^-) where $i \in I$ are the same and equal to one. As such, the weights of all four insertion operators (iw_j^+) where $j \in J$ are the same in the first iteration and equal to one. Next, the probability for the selection of each of these operators is computed as below:

$$P_i^{rw} = \frac{rw_i^-}{\sum_i rw_i^-} \qquad \forall i \in I \tag{5-7}$$

$$P_j^{iw} = \frac{iw_j^+}{\sum_j iw_j^+} \qquad \forall j \in J \tag{5-8}$$

Next, the roulette wheel selection is employed to select one of these removal and insertion operators. Based on the recorded performance of operators after each iteration, their weights are updated and adjusted iteratively, and their probabilities are computed as given in Eq. (5-7) and (5-8). Cumulative probability must be computed after updating the probability of operators. In this regard, to update the weights, we should give a score (Ω) to the operators as follows:

$$\Omega = \begin{cases} \varpi_1 & If\ the\ new\ solution\ (S_{new}) is\ the\ new\ gloabl\ best\ (S_{best}). \\ \varpi_2 & If\ the\ new\ solution\ is\ accepted. \\ \varpi_3 & If\ the\ new\ solution\ is\ rejected. \end{cases} \tag{5-9}$$

where ϖ_1, ϖ_2 and ϖ_3 are the parameters and we have the following relation: $\varpi_1 \geq \varpi_2 \geq \varpi_3$. For example, these parameters can be set as 0.5, 0.3, and 0.2 for ϖ_1, ϖ_2 and, ϖ_3 respectively. If $\lambda \in [0,1]$ is a decay parameter, the weights are updated as follows:

$$rw_i^- = \lambda \times rw_i^- + (1-\lambda)\Omega \qquad \forall i \in I \tag{5-10}$$

$$iw_j^+ = \lambda \times iw_j^+ + (1-\lambda)\Omega \qquad \forall j \in J \tag{5-11}$$

After doing both removal and insertion operators, the new solution is called as S_{new}. If this solution is better than the best solution ever found, it should be replaced. Otherwise, it may be accepted or rejected based on a probability calculated as follows:

$$p = e^{-\Delta/Tem} \qquad where\ \ \Delta = |f(S_{new}) - f(S_{best})| \tag{5-12}$$

In Eq. (5-12) parameter *Tem* is the current temperature and updated at each iteration as follows:

$$Tem = redu \times Tem \quad (5\text{-}13)$$

In Eq. (5-13) parameter $redu$ is the damping rate of the initial *Tem*. The above decision rule is taken by SA to give a chance to a solution if it is not better than the best solution ever found. Otherwise, the algorithm goes to the next phase where there are several sub-iterations using a local search operator to improve this new solution found by the algorithm. We want to have some minor changes under the current solution using a local search operator by removing one or two requests and adding them randomly. The hybrid ALNS-SA-based algorithm checks the termination criterion where the algorithm should satisfy the maximum number of iterations ($MaxIt$). Finally, a brief review of the proposed hybrid ALNS-SA-based is shown in the pseudo-code given in Fig. 5.2. The developed metaheuristic has the following steps generally:

Step 0: Set the weights of operators and create a feasible solution (S_o) which is the current best solution (S_{best}). This solution is found by the constructive heuristic algorithm as given in Section 5.3. This step is shown in lines 1 to 7 in the pseudo-code of our metaheuristic algorithm.

Step 1: Define the probabilities of each operator and select a pair of removal-insertion operators considering the roulette wheel selection. This step is done in lines 9 to 10 in the pseudo-code of our metaheuristic algorithm.

Step 2: Apply these operators and make a new solution (S_{new}). This step is done in lines 11 and 12 in the pseudo-code of our metaheuristic algorithm.

Step 3: Apply the decision rule to accept or reject S_{new} using the current temperature and damping ratio and update the current best solution ever found (S_{best}). This step refers to lines 13 to 20 in the pseudo-code of our metaheuristic algorithm.

Step 4: Determine the score of employed operators referring to line 22 in the pseudo-code of our metaheuristic algorithm.

Step 5: Apply the local search operator to improve the current best-known solution ever found (S_{best}). This step refers to lines 24 to 27 in the pseudo-code of our metaheuristic algorithm.

Step 6: Update the weight of operators where this step is done in line 29 in the pseudo-code of our metaheuristic algorithm.

Step 7: If the algorithm is terminated, output the current best-known solution ever found (S_{best}). Otherwise, go to **Step 1**. This step is written in lines 30 to 31 for the pseudo-code of our metaheuristic algorithm.

1: Set the maximum number of iterations (*MaxIt*) and sub-iterations (*SubIt*)
2: Run the constructive heuristic algorithm to define a feasible solution (S_o).
3: Define the initial weights $rw_i^- = 1$ where $i \in I$ and $iw_j^+ = 1$ where j ∈ J;
4: $S_{best} = S_0$;
5: It1=0;
6: T=*Tem*;
7: $\alpha = redu$;
8: **While** $It1 < MaxIt$
9: Run Eqs. (5-7) and (5-8) to compute P_i^{rw} and P_j^{iw};
10: Run the roulette wheel selection for removal (P_i^{rw}) and insertion (P_j^{iw}) operators;
11: Apply these selected removal and insertion operators.
12: Build a new optimal solution as S_{new}.
13: **If** $f(S_{new}) > f(S_{best})$
14: $S_{best} = S_{new}$;
15: **Elseif** $f(S_{new}) < f(S_{best})$
16: $\Delta = |f(S_{new}) - f(S_{best})|$
17: $p = e^{-\Delta/T}$
18: **If** $rand \leq p$
19: $S_{best} = S_{new}$;
20: **Endif**
21: **Endif**
22: Write the score of these selected removal and insertion operators as given in Eq. (5-9).
23: *It2*=0;
24: **While** $It2 < SubIt$
25: Apply the local search operator to improve the current best-known solution found (S_{best}).
26: $It2 = It2 + 1$;
27: **Endwhile**
28: $It1 = It1 + 1$;
29: Update the weights (rw_i^-; iw_j^+) using Eqs. (5-10) and (5-11).
30: **Endwhile**
31: Output the best-known solution ever found (S_{best}).

Figure 5-2: The pseudo-code for the proposed ALNS-SA-based metaheuristic algorithm

5.3 A constructive heuristic algorithm

One of the disadvantages of neighborhood-based metaheuristics such as ALNS and SA is that their performance is strongly related to the quality of initial solutions [6-7]. In this regard, this study proposes a constructive exploration of the proposed problem to create an initial solution which is the Step 0 for the main metaheuristic algorithm written in Section 5.2. To define a solution and design x_a^k, we first sort the volume of requests from largest to smallest. In this way, we sort the capacity of transportation services from the largest to the smallest. Generally, we not only sort the volume of requests but also the capacity of transportation services. In this order, we consider the shipper-demand requests, first the contract-based requests, and then the non-contract requests.

Since all requests and services are transferred from terminal A (origin) to terminal B (destination) in the single-segment corridor network, an innovative priority-based algorithm is proposed for assigning shipments to transportation services (x_a^k), and arcs ad-hoc (s_a^k). The proposed algorithm is generally a development of the exploratory algorithm for the bin-packing problem proposed by Crainic et al. [8]. The main principles of the proposed priority-based heuristic are working with the following criteria:

- Sort the volume of shipments and the available capacity of transportation services.
- Among the available services, choose a service that has the lowest time interval based on the comparison of timetable of services with the time window of shipments.

1: Generate a test problem and define its time-space network.
2: Define the order of shipper-demand requests from contract-based (K^C) to non-contract (K^{NC}) with the highest volume (w_k).
3: Call the redefined order of requests as $\widehat{K}$.
4: **For** each $k \in \widehat{K}$
5: **For** each $a \in A^E$
6: **While** $remaining\ capacity\ is\ enough$
7: Assign the request *k* to this transportation service.
8: Update ($z_k = \bar{z}_k, y_a = \bar{y}_a, x_a^k = \bar{x}_a^k$).
9: **Endwhile**
10: Compute the remaining capacity of this execution arc.
11: **Endfor**
12: **If** the request k ∈ K^{NC} is not assigned to any execution arc.
13: Assign it to an ad-hoc arc ($s_a^k = s_a^k$).
14: $\bar{z}_k$=1;
15: **Endif**
16: Compute the holding variables ($r^k_{(o_{(k)},t)}, r^k_{(d_{(k)},t)}$) to reduce the penalty cost ($\psi^k_{(o_{(k)},t)}, \psi^k_{(d_{(k)},t)}$) or link the pickup to the delivery.
17: **Endfor**
18: Compute the total profit for all accepted requests ($z_k = \bar{z}_k$).

Figure 5-3: The pseudo-code of the constructive heuristic algorithm

To explain the procedures of our constructive heuristic algorithm, Fig. 5.3 shows the pseudo-code for this constrictive heuristic. To illustrate how we apply this heuristic algorithm, an example is provided here to explain lines 6 to 15 where a priority-based heuristic algorithm from Crainic et al. [8] is applied. Note that there are six shipper-demand requests and the requests with the index of "1" to "4" are contract-based, i.e., $k = \{1, 2, 3, 4\} \in K^C$ and the requests with the index of "5" to "6" are the non-contract ones, i.e., $k = \{5, 6\} \in K^{NC}$. As such, there are only three transportation services with the index of "1" to "3", i.e., $a = \{1,2,3\} \in A^E$. These numbers show the index of the

requests and transportation services. As such, there is a priority based on the volume of requests and the capacity of services. The priority of contract-based requests is $\{1, 6, 5, 3\} \in K^C$. In addition, the priority of non-contract requests is $\{2, 4\} \in K^{NC}$. Lastly, the priority of transportation services is $\{1,3, 2\} \in A^E$. Finally, we assign requests to services based on their priority for our time window. In addition to the capacity of transportation services, the heuristic algorithm also focuses on the deviation of departure time from the desired pickup time and service arrival time from the delivery time of shipments. This criterion is called time interval and less value is preferred while the highest capacity is preferred in another criterion. If the capacity of the two transportation services is the same, the algorithm assigns a transportation service that has less time interval calculated in the following formula:

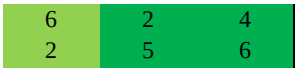
$$\hat{a} = \{\hat{a} \in A^E | argmin \left(\left| \alpha_{(a)} - \alpha_{(k)} \right| + \left| \beta_{(a)} - \beta_{(k)} \right| \right) \forall k \in K\} \quad (5\text{-}14)$$

Having more justifications and clarifications for the proposed priority-based heuristic algorithm focusing on available capacity and time interval, an illustrative example explained later, is detailed here to present how this allocation is performed, as shown in Figure 5.4. To solve this example, we have six stages to define a solution.

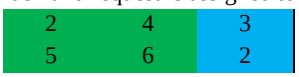
The stages for the assignment of shipper-demand requests to the transportation services are as follows:

Stage 1: The first shipper-demand request (*k*=1) with the priority "1" is assigned to the first transportation service ($a = 1$) and the remaining capacity is updated ($x_1^1 = 1$).

Stage 2: The fourth shipper-demand request ($k = 4$) with the priority "3" is assigned to the first transportation service ($a = 1$) because the volume of request (k=4) is smaller than remaining capacity of service (a=1). In this regard, $x_1^4 = 1$ and the remaining capacity is updated.



Stage 3: The third shipper-demand request ($k = 3$) with the priority "5" is assigned to the first transportation service ($a = 1$). In this regard, $x_1^3 = 1$ and the remaining capacity is updated. Since the remaining capacity is not enough to add another request, the next shipper-demand request is assigned to the next service.



Stage 4: The second shipper-demand request ($k = 2$) with the priority "6" is assigned to the third transportation service ($a = 3$) with the smallest time interval. In this regard, $x_3^2 = 1$ and the remaining capacity is updated.

Stage 5: The fifth shipper-demand request ($k = 5$) with the priority "2" is assigned to the third transportation service ($a = 3$) because the volume of request (k=5) is smaller than remaining capacity of service (a=3). In this regard, $x_3^5 = 1$ and the remaining capacity is updated. Since the remaining capacity is not enough to add another request, the next shipper-demand request is assigned to the next service.

Stage 6: The sixth shipper-demand request ($k = 6$) with the priority "4" is assigned to the second transportation service ($a = 2$) with the smallest time interval. In this regard, $x_2^6 = 1$ and the remaining capacity is updated.

Figure 5-4: Priority-based heuristic algorithm for the assignment of shipper-demand requests to the transportation services

Having an example displayed in Figure 5.4, the priority of contract-based requests was {1,6,5,3} for $k \in \{1,4,3,2\} \in K^C$ respectively and the priority of non-contract requests was {2,4} for $k \in \{5,6\} \in K^{NC}$, respectively. This new sort of shipper-demand request is defined, K'', as the redefined order. The priority-based heuristic works with a new set of assignment priorities. For example, if the set of shipper-demand requests is $K=K^C \cup K^{NC}$, the redefined order of requests is defined as K''. In another way, the algorithm works with a new sort of transportation services based on the largest capacity and smallest time interval as computed in Eq. (5-14). In the same way, the algorithm works with a new sort of transportation service. Accordingly, the priority of transportation services was $a \in \{1,3,2\} \in A^E$. All contract-based shipper-demand requests have been assigned to the available transportation services. If a non-contract shipper-demand request is

available and there was no transportation service with enough remaining capacity ($\bar{x}_a^k = 0$), this request should be assigned to an ad-hoc arc (s_a^k). The main criteria to terminate the algorithm are to have either no remaining capacity to add an unaccepted request to the available transportation services which were sorted in a priority-based concept, or all the requests have been accepted. If a shipper-demand request is assigned to the ad-hoc arc ($\bar{s}_a^k$=1), the opportunity loss cost (oc_a^k) is considered as the revenue and added to the objective function. Based on the constraint set (4-2), if a shipper-demand request is assigned to an ad-hoc arc ($\bar{s}_a^k = 1$), this request is not approved ($\bar{z}_k = 0$). For each unaccepted request, there is an ad-hoc arc using Eq. (5-15):

$$\bar{z}_k = 0, \quad if\ \bar{s}_a^k = 1, \qquad \forall k \in K, a \in A^D \tag{5-15}$$

After the assignment of execution arcs by x_a^k, while avoiding the case of early and late pickup and delivery for the accepted requests ($\bar{x}_a^k = 1$), the shipments should be kept at their origin terminal ($r^k_{(o_{(k)},t)}$) and their destination terminal ($r^k_{(d_{(k)},t)}$). In this regard, the open execution arcs are recognized by Eq. (5-16):

$$\bar{y}_a = \begin{Bmatrix} 1 & if \sum_{k \in K} x_a^k \geq 1,\ \forall a \in A^E \\ 0 & Otherwise \end{Bmatrix} \tag{5-16}$$

Based on the transferred shipper-demand requests, the initial solution of $z_k = \bar{z}_k$ is computed as follows:

$$\bar{z}_k = \begin{Bmatrix} 1 & if \sum_{a \in A^E} x_a^k = 1,\ \forall k \in K \\ 0 & Otherwise \end{Bmatrix} \tag{5-17}$$

Finally, we define the penalty cost for the pickup and delivery times if an accepted shipper-demand request ($\bar{z}_k = 1$) does not meet the desired time window. After the calculation of penalty costs ($\sum_{k \in \mathcal{K}} \sum_{t \in \mathcal{T}} \left[\psi^k_{(o_{(k)},t)} \sum_{a \in \mathcal{A}^E} \alpha_{(a)} x_a^k + \psi^k_{(d_{(k)},t)} \sum_{a \in \mathcal{A}^E} \beta_{(a)} x_a^k\right]$), and solving all decision variables, the objective function in Eq. (4-1), is calculated. Aforementioned procedures are done in lines 12 to 18 in the pseudo-code of our constructive heuristic algorithm shown in Figure 5.3.

5.4 Removal operators

Since the proposed metaheuristic has a two-phase framework to destroy a solution and then repair it heuristically, here, the removal operators are defined to remove either services or requests or

both in a systematical way. In this regard, a percentage of requests or services or both is considered to be removed randomly or based on some criteria like cost, utilization, efficiency and so on. This partial destruction may transform our current feasible solution to be an infeasible one.

To make these removal operators, we should define $l \in \{1, 2, \dots, MaxIt\}$ as the index of iterations by the algorithm where at the first iteration (*l=1*), the current solution is taken from the constructive heuristic algorithm ($S_{best} = S_0$). Since the independent variable x_a^k is the main part of the solution representation, the removal operators update this variable at the iteration *l*, as x_a^{kl}.

One important factor which has a high impact on the optimality of a solution is the cost of open services. Hence, removal operators may select a percentage of open services to remove them from the current solution. In this regard, there is a subset of open services (A^l) which can be defined as follows:

$$A^l = \{a \in A^E | \sum_{k \in K} x_a^{kl} \geq 1\} \tag{5-18}$$

The removal operators should deduct the set of A^l in each iteration *l* and reconstruct the set of A^{l+1} for the next iteration $l + 1$ to make the tentative solution $x_a^{k,l+1}$.

Another important factor for a removal operator is to select subset of requests which are accepted ($\widehat{K}$) at iteration *l*. Accordingly, we can define the variable of z_k^l as follows:

$$z_k^l = \begin{cases} 1 & if \sum_{a \in A^E} x_a^{kl} = 1, \quad \forall k \in K, l \in L \\ 0 & Otherwise \end{cases} \tag{5-19}$$

z_k^l shows that at the iteration *l*, which requests have been accepted and rejected. The proposed metaheuristic from iteration *l* to *l*+1 should decide on z_k^l to choose that which set of requests z_k^{l+1}should be selected in the next iteration.

Having a summary of the main elements of removal operators, we focus on two sub-sets of services and requests defined respectively in Eqs. (5-18) and (5-19) to remove some random elements of a solution. To destroy a solution, there is one parameter to show the percentage of elements which must be removed and can be set between zero and one. All in all, this study uses the following removal operators:

5.4.1 Random service-based removal

For the random service-based removal operator, a bunch of open services from the set of $\hat{y}_a^l$, would be randomly selected. In this regard, we select $\hat{y}_a^l$ meaning the arc is open at this iteration as given in the following equation:

$$\hat{y}_a^l = \begin{cases} 1 & if \sum_{k \in K} x_a^{kl} \geq 1,\ \forall a \in A^E, l \in L \\ 0 & Otherwise \end{cases} \quad (5\text{-}20)$$

The number of deleted services is a random number between one and the maximum number of open services. It should be noted that all requests related to this service are then deleted. Among them, contract-based requests should be assigned to other services while constructing a feasible solution.

5.4.2 High-cost service-based removal

The high-cost service removal operator focuses on removing some members from set A^l on the dependent design variable ($\hat{y}_a^l$). In this regard, the algorithm focuses on variables linking with arcs where each of them is corresponding to one or more shipper-demand requests ($\hat{y}_a^l$). We should compute the total cost of each open transportation service ($a \in A^E$) per unit of capacity as follows:

$$tc_a = (\sum_{k \in K} c_a^k \bar{x}_a^k + f_a \bar{y}_a)/u_a, \qquad \forall a \in A^E \quad (5\text{-}21)$$

A bunch of services a_l' which have the highest total cost, is selected as follows:

$$\hat{a}^l = \{a \in A^l | \operatorname{argmax}(tc_a)\ if \hat{y}_a^l = 1\} \quad (5\text{-}22)$$

The number of removed services from the set $\hat{a}^l$ is a random number between one and the maximum number of open services. This set shows a subset of services for which the value of their total cost is the highest. Based on this strategy, one or more arcs from the set of $\hat{a}^l$ is removed and then, if one of the shipper-demand requests is not satisfied by removing this service, an ad-hoc arc is added to the solution to escape from infeasibility ($s_a^k = 1$) and this arc defines the opportunity loss cost to the objective function (oc_a^k). It should be noted that if the set $\hat{y}_a^l$ from Eq. (5-18) has still one arc which is responsible for one of shipper-demand requests, the service-based removal is possible.

5.4.3 Low-utilization service-based removal

This removal operator identifies highly active services that may match more than one for transportation of shipments. Since the fixed cost of services is very high, it is efficient if each service handles more than one request. In this regard, contrary to efficient ones, we need to identify a transportation service that is useless. This means that it has the least usage based on the following equation:

$$a'_l = \{a \in A^l | \operatorname{argmin} \left((\sum_{k \in K} x_a^{kl}) / |K| \right) \} \quad (5\text{-}23)$$

A number of services from the set of a'_l is selected. The number of removed services from the set a'_l is a random number between one and the maximum number of open services. In this regard, the current solution is destroyed.

5.4.4 Low-profit request-based removal

Another destruction operator is the low-profit request-based removal. To destroy the current solution, we can remove a bunch of non-contracts which are accepted shipper-demand requests, abbreviated as $\widehat{K}$ which is a sub-set of $K = K^C \cup K^{NC}$. The algorithm removes some orders which has the lowest net profit (ρ_k) among the accepted non-contract requests. In this regard, at each iteration *l*, we can define the following sub-set:

$$\widehat{K}^l = \{k \in \widehat{K} | z_k^l = 1, k \in K\} \quad (5\text{-}24)$$

where $\widehat{K}^l$ shows the set of accepted requests at iteration *l*. Among them, a bunch of requests which have the lowest net profit, would be selected and them removed from the set of $\widehat{K}^l$In this regard, the following formula is defined:

$$k''' = \{k \in \widehat{K}^l | \operatorname{argmin}(\rho_k) \} \quad (5\text{-}25)$$

where k''' is sub-set of non-contract requests which must be removed from the current solution and this operator cannot be used if the set of $\widehat{K}^l$is empty.

5.4.5 Low-volume request-based removal

Another destruction operator is the low-volume request-based removal. This time, we should select a bunch of requests randomly which have the lowest volume. These requests would be selected and them removed from the set of $\widehat{K}^l$. In this regard, the following formula is defined:

$$k = \{k \in \widehat{K}^l | \operatorname{argmin}(w_k) \} \tag{5-26}$$

where k is sub-set of non-contract requests with the lowest volume which must be removed from the current solution and this operator cannot be used if the set of $\widehat{K}^l$ is empty.

5.4.6 Cluster request-based removal

In this operator, the destruction of one of the open services with the maximum accepted non-contract shipper-demand requests, is chosen. Then, a bunch of requests related to this service should be removed. This cluster contains a set of requests where pickup and delivery times of non-contract requests are as follows:

$$k = \left\{k, k' \in \widehat{K}^l \middle| |\alpha_k - \alpha_{k'}| + |\beta_k - \beta_{k'}| \leq \Delta \text{ and } \sum_{k \in \widehat{K}^l} x_a^{kl} + \sum_{k' \in \widehat{K}^l} x_a^{k'l} = 1,\right\} \tag{5-27}$$

where $k, k'$are the sub-set of $\widehat{K}^l$defining a cluster of shipper-demand requests which have a deviation of Δ which are assigned to the same service meaning their pickup and delivery times are close to each other. For example, we have four non-contract requests (k_1, k_2, k_3, k_4) associated with a service *a*, in a time-space network where *T*=12. Their pickup and delivery times are (4, 6), (2, 9), (4, 7) and (5, 11) if the maximum allowable deviation is $\Delta = 4$. In this regard, the cluster of (k_1, k_3) is selected to be removed from this service.

5.4.7 Hybrid service-request-based removal

Another removal operator is a combination of the above efficient ideas of other removal operators. In this regard, we have considered their advantages for selecting a service and deleting a bunch of requests based on the following principles:

- Minimum service usage should be found. As given in Eq. (5-23), the service that has fewer requests for the sender is selected. Such services are not efficient for designing a service network schedule.

- Selection of requests is based on satisfaction of the time windows which are close with each other. For a service, if the time window of requests is close to each other, they are grouped in a cluster similar to Eq. (5-27).

In summary, hybrid service-based deletion is defined as follows: First, a low-utilization service given in Eq. (5-23) is selected. Then, a group of requests that have a very close time window in this service are selected to be removed from Eq. (5-27).

5.4.8 Random-request-based removal

Finally, the last removal operator selects the requests randomly to destroy a solution. In this regard, a bunch of requests removed have been selected randomly. Accordingly, we select $\widehat{K}^l$ meaning the request is accepted at an iteration as given in the Eq. (5-24). These requests would be selected randomly and them removed from the set of $\widehat{K}^l$. In this regard, the following formula is defined:

$$\check{k} = \left\{k \in \widehat{K}^l \middle|\, z_k^l = 1 \; where \; k \; is \; randomly \; selected \right\} \tag{5-28}$$

where $\check{k}$ is sub-set of $K = K^C \cup K^{NC}$. The number of removed requests is a random number between one and the maximum number of requests. It should be noted that contract-based requests must be assigned to other services in the insertion operators.

5.5 Insertion operators

After destroying the current solution, the requests and/or services which are not the part of current solution, either because they are just removed or there was no feasible place for them, are considered for reconstruction, here in our insertion operators. Then, this repaired solution is compared with the best-known solution in the current iteration based on Eq. (5-9) to update this best-known solution ever found in the algorithm. Here, there are four insert operators, two of which focus on requests and one of which focuses on services and the last one considers both requests and services randomly.

5.5.1 Maximum-volume request insertion

This insertion operator aims to insert new requests which may be contract-based or non-contract requests which have not been accepted. For this purpose, the following set of requests that have the maximum volume is selected as follows:

$$k = \{k \in K | \operatorname{argmax}(w_k)\ if z_k^l = 0\} \quad (5\text{-}29)$$

The number of selected shipments from the above set is based on the authorized capacity of open services. The assignment of requests to services is based on the priority algorithm presented in Figure 5.3, in which the type of requests is available to the transportation services based on the maximum volume. The other decision variables are updated as described in the pseudocode in Figure 5.4.

5.5.2 Maximum-net profit request insertion

Here, this operator aims to insert new requests that may be contract-based or non-contract requests which have not yet been accepted in this iteration. For this purpose, we have defined the following subset:

$$k = \{k \in K | \operatorname{argmax}(\rho_k)\ if z_k^l = 0\} \quad (5\text{-}30)$$

As done in the previous insertion operator, the number of selected sender requests is based on the authorized capacity of the open service. This allocation is done by the priority-based algorithm presented in Figure 5.3 based on the maximum net profit of the requests. The rest of the decision variables are evaluated similarly in Figure 5.4.

5.5.3 Minimum-cost service insertion

If the current services do not have enough capacity to insert a new shipment, we need to open a number of existing bunch services separately, and based on this, we can apply the minimum service operator to repair this solution. In this regard, among the shipments from the set of constraints (4-2), if a service is available and not open with the lowest service cost, a transportation service is replaced. This new transportation service can be found with the following equation:

$$\breve{a}^l = \{a \in (A^E - A^l) | \operatorname{argmin}(tc_a)\ if \sum_{k \in K} x_a^{kl} = 0\} \quad (5\text{-}31)$$

After this allocation, the other decision variables are updated, and the solution is calculated as shown in Figure 5.4.

5.5.4 Random request-service insertion

The latest operator helps to add requests and services randomly. In this method, we must first assign contract-based requests that have been deleted by the removal operators to the services. If the available services have the required capacity, the requests will be randomly assigned to one of these services where we first consider the contract-based and then non-contract ones. Otherwise, another service that closes in this iteration is randomly selected to be open. For all allocations, the algorithm first examines the remaining capacity of the services and then the time interval as shown in Figure 5.4.

5.6 Local search operator

As mentioned earlier, the proposed ALNS-SA-based algorithm also uses a local search operator to make very small changes on the current best-known solution ever found by the pair of removal-insertion operator in each iteration. In this regard, there is a maximum number of sub-iterations in the main loop of the proposed algorithm to exploit current solution while giving this chance to escape from local optimal solutions found by the pairs of removal-insertion operator. The main components of the proposed local search operator are to remove one or two requests randomly and then insert them to make a small change to the current best-known solution. The following steps are done in the local search operator:

Step 1: Specify one or two requests which should be removed from the current accepted requests in the set $z_k^l = 1$.

Step 2: Update the decision variables for x_a^{kl}, z_k^l and $\hat{y}_a^l$.

Step 3: Assign the contract-based requests which are not accepted yet to a service randomly where the remaining capacity should meet the volume of such shipments and consider the time interval of departure, and arrival times of services.

Step 4: Assign one or two non-contract requests randomly to a service where the remaining capacity should meet the volume of such shipments and consider the time interval of services for the departure and arrival time.

Step 5: Update auxiliary decision variables and the objective function for this solution.

Step 6: Exchange this new solution with the current best-known solution ever found (S_{best}) if it is better. Otherwise, go to **Step 7** to check the termination criterion.

Step 7: If the local search operator is terminated, output the best solution ever found (S_{best}). Otherwise, go to **Step 1**.

CHAPTER 6 COMPUTATIONAL EXPERIMENTS

A comprehensive analysis is done to assess the performance of our metaheuristic and its components. Firstly, the test problems are studied and generated on different scales and levels. Then, the performance of our constructive heuristic algorithm is evaluated by the solution quality, i.e., the value of the objective function and its deviation to the final solution obtained by the ALNS metaheuristic. We then analyze the efficiency of removal and insertion operators while studying the solution quality and CPU time. It should be noted that all the data and codes of heuristics, operators and metaheuristics were run on MATLAB software R2013a using a 64-bit 2.5 GHz Intel (R) Core (TM) i7 operating system and 8 GB of memory.

6.1 Test instances

This thesis generates different test instances while analyzing the complexity of our optimization model and showing the performance of the solution algorithm. To have an unbiased comparison of our metaheuristic algorithm, different sets of instances must be used, i.e., sets A and B. Hence, these sets of instances are used where both of them have the same sizes using their random functions to generate the parameters. In addition, to show the high performance of algorithm, we consider two versions of this algorithm, i.e., random and calibrated versions where the calibrated version is resulted from solving the set of instances A. These versions are employed to compare the results of our metaheuristic algorithm with the exact solver. To have a general view on our instances solved and analyzed in Section 6.2 and 6.3, the following justifications must be clarified:

- Two sets of instances including A and B are generated randomly which have the same size (Table 6.1) while using random distribution functions (Table 6.2).
- The calibration of parameters for our metaheuristic is done by using the instances A. In this regard, the parameters of our metaheuristic are tuned to find the best performance of our metaheuristic algorithm using instances A (See Section 6.2).
- Evaluation of components of our metaheuristic algorithm is done by using instances A similar to calibration section (See Section 6.3.1 and 6.3.2).
- For the comparison of our metaheuristic with the exact solver, this study uses the instances B which have been generated randomly in a same size as done in the instances

A. These instances are solved in Section 6.3.3 to have unbiased comparison for our metaheuristic algorithm where the main metaheuristic is divided into two cases, i.e., random, and tuned versions of our metaheuristic algorithm.

Each set of instances, i.e., A and B includes 13 random tests in different sizes including small, medium, large, and very large sizes. In this regard, four complexity levels from small, medium, large, and very large scales using P1 to P3; P4 to P6; P7 to P9 and P10 to P13 are respectively considered where Table 6.1 reports the size of test problems. For these scales, 40, 80, 120, 140, and 160 requests are applied. As such, the total number of services is 170, 210, 250, 270 and 290 for these tests. The number of time periods is set as 7, 14, and 21 periods. We have considered 30 percent of requests for urgent requests and the rest of them are the standard ones. The same classification is considered for fast and regular transportation services.

The data generation is explained in Table 6.2 to show how the parameters are valued and simulated for all instances in both sets of A and B. In this regard, the functions of MATLAB software were employed. To make our test problems reliable, their logic is taken from the base paper of Taherkhani et al. [10]. For example, the travel time for urgent requests is lower than the individual ones. The fixed cost of transportation services is greater than the transportation costs. The data generation reported in Table 6.2 is feasible. There is no redundancy in data generation since there is a reasonable logic for producing all input parameters. The range of parameters is based on the time-space to be logical with regards to the problem definition and problem settings presented earlier.

Table 6-1 Size of test problems for both sets of A and B

Complexity levels	Tests	Number of requests				A^E	$\|T\|$
		$\|K^C\|$		$\|K^{NC}\|$			
		Standard	Urgent	Standard	Urgent		
Small	P1	30		10		170	7
		21	9	21	9		
	P2	10		30		170	7
		7	3	7	3		
	P3	20		20		170	7
		14	6	14	6		
Medium	P4	50		30		210	14
		35	15	35	15		
	P5	30		50		210	14
		21	9	21	9		
	P6	40		40		210	14
		28	12	28	12		
Large	P7	90		30		250	21
		63	27	63	27		
	P8	30		90		250	21
		21	9	21	9		
	P9	60		60		250	21
		42	18	42	18		
Very large	P10	100		40		270	7
		50	50	50	50		
	P11	80		60		270	7
		40	40	40	40		
	P12	60		80		270	7
		30	30	30	30		
	P13	110		50		290	7
		55	55	55	55		

Table 6-2 Range of parameters for both sets

Parameters	Range
ρ_k	*1000+2000*rand(1,K)*
w_k	*(0.8+0.5*rand(1,K)).* ρ_k/10*
α_k	*randi([1 T-1],1,K)*
β_k for standard requests	*min(α_k+randi([floor(T/5),ceil(T/3)],1,1),T)*
β_k for urgent requests	*α_k +min(α_k +randi([floor(T/10),ceil(T/5)],1,1),T)*
$\psi^k_{(o_{(k)},t)}$	*(0.2+0.3*rand(1,1))* w_k*
$\psi^k_{(d_{(k)},t)}$	*(0.2+0.3*rand(1,1))* w_k*
f_a	*200+200*rand(1,A)*
c^k_a	*0.5+1.5*rand(1,A)).* w_k*
$\bar{c}^k_a$	*(0.1+0.2*rand(1,1))* w_k*
oc^k_a	*0.3*(max(c^k_a)+max(f_a)- ρ_k)*
u_a	*(2+2*rand(1,A))*sum(w_k)/A*
$u^{MH}_{(o_{(k)},t)}$, $u^{MH}_{(d_{(k)},t)}$	*0.25*sum(w_k)*
$\alpha_{(a)}$	*randi([1 T-1],1,A)*
$\beta_{(a)}$	*min($\alpha_{(a)}$ +randi([1,2]),T)*

**rand* is a function to generate random continuous numbers between zero and one.
**randi* is a function to generate random integer numbers between a lower bound and an upper bound.
**sum* is a function to sum a matrix.
**round* is a function to transform continuous numbers to integer ones while rounding them.
* *floor* is a function to transform continuous numbers to integer ones while cutting them minimally.
* *ciel* is a function to transform continuous numbers to integer ones while cutting them maximally.
* *min* is a function to select the minimum array in a matrix.
* *max* is a function to select the maximum array in a matrix.

6.2 Parameter settings

Generally, in any paper to use a metaheuristic for solving an optimization model, it is important to tune its parameters [51-52]. A well-tuned metaheuristic shows the highest level of efficiency for solving the test problems on average [53-54]. Generally, the ALNS is very sensitive to the scores based on the performance record and the calibration seems necessary in the literature before the comparison of its overall performance with an exact solver or other metaheuristics [55-56]. From another point of view, one disadvantage of the proposed ALNS-SA-based metaheuristic is to have many input parameters which they can be listed as follows:

$MaxIt$	Maximum number of iterations
$SubIt$	Maximum number of sub-iterations
$redu$	Rate of reduction of temperature
Tem	Initial temperature
Q	Percentage of removed requests or services
ϖ_1	Score for the first case
ϖ_2	Score for the second case
ϖ_3	Score for the third case where ϖ_3=$1-\varpi_1-\varpi_2$;

Based on the above notations, the proposed metaheuristic algorithm has eight parameters where the third case of score parameter is dependent on other cases and cannot be considered as an independent main parameter. Therefore, we have seven main parameters and if we consider three candidate values for each of them, the total number of experiments using a full factorial method is 37=2187. It is not logical to run the algorithm 2187 times for each test. To this end, this study applies the Taguchi experimental design method defined in the paper of Taguchi and Jugulum [9] to reduce the number of runs for each test. The Taguchi method works with a set of predefined orthogonal arrays while using two popular metrics to analyze the performance of a tuned level of metaheuristic algorithm. In this regard, Signal to Noise (S/N) and Relative Percentage Deviation (RPD) are two metrics to control the quality of candidate values for the algorithm's parameters. For a maximization optimization model like our model, the S/N ratio is computed as follows:

$$S/N = 10\log_{10}(objective\ function)^2 \tag{6-1}$$

where the objective function is the amount of average objective functions for all the test problems. A higher value of S/N ratio is preferable in our analyses.

In addition, addition, the RPD metric is employed to analyze the efficiency of a tuned level of metaheuristic algorithm. For the proposed optimization problem as a maximization problem, the RPD is formulated as follows:

$$RPD = \frac{Max_{sol} - Alg_{sol}}{Max_{sol}} \tag{6-2}$$

For each run of the metaheuristic algorithm (i.e., there are 27 runs total), we want to solve all the tests, i.e., P1 to P9 and then its average of the objective function is considered. $Maxsol$ is the maximum value among all the average of test problems while $Algsol$ is the output of our metaheuristic algorithm for each experiment. A lower value for the RPD is preferable for a tuned level of a metaheuristic.

As mentioned earlier, the proposed algorithm has seven main parameters and if we consider three candidate values for each parameter, Table 6.3 defines the values of each candidate value where the Taguchi method suggests the orthogonal array L27 for the levels of our metaheuristic in this table. In this regard, we should run each test problem 27 times based on the orthogonal array. After running them, the results of tuning are provided in Table 6.4 to study the average of RPD and S/N ratio in the proposed problem. It should be noted that all these runs were implemented on the test instances from the set instance of A.

Table 6-3 The list of parameters of ALNS-SA-based algorithm

Parameter	Levels		
	1	2	3
$MaxIt$	500	1000	2000
$SubIt$	20	30	50
$redu$	0.9	0.99	0.999
Tem	10000	15000	20000
Q	0.1	0.3	0.5
ϖ_1	0.2	0.4	0.6
ϖ_2	0.1	0.3	0.4

Table 6-4 . Experiments of tuning of ALNS-SA-based metaheuristic using the orthogonal array

Number of experiments	Parameters and their levels							Mean RPD	Mean S/N
	$MaxIt$	$SubIt$	$redu$	Tem	Q	ϖ_1	ϖ_2		
L1	1	1	1	1	1	1	1	0.145436	90.71379
L2	1	1	1	1	2	2	2	0.168041	90.48094
L3	1	1	1	1	3	3	3	0.146809	90.69983
L4	1	2	2	2	1	1	1	0.126381	90.90534
L5	1	2	2	2	2	2	2	0.140304	90.7658
L6	1	2	2	2	3	3	3	0.12838	90.88544
L7	1	3	3	3	1	1	1	0.087035	91.28798
L8	1	3	3	3	2	2	2	0.07917	91.3625
L9	1	3	3	3	3	3	3	0.088988	91.26938
L10	2	1	2	3	1	2	3	0.105347	91.11199
L11	2	1	2	3	2	3	1	0.103312	91.13173
L12	2	1	2	3	3	1	2	0.117483	90.99336
L13	2	2	3	1	1	2	3	0	92.0789
L14	2	2	3	1	2	3	1	0.093936	91.22208
L15	2	2	3	1	3	1	2	0.08953	91.26421
L16	2	3	1	2	1	2	3	0.078804	91.36594
L17	2	3	1	2	2	3	1	0.063706	91.50714
L18	2	3	1	2	3	1	2	0.070892	91.44023
L19	3	1	3	2	1	3	2	0.074078	91.41039
L20	3	1	3	2	2	1	3	0.071467	91.43485
L21	3	1	3	2	3	2	1	0.086067	91.29719
L22	3	2	1	3	1	3	2	0.054117	91.59565
L23	3	2	1	3	2	1	3	0.051332	91.62118
L24	3	2	1	3	3	2	1	0.071293	91.43648
L25	3	3	2	1	1	3	2	0.044896	91.67991
L26	3	3	2	1	2	1	3	0.042308	91.70342
L27	3	3	2	1	3	2	1	0.04795	91.6521

To study which value of the parameters has the highest performance, we should compute the mean RPD and S/N ratio for each level for the parameters of our metaheuristic algorithm. The results of our analyses are shown in Figure 6.1 and 6.2 where a higher value of S/N ratio is preferable while a lower value of RPD brings a better capability for the proposed algorithm. After looking at these figures, we can determine which level is the most suitable and efficient for the proposed algorithm. Based on our evaluations, the tuned value of each parameter is reported in Table 6.5.

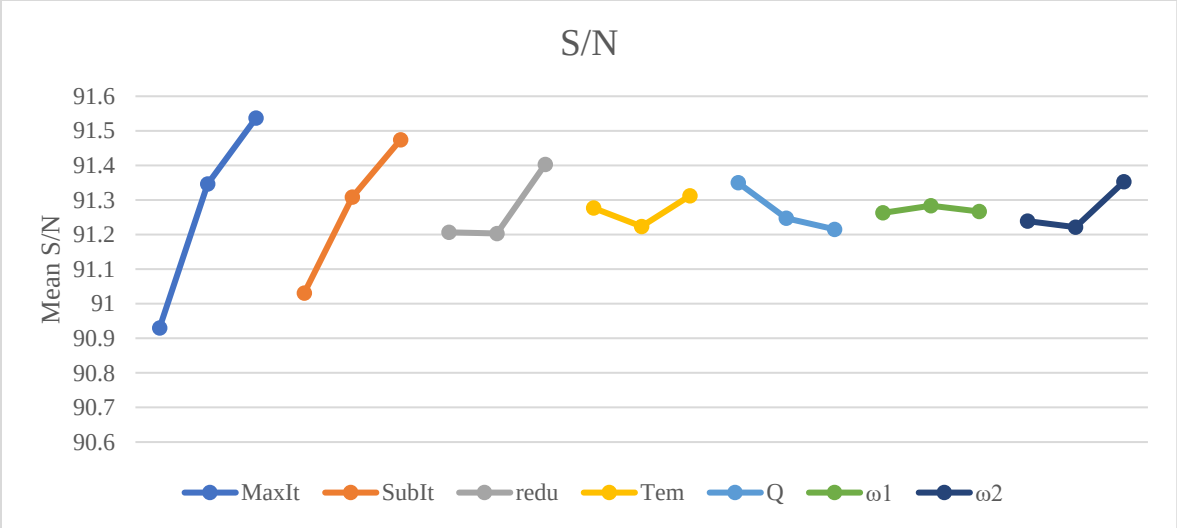


Figure 6-1 The behavior of ALNS-SA-based in the term of S/N ratio

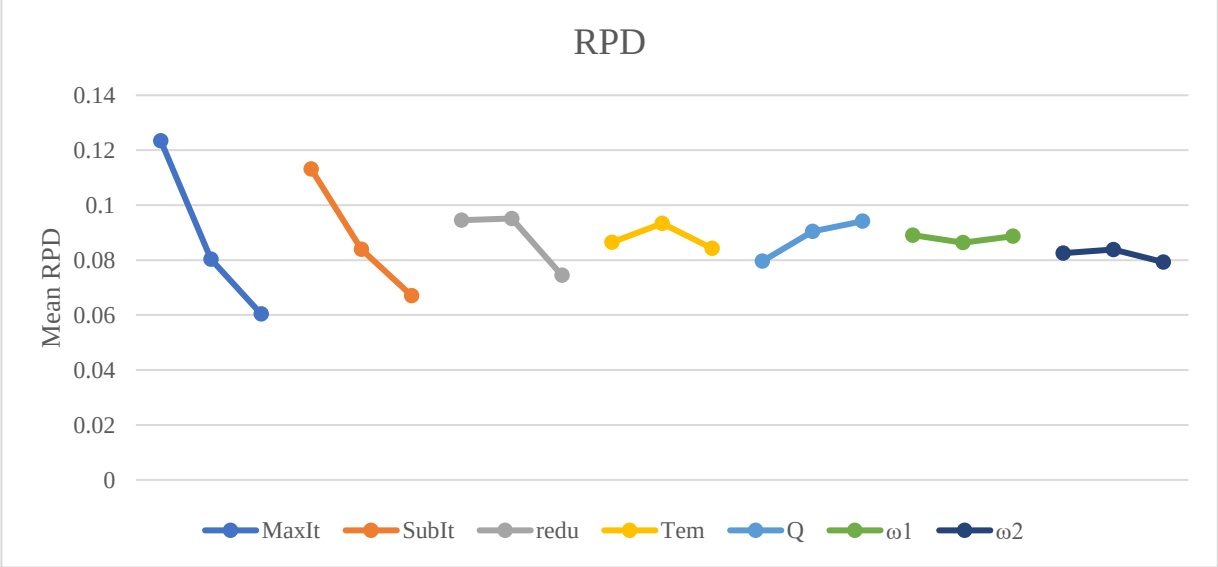


Figure 6-2 The behavior of ALNS-SA-based in the term of RPD

Table 6-5 Tuned values of our metaheuristic's parameters

Metaheuristic	Parameter	Best level
ALNS-SA-based algorithm	$MaxIt$	2000
	$SubIt$	50
	$redu$	0.999
	Tem	20000
	Q	0.1
	ϖ_1	0.4
	ϖ_2	0.4
	ϖ_3	0.2

6.3 Evaluation of algorithmic components

After the confirmation to have a well-tuned metaheuristic, we can run it while doing some sensitivity analyses for evaluating its performance. Here, we first analyze the performance of our constructive heuristic while studying the solutions' quality of the metaheuristic algorithm and then, comparing the solution of the constructive heuristic with the final solution obtained by the full algorithm. Then, a comprehensive analysis to evaluate the efficiency of removal and insertion operators to show the performance of operators individually. Finally, the overall performance of our ALNS- SA-based algorithm is evaluated by the exact solver and then convergence analysis to the optimality for each test problem is done. It should be noted that to have an unbiased comparison, we have used the instances set B instead of A which has been used for the calibration.

6.3.1 Evaluation of our constructive heuristic algorithm

Based on the concept of ad-hoc arcs or the extra capacity, we have defined a very large negative value of the objective function for unaccepted contract-based requests. It means that a negative value of the objective function for a solution means that it includes one or more unaccepted contract-based requests. However, a positive value for a solution means that all contract-based requests have been accepted. For the proposed constructive heuristic algorithm, it is possible to have some negative values for the objective function of initial solutions as presented in Table 6.6. Here, we first check if the initial solution found by the constructive heuristic is positive or negative. Then, the improvement in the table means that we check that how this initial solution is close or

far from the final solution obtained by the metaheuristic. It is computed by the relative deviation of the initial solution from the final solution obtained by the ALNS.

From the results presented in Table 6.6, the constructive heuristic algorithm is able to find high-quality solutions in small instances, i.e., P1 to P3. However, for large-scale test problems, the initial solution obtained by the constructive heuristic is not as good as small ones since there are some contract-based requests which are not accepted. This highlights the role of our search space using the removal-insertion operators to improve this initial solution by our metaheuristic algorithm. This table also reported the final solution obtained by the proposed metaheuristic and the CPU time for all iterations. Having a look at the results, we can compare check the quality of solutions in the proposed metaheuristic algorithm.

Another important criterion is to analyze the amount of improvement by the main metaheuristic on this initial solution. To compute this improvement, we compute the gap between the solution from the constructive heuristic and the one obtained by the metaheuristic algorithm. If the initial solution will be close to the final solution found by the metaheuristic, it confirms its high performance. The gap between the heuristic and the final solution from the metaheuristic in very large scales, i.e., P10 to P13, is around 0.2. Based on these evaluations, Figure 6.3 shows this fact that the initial solutions for P1, P2, P3 and P6 as well as P10 to P13 test problems, are optimal and close to the final solution obtained by the metaheuristic.

Table 6-6 Evaluation of our constructive heuristic and the overall ALNS-SA-based metaheuristic.

Number of tests problem	Constructive heuristic algorithm		Metaheuristic algorithm	
	Solution	Improvement done by the metaheuristic	Solution	CPU time (Seconds)
P1	39176	5.79%	41583	41.70
P2	28919	27.44%	39857	41.25
P3	41278	13.05%	47476	40.83
P4	-6500.6	114.77%	44001	57.22
P5	-13779	131.61%	43588	57.22
P6	4929.4	90.23%	50460	62.02
P7	-141300	592.52%	28688	69.61
P8	-141750	612.31%	27669	65.73
P9	-109300	509.25%	26708	72.53
P10	147796	19.32%	183180.3	38.01
P11	145972	20.30%	183150.6	36.09
P12	132906	20.41%	166994.5	36.99
P13	158745	22.18%	203990.4	42.70

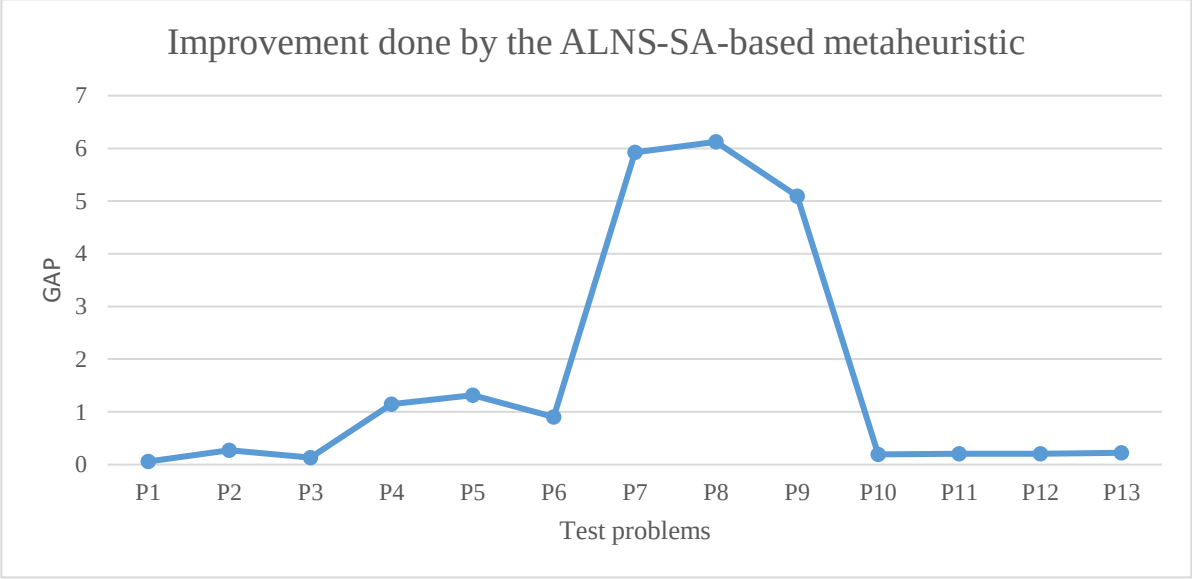


Figure 6-3 The evaluation of the gap improved by the metaheuristic.

6.3.2 Comparison of removal-insertion operators

One significant contribution of our metaheuristic algorithm is to have different removal and insertion operators to destroy a solution and then repair it efficiently. One open question is that among these operators, which one is the most efficient and has the key role to help the main algorithm for finding an optimal solution. The goal of this sub-section is to analyze each pair of removal-insertion operators and select the most efficient ones. It should be noted that the comparison of removal and insertion operators individually, is done in the literature review of ALNS multiple times [55-56].

In this regard, one test problem, here, P5 as a medium-size test problem is selected to do these analyses. Based on the literature of ALNS, a medium-size test problem can better evaluate the performance of the proposed algorithm [55-56]. To do our sensitivity analyses, we redesign the tuned metaheuristic's parameters to use one pair of removal-insertion operators in all run times. Since there are eight removal operators and four insertion operators, there are 32 times for our sensitivity analyses on the performance of removal-insertion operators. Table 6.7 are the results of our analyses accordingly. We have three main criteria for each analysis including the objective function, CPU time and the amount of improvement done by the metaheuristic in comparison with the initial solution.

The variations of the solutions obtained by the pair of removal-insertion operators show that there is no significant difference between the CPU time. However, to analyze the performance of each operator, Fig. 6.4 shows the comparison for each pair of removal-insertion operators based on the solution quality while analyzing the improvement done by the main algorithm.

Table 6-7 Comparison of removal-insertion operators

Index of runs	**Removal operators**	**Insertion operators**	**Objective function**	**CPU Time (Seconds)**	**Relative changes of the solutions by the pair of removal-insertion operator**
T1	Random service-based removal	Maximum-volume request insertion	4.3708e+04	54.41	1.33
T2		Maximum-net profit request insertion	4.4400e+04	58.11	1.32
T3		Minimum-service cost insertion	4.8739e+04	61.45	1.28
T4		Random request-service insertion	4.4422e+04	56.16	1.31
T5	High-cost service-based removal	Maximum-volume request insertion	4.4536e+04	55.67	1.31
T6		Maximum-net profit request insertion	4.2703e+04	56.87	1.32
T7		Minimum-service cost insertion	4.6831e+04	55.70	1.30
T8		Random request-service insertion	4.5418e+04	45.79	1.32
T9	Low-utilization service-based removal	Maximum-volume request insertion	4.4365e+04	46.56	1.32
T10		Maximum-net profit request insertion	4.6307e+04	51.73	1.29
T11		Minimum-service cost insertion	4.9313e+04	51.72	1.27
T12		Random request-service insertion	4.3711e+04	49.70	1.32
T13	Low-profit request-based removal	Maximum-volume request insertion	4.5288e+04	50.63	1.32

Table 6-7: Comparison of removal-insertion operators (cont'd)

T14	Low-profit request-based removal	Maximum-net profit request insertion	4.4952e+04	51.24	1.29
T15		Minimum-service cost insertion	4.6616e+04	50.64	1.32
T16		Random request-service insertion	4.3855e+04	51.09	1.33
T17	Low-volume request-based removal	Maximum-volume request insertion	4.5043e+04	50.37	1.32
T18		Maximum-net profit request insertion	4.3603e+04	56.81	1.32
T19		Minimum-service cost insertion	4.8284e+04	58.02	1.26
T20		Random request-service insertion	4.5020e+04	50.30	1.31
T21	Cluster request-based removal	Maximum-volume request insertion	4.5060e+04	50.39	1.31
T22		Maximum-net profit request insertion	4.5888e+04	50.90	1.31
T23		Minimum-service cost insertion	4.6405e+04	50.11	1.28
T24		Random request-service insertion	4.4990e+04	49.90	1.32
T25	Hybrid service-request-based removal	Maximum-volume request insertion	4.5539e+04	58.52	1.30
T26		Maximum-net profit request insertion	4.5423e+04	55.69	1.29
T27		Minimum-service cost insertion	4.7207e+04	55.65	1.28
T28		Random request-service insertion	4.3644e+04	56.03	1.31
T29	Random request-based removal	Maximum-volume request insertion	4.4454e+04	56.05	1.31
T30		Maximum-net profit request insertion	4.4439e+04	56.00	1.32

Table 6-7 Comparison of removal-insertion operators (cont'd and end)

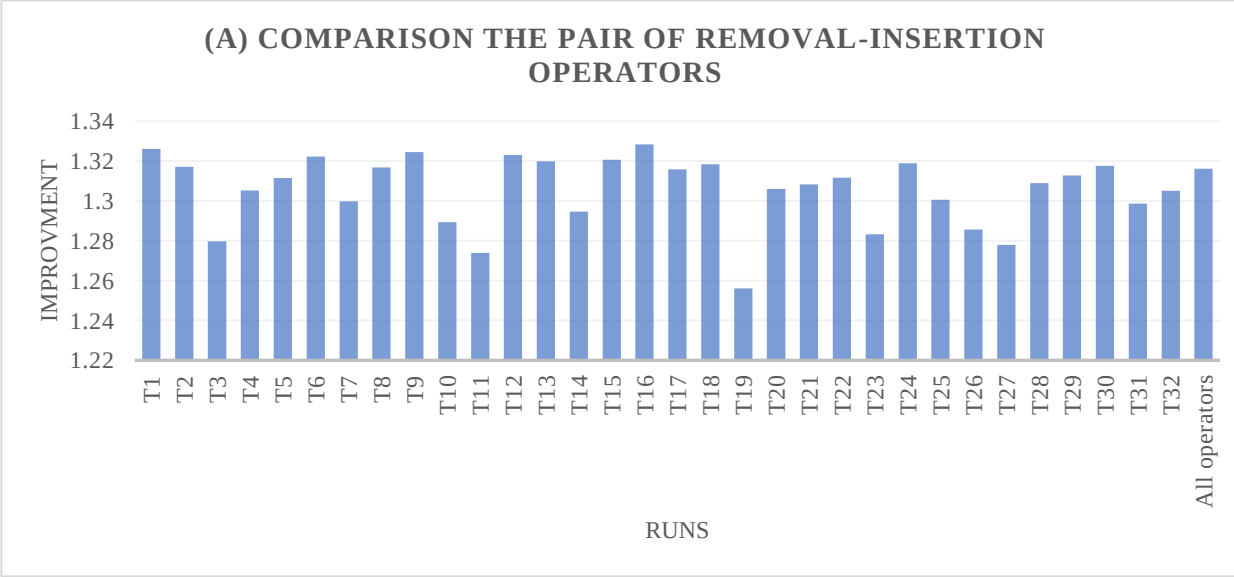
T31	Random request-based removal	Minimum-service cost insertion	4.6818e+04	56.21	1.30
T32		Random request-service insertion	4.3679e+04	57.14	1.31

Figure 6.4 is divided into three sub-figures where we first compare each pair of removal-insertion operators with to gather (Figure 6.4(A)), then study the performance of removal (Figure 6.4(B)) and insertion operators (Figure 6.4(C)) individually.

As shown in Figure 6.4(A), sometimes, the combination of removal and insertion operators is not successful and cannot improve the quality of a solution. Among 32 runs as reported in Table 6.7, the combination of low-volume request-based removal with the minimum-service cost insertion has the weakest performance in comparison with other combinations. Contrary to this combination, the solution from the low-volume request-based removal with the minimum-service cost insertion operators, is highly efficient and outperforms other pairs of removal-insertion operators. To analyze each operator individually, we computed the average of results for each operator as reported in Table 6.7.

What can be concluded from Figure 6.4(B) is that the performance of hybrid service-request-based removal operator has the lowest performance in comparison with other operators. However, the low-profit request-based removal has the highest efficiency in this comparison.

As indicated in Figure 6.4(C), the performance of minimum service cost insertion operator has the lowest efficient case. However, the maximum-volume request insertion operator shows the highest performance in this comparison. In the next section, we will compare the final algorithm with the exact solver.



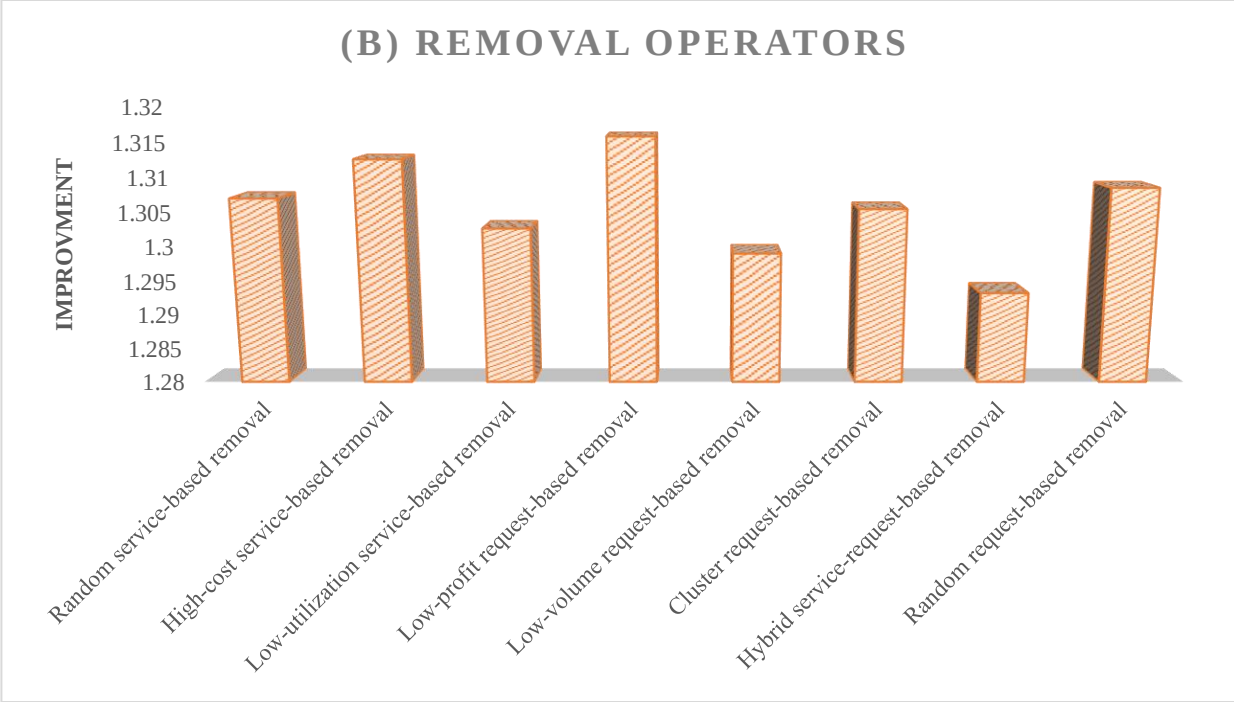


Figure 6-4 Comparison of operators based on the solution quality

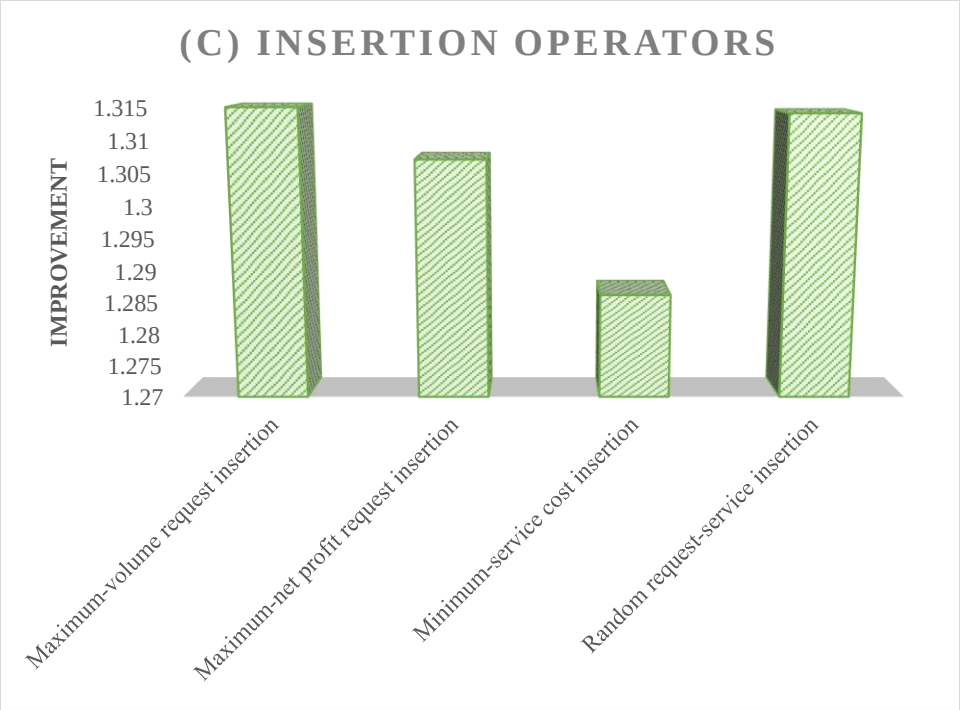


Figure 6-5 Comparison of operators based on the solution quality (cont'd and end)

6.4 Comparison with the exact solver

Here, we compare the proposed metaheuristic algorithm with the exact solver. As mentioned earlier, there are two versions for the proposed metaheuristic algorithm where the first one is the tuned version, and the second one is our metaheuristic with random values of parameters. For the random set of parameters, we have defined the values of parameters as reported in Table 6.8.

Table 6-10 The random set of parameters of our metaheuristic algorithm

Metaheuristic	Parameter	Value
ALNS-SA-based algorithm	$MaxIt$	2000
	$SubIt$	30
	$redu$	0.9
	Tem	10000
	Q	0.5
	ϖ_1	0.2
	ϖ_2	0.4
	ϖ_3	0.4

To have a fair and unbiased comparison, all the test problems have been generated randomly where the number of time periods is set to seven. This set of instances is called as set of B. If we can consider the test problems generated for Table 6.4 and 6.5 using as the set of A, the test problems here solved in Table 6.9 can be considered as the instance set B which has the same size in comparison with the instance set of A which had been used for the calibration. We have used the random set of parameters for the instance set B which was presented in Table 6.8. We should say that both sets of instances are different since they are using random functions given in Table 6.2.

For the instance set B, we also should say that our exact solver was so time-consuming for solving such test problems with more than seven time periods that is why we have generated all the test problems. The termination criterion was 3600 seconds for the exact solver, and we can find an optimal solution with no absolute gap between the upper and upper bounds. It should be noted that the exact solver was implemented by the CPLEX solver from GAMS 24.7.4 software.

To analyze our metaheuristics statistically, for each instance, we have run the algorithms for experience. From the literature, since the metaheuristics are random, we should run them for more than 5 times to have a reliable result [55]. From these run times, we have reported the best, the worst, the average and the standard deviation of our solutions. It means that among these run times, the best solution is the maximum solution ever found during these solutions. The worst solution is the minimum one. The average and standard deviation of these solutions is also reported. Moreover, the average CPU time and optimality gap in comparison with the solution found by the exact solver is provided. Although a higher value for the best solution, worst solution, and average

solution, is preferred, a lower value for the standard deviation, CPU time and the optimality gap are in our interest. All these results are provided in Table 6.9.

Having a general overview, in Table 6.9, we can find that our metaheuristic in both cases of random version and tuned version create very high-quality solutions. The criteria of CPU time and optimality gap are analyzed in Figure 6.5. To show that the CPU time of our metaheuristic is reasonable in comparison with the exact solver, Fig. 6.5(a) is referred. To confirm that both random and tuned versions of our metaheuristic can find strong solutions with low optimality gap, Figure 6.5(b) is available. To analyze the robustness of our solutions, the best solution ever found by our metaheuristic, is studied in Figure 6.6. Finally, to study the accuracy of our metaheuristic in both versions, some statistical analyses using 95% confidence level where the standard deviation of our solutions is normalized and accordingly, the interval plot shows in Figure 6.7. It should be noted that our solutions for P7 and P9 are feasible but may not be optimal as the exact solver was not able to find the optimal one.

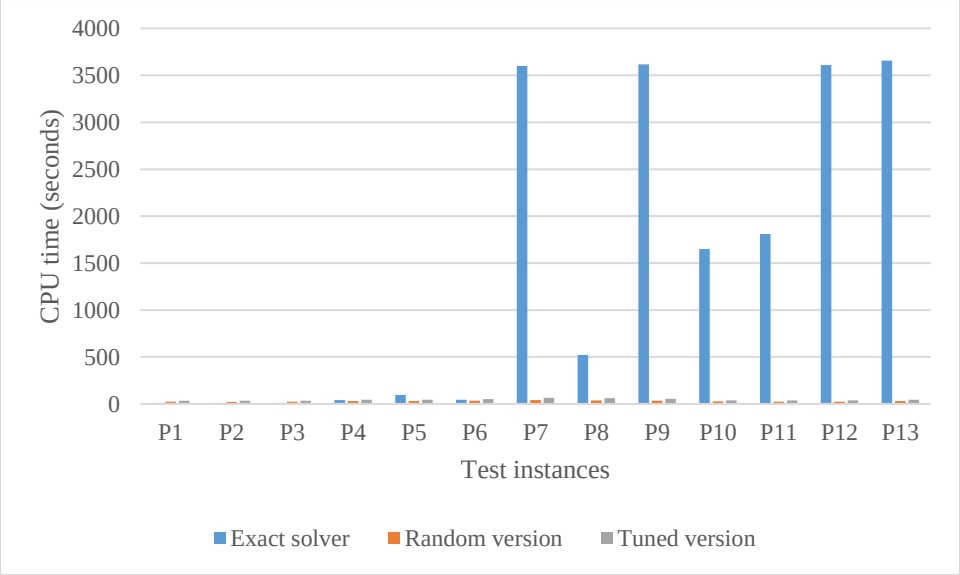
Table 6-11 . The comparison of proposed metaheuristic with the exact solver (the CPU time is in seconds)

Instances	Exact solver		Proposed metaheuristic algorithm											
			Random values of parameters						Tuned values of parameters					
	Solution	Average CPU time	Best solution	Worst solution	Average solution	Standard deviation	Average CPU time	Optimality gap	Best solution*	Worst solution*	Average solution	Standard deviation	Average CPU time	Optimality gap
P1	55018.00	10.99	48186.62	44380.34	46415.83	1323.53	22.57	0.12	48688.67	47487.25	48181.73	443.73	34.72	0.12
P2	47994.00	10.97	41793.48	37856.74	38639.22	1357.04	21.91	0.13	41711.86	39317.81	40733.10	735.68	34.50	0.13
P3	50618.00	9.70	42013.43	40171.80	40666.70	514.82	22.31	0.17	43487.92	41922.08	42743.44	513.16	35.37	0.14
P4	114923.00	40.39	100326.70	99025.82	99730.07	460.63	29.38	0.13	100732.20	99671.84	100195.40	367.98	45.79	0.12
P5	111927.00	96.00	97408.37	92502.05	93751.07	1589.61	31.34	0.13	96495.90	94097.13	95485.40	655.51	44.44	0.14
P6	108779.00	45.34	93978.73	91878.73	92807.36	704.36	34.09	0.14	94082.44	91902.58	93117.70	659.29	52.15	0.14
P7*	175268.11	3610.00	154622.60	151808.50	153316.40	948.78	42.25	0.11	155016.10	153151.90	154209.90	563.28	63.33	0.11
P8	178151.00	523.00	154769.60	153023.10	153975.20	572.53	39.12	0.13	156326.40	154017.80	155529.80	706.47	61.62	0.12
P9*	171138.67	3615.00	151217.90	148766.40	150068.50	842.64	35.15	0.11	151404.50	148927.90	150604.10	762.62	53.80	0.11
P10	208721.10	1650.00	182459.90	180325.00	181592.70	735.82	25.93	0.13	184184.00	182327.30	183180.30	794.57	38.01	0.12
P11	209104.10	1810.00	182651.80	180939.50	182087.60	612.21	24.94	0.13	184760.40	181683.20	183150.60	781.80	36.09	0.12
P12*	195581.70	3610.00	166450.80	164851.00	165843.40	560.08	24.69	0.15	167681.20	166059.60	166994.50	662.82	36.99	0.14
P13*	231349.50	3656.00	203137.80	201332.60	202046.50	700.14	29.93	0.12	205267.60	202315.20	203990.40	910.48	42.70	0.11

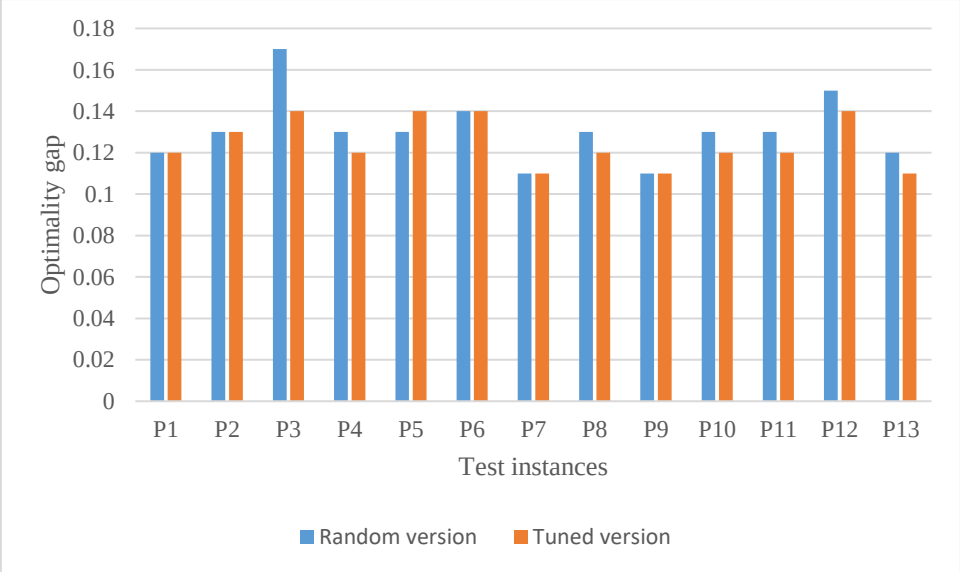
*Those are the best feasible solutions not optimal.

In Figure 6.5(a), for large-scale instances like P8 and very large-scale instances like P10 to P13, the exact solver is too time-consuming where there is a clear difference between the CPU time of our metaheuristic in both versions with the exact solver. However, there is no big difference between our metaheuristics and the exact solver in other test problems. In most of test problems, we can understand that the random version of our metaheuristic algorithm is faster than the tuned version. The main finding from this chart is that our metaheuristic algorithm is able to find an optimal solution quicker than the exact solver.

What can be envisaged from Figure 6.5(b) is that both versions of our metaheuristic have an acceptable optimality gap which is lower than 0.2 in all instances. For small instances, the optimality gap for the tuned version is significantly better than the optimality gap for the random version. Although in all instances, the optimality gap for the tuned version is lower than the one obtained by the random version, their difference in medium and large instances is not significant. For very large-scale instances, the optimality gap is around 10 to 14 percent which is acceptable for solving a very complex optimization problem like SSND for our proposed ALNS.



(a)



(b)

Figure 6-6 Behavior of our proposed metaheuristic based on CPU time (a) and optimality gap (b)

Having a look at Figure 6.6, the main finding is that the quality of our metaheuristic is acceptable as the best solution ever found is very close to the one obtained by the exact solver. Another important finding from this chart is that our metaheuristic algorithm is robust, and it can find a high-quality solution even if it was not tuned. From this chart, we can see that if the algorithm was run ten times, the best solution is very close to each other. In this regard, for solving very large-

scale instances, we can see that the best solution from tuned and random versions, is very close to each other.

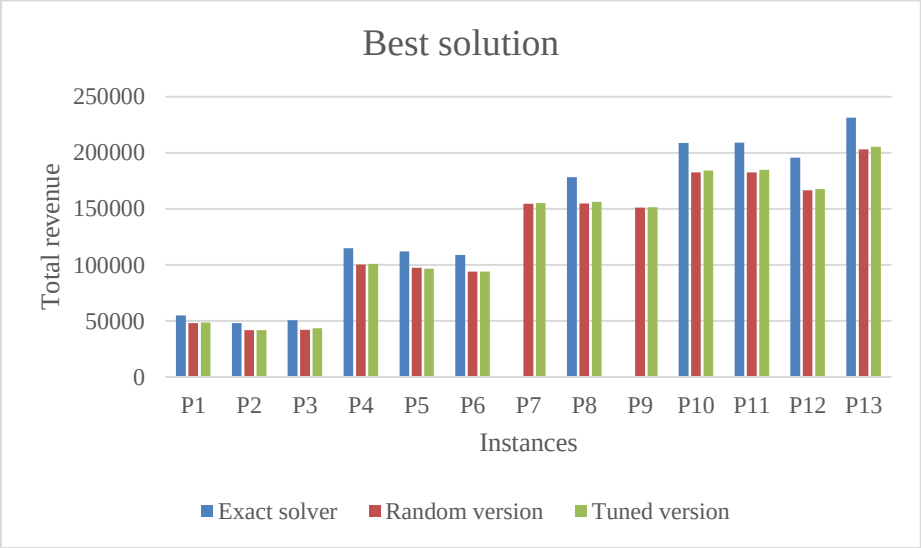


Figure 6-7 Comparison of best solution ever found with the exact solver

What is evident from Figure 6.7 is that the accuracy of our tuned version is much higher than the random version where a lower value for the plot is preferable. This plot was computed by the normalized standard deviations of solutions and the main finding is that if we have used the tuned version of our metaheuristic, the accuracy for finding the optimal solution is higher than the one obtained by the random version of our metaheuristic algorithm.

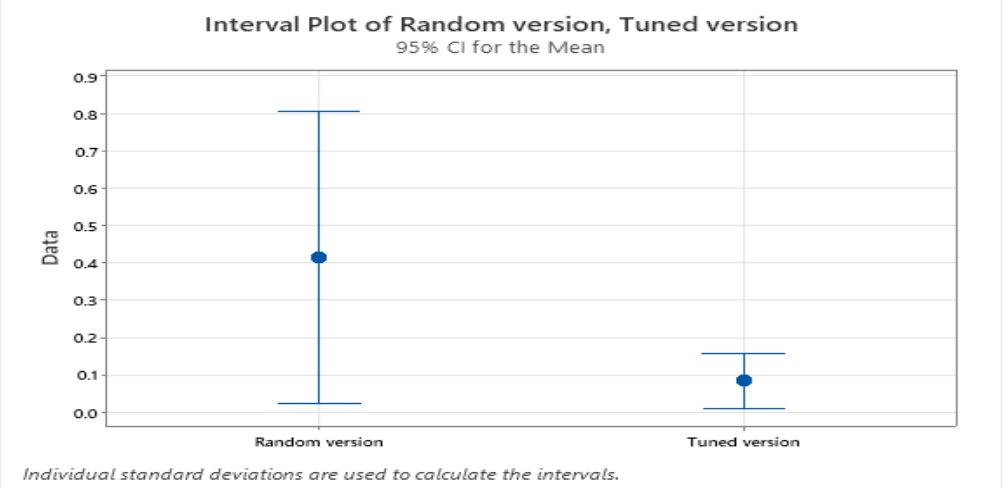


Figure 6-8 Interval plot based on 95% confidence level for normalized standard deviation of solutions

In conclusion, there are two main results from the comparison of the exact solver with our proposed hybrid ALNS-SA-based metaheuristic algorithm in both random and tuned versions. The first one is that in both versions, we can find an optimal solution by our metaheuristic if we run it ten times minimally. For example, the optimality gap for very large-scale instances is around 11% for our metaheuristic algorithms which can be acceptable for solving a very complex optimization problem like the SSND. The last one is that a tuned version of our metaheuristic has better accuracy, and it usually finds the optimal solution in most instances.

6.5 Evolution of the best solution

In this sub-section, more analyses on the performance of the proposed metaheuristic were done by the convergence analysis. Based on the tuned values, the behavior of the proposed algorithm is shown in Figure 6.8 where the behavior of the proposed algorithm is different in each test problem. For small scales, the convergence rate of the proposed algorithm is robust except for P2 where there are some variations in the last iterations. For medium sizes, i.e., P4 to P6, the behavior of our algorithm for solving the proposed model shows a strong convergence rate. The same conclusion can be stated for the large-scale tests, i.e., P7 to P9 where the results are stronger than the medium sizes. Generally, one result is that the proposed algorithm is highly efficient especially for large-scale networks when there are many alternatives transportation services and requests on the planning horizon.

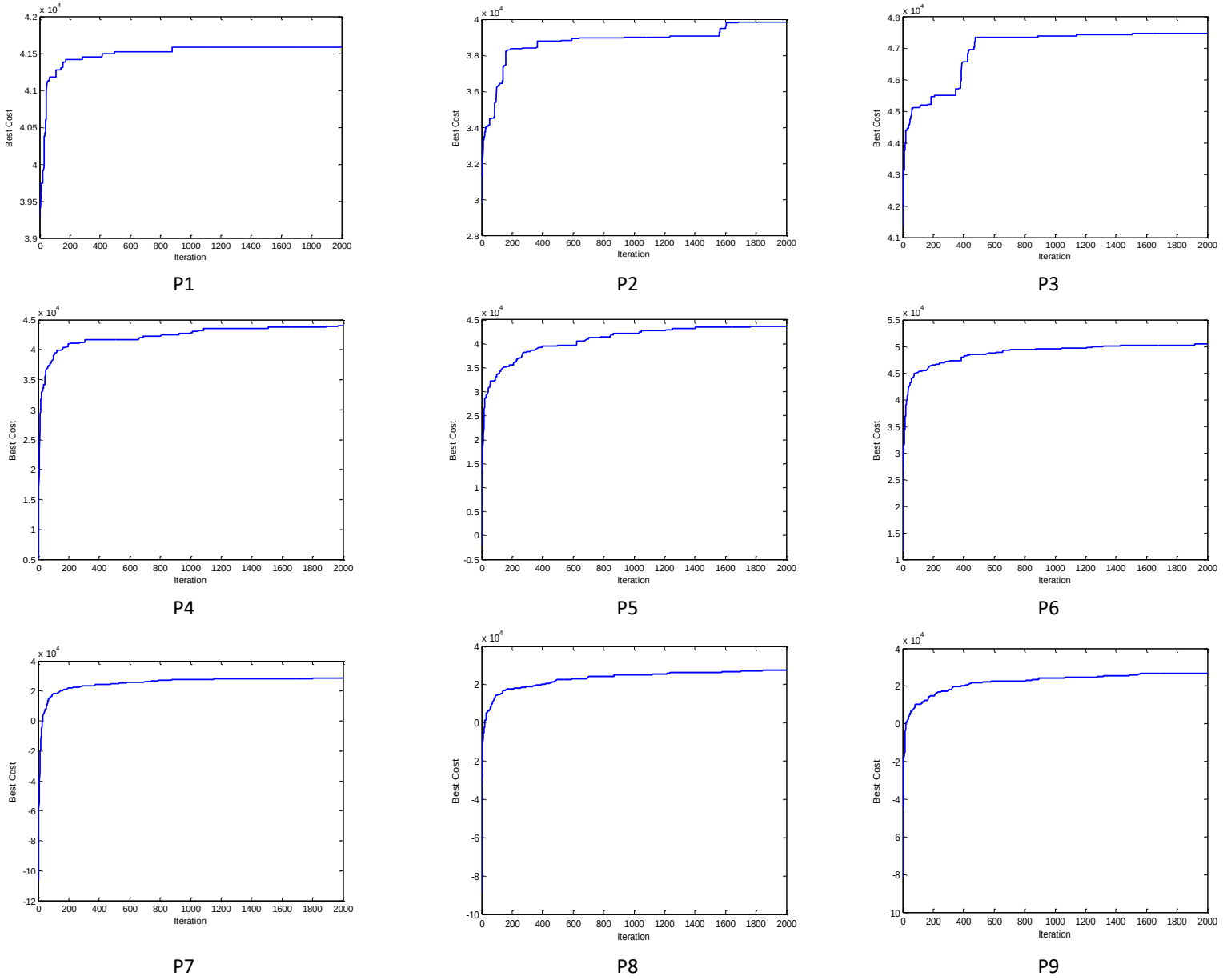


Figure 6-9 Convergence rate of our hybrid ALNS-SA-based metaheuristic in each test problem.

6.6 Discussions

Academically, one general goal of the SSND is to coordinate a set of shippers and carriers while satisfying the requests by a set of alternative decision variables. Although many studies have been done to formulate the SSND, there is no optimization model to formulate a single-segment corridor network design while considering the possibility of extra capacity or ad-hoc arcs for unaccepted requests to define the opportunity loss cost. The proposed maximization model has the option to

accept or reject the requests where there is a big penalty for unaccepted contract-based requests (i.e., their solution has a negative sign) and there is the opportunity loss cost for unaccepted non-contract requests (i.e., their solution has a positive sign). In addition to the development of a new optimization model, a novel metaheuristic based on the combination of ALNS, SA, and local search operator. As far as we reviewed in the literature, there is no study in the area of SSND to employ the ALNS.

The solution is able to select the non-contract requests while considering the opportunity loss cost for unaccepted ones. The solution is also assigned the transportation services based on the capacity constraint and the time interval. In this regard, the proposed ALNS-SA-based metaheuristic starts with an initial solution from a constructive heuristic algorithm. Then, eight removal operators are employed to destroy a solution and then repair it by four insertion operators. This solution is evaluated by the decision rule from SA. Then, a subloop is designed to improve this solution iteratively using minor changes for removing requests and adding them randomly. Based on these characteristics, we can confirm that the proposed metaheuristic is new and has been never applied to solve an optimization problem.

In this section, different test problems were generated in different scales (Table 6.1) and using random numbers generated by MATLAB software (Table 6.2). Since the proposed ALNS-SA-based algorithm has many input parameters, it is essential to tune it to improve its performance (Table 6.3). In this regard, the Taguchi experimental design method was applied to select 27 experiments among 2187 ones for each test problem (Table 6.4). The RPD and S/N ratio metrics were employed to find the most optimal value for each parameter of our ALNS-SA-based algorithm (Table 6.5).

We first compared the solution quality for our constructive heuristic algorithm in comparison with the proposed metaheuristic to show how the proposed metaheuristic was successful to improve the initial solution (Table 6.6). Then, the proposed algorithm was run for each pair of insertion-removal operator individually as reported in Table 6.7. Based on the performance of each pair, all the removal and insertion operators were analyzed and one finding is that low-profit request-based removal (Fig. 6.3) and maximum-volume request insertion (Fig. 6.4) are the most efficient ones.

Another important part of this was the comparison of our metaheuristic with the exact solver as reported in Table 6.8. To confirm that the proposed metaheuristic is strong even it was tuned or a

random set of parameters, i.e., the proposed metaheuristic is divided into two versions. The criteria of CPU time and optimality gap were analyzed in Figure 6.5. To show that the CPU time of our metaheuristic is reasonable Fig. 6.5(a) was provided. To confirm that both random and tuned versions of our metaheuristic can find strong solutions with low optimality gap, Figure 6.5(b) was available. We can see that our metaheuristic algorithm has an optimality gap around 11% for solving very large-scale instances fast while the CPU time of the exact solver is very high and unreasonable. Therefore, we can say that the proposed algorithm was not very good at solving the proposed SSND. To analyze the robustness of our solutions, the best solution ever found by our metaheuristic, was studied in Figure 6.6. To study the accuracy of our metaheuristic in both versions, some statistical analyses using 95% confidence level for the normalized standard deviation of our solutions, the tuned version of our metaheuristic was more accurate than the one obtained by the random version as shown in Figure 6.7. Finally, the convergence analysis was done to see the behavior of the proposed algorithm in each test problem (Fig. 6.8).

CHAPTER 7 CONCLUSION AND RECOMMENDATIONS

This thesis developed an ALNS metaheuristic algorithm for a single-segment corridor network in an M1M system for planning at the tactical level. This system including a set of shippers and carriers was modeled to plan the shipments which must be picked up at the origin terminal and delivered at the destination terminal where IDSP was used to manage this M1M system.

The scope of this thesis refers to optimization for the freight transportation system where our problem was considered as a scheduled service network design model on a time-space network to be performed over the planning horizon. Our model formulated as a single-segment corridor network by selecting profitable requests from contract-based and non-contract requests, choosing a set of individuals of scheduled services, and identifying the itineraries of shipper-demand requests, and keeping them at the warehouse capacity of terminals.

This thesis employed the concept of revenue management for the single-segment corridor network where the main objective is to maximize the total profit for all accepted requests and the opportunity loss cost for unaccepted ones while considering the fixed costs, transportation costs and penalty costs if the schedule does not meet the time window.

The main challenge of this thesis is the solving of our single-segment corridor network in a reasonable time. The complexity results from time-space network and the number of shipments, i.e., contract-based, and non-contract requests in large-scale instances. Therefore, exact algorithms are not efficient to solve service network design problems where this thesis for the first time applies an ALNS metaheuristic combining SA and the local search for solving the mentioned problem.

This thesis focused on the development of a new metaheuristic that considers ALNS as the main loop while SA and local search methods are assigned as sub-loops. A general statement is that the proposed algorithm based on ALNS, and SA was acceptable in solving the single segment corridor network problem considering execution and ad-hoc arcs for solving very large-scale instances. It starts with an initial solution of a constructive heuristic algorithm in which a priority-based decision rule is designed. Then, the most important parts were the eight removal operators to destroy some selected services and accepted requests from the current solution as well as four insertion operators to repair it. Among all removal and insertion operators, the removal based on low-profit requests and maximum volume request insertion operators is the most efficient.

After studying the performance of our constructive heuristic algorithm and the components of our removal and insertion operators, we analyzed our metaheuristic algorithm in both versions of random-based and calibrated ones statistically in terms of computational time as well as optimality gap in comparison with the exact solver. The main finding is that the proposed algorithm finds solutions with the optimality gap which is around 11% for solving very large-scale instances. Most importantly, CPU time of our ALNS metaheuristic algorithm is less than one minute while the exact solver needs around one hour or more. Therefore, the proposed ALNS cannot find high-quality solutions. The solutions returned by ALNS display gaps higher than 10%, which is disappointing to be very efficient for solving our SSND. Generally, the exact method returns a better solution in comparison with the ALNS in all instances.

Although this thesis made a significant contribution to the development of a new solution approach based on ALNS, SA, and local search for solving a single-segment corridor network design problem, some limitations could be addressed in our future work. Firstly, the proposed algorithm may be improved by the adaptive memory search operator to change the removal or insertion operator in each iteration using a tabu list. Secondly, several prioritized time windows can be considered as constraints to increase the complexity of the proposed model. Finally, different characteristics of transportation services can be designed as a decision variable instead of an input parameter in the proposed model.

REFERENCES

[1] Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. In International conference on principles and practice of constraint programming (pp. 417-431). Springer, Berlin, Heidelberg.

[2] Schrimpf, G., Schneider, J., Stamm-Wilbrandt, H., & Dueck, G. (2000). Record breaking optimization results using the ruin and recreate principle. Journal of Computational Physics, 159(2), 139-171.

[3] Qian, W., Chai, J., Xu, Z., & Zhang, Z. (2018). Differential evolution algorithm with multiple mutation strategies based on roulette wheel selection. Applied Intelligence, 48(10), 3612-3629.

[4] Yang, J., & Soh, C. K. (1997). Structural optimization by genetic algorithms with tournament selection. Journal of computing in civil engineering, 11(3), 195-200.

[5] Zachariadis, E. E., & Kiranoudis, C. T. (2011). A local search metaheuristic algorithm for the vehicle routing problem with simultaneous pick-ups and deliveries. Expert Systems with Applications, 38(3), 2717-2726.

[6] Azi, N., Gendreau, M., & Potvin, J. Y. (2014). An adaptive large neighborhood search for a vehicle routing problem with multiple routes. Computers & Operations Research, 41, 167-173.

[7] Crainic, T. G. (2000). Service network design in freight transportation. European journal of operational research, 122(2), 272-288.

[8] Crainic, T. G., Fomeni, F. D., & Rei, W. (2021). Multi-period bin packing model and effective constructive heuristics for corridor-based logistics capacity planning. Computers & Operations Research, 132, 105308.

[9] Taguchi, G., & Jugulum, R. (2002). The Mahalanobis-Taguchi strategy: A pattern technology system. John Wiley & Sons.

[10] Taherkhani, G., Bilegan, I. C., Crainic, T. G., Gendreau, M., & Rei, W. (2022). Tactical capacity planning in an integrated multi-stakeholder freight transportation system. Omega, 110, 102628.

[11] Zhang, X., & Liu, X. (2022). A two-stage robust model for express service network design with surging demand. European Journal of Operational Research, 299(1), 154-167.

[12] Crainic, T. G. (2003). Long-haul freight transportation. In Handbook of transportation science (pp. 451-516). Springer, Boston, MA.

[13] Cordeau, J. F., Toth, P., & Vigo, D. (1998). A survey of optimization models for train routing and scheduling. Transportation science, 32(4), 380-404.

[14] Chouman, M., & Crainic, T. G. (2021). Freight Railroad Service Network Design. In Network Design with Applications to Transportation and Logistics (pp. 384-426). Springer, Cham.

[15] Christiansen, M., Fagerholt, K., Nygreen, B., & Ronen, D. (2007). Maritime transportation. Handbooks in operations research and management science, 14, 189-284.

[16] Christiansen, M., Hellsten, E., Pisinger, D., Sacramento, D., & Vilhelmsen, C. (2020). Liner shipping network design. European Journal of Operational Research, 286(1), 1-20.

[17] Bakir, I., Erera, A., & Savelsbergh, M. (2021). Motor Carrier Service Network Design. In Network Design with Applications to Transportation and Logistics (pp. 427-467). Springer, Cham.

[18] Crainic, T. G., & Kim, K. H. (2007). Intermodal transportation. Handbooks in operations research and management science, 14, 467-537.

[19] SteadieSeifi, M., Dellaert, N. P., Nuijten, W., Van Woensel, T., & Raoufi, R. (2014). Multimodal freight transportation planning: A literature review. European journal of operational research, 233(1), 1-15.

[20] Crevier, B., Cordeau, J. F., & Savard, G. (2012). Integrated operations planning and revenue management for rail freight transportation. Transportation Research Part B: Methodological, 46(1), 100-119.

[21] Wang, Y., Bilegan, I. C., Crainic, T. G., & Artiba, A. (2014). Performance indicators for planning intermodal barge transportation systems. Transportation Research Procedia, 3, 621-630.

[22] Bilegan, I. C., Brotcorne, L., Feillet, D., & Hayel, Y. (2015). Revenue management for rail container transportation. EURO Journal on Transportation and Logistics, 4(2), 261-283.

[23] Wang, Y., Bilegan, I. C., Crainic, T. G., & Artiba, A. (2016, September). A revenue management approach for network capacity allocation of an intermodal barge

transportation system. In International Conference on Computational Logistics (pp. 244-257). Springer, Cham.

[24] Van Riessen, B., Negenborn, R. R., & Dekker, R. (2017). The Cargo Fare Class Mix problem for an intermodal corridor: revenue management in synchromodal container transportation. Flexible Services and Manufacturing Journal, 29(3), 634-658.

[25] Bilegan, I. C., Crainic, T. G., & Wang, Y. (2022). Scheduled service network design with revenue management considerations and an intermodal barge transportation illustration. European Journal of Operational Research, 300(1), 164-177.

[26] Marasco, A. (2008). Third-party logistics: A literature review. International Journal of production economics, 113(1), 127-147.

[27] Jayaram, J., & Tan, K. C. (2010). Supply chain integration with third-party logistics providers. International Journal of production economics, 125(2), 262-271.

[28] Giri, B. C., & Sarker, B. R. (2017). Improving performance by coordinating a supply chain with third party logistics outsourcing under production disruption. Computers & Industrial Engineering, 103, 168-177.

[29] Lieb, R., & Bentz, B. A. (2005). The use of third-party logistics services by large American manufacturers: the 2004 survey. Transportation journal, 44(2), 5-15.

[30] Aguezzoul, A. (2014). Third-party logistics selection problem: A literature review on criteria and methods. Omega, 49, 69-78.

[31] Dufour, É., Laporte, G., Paquette, J., & Rancourt, M. È. (2018). Logistics service network design for humanitarian response in East Africa. Omega, 74, 1-14.

[32] Giusti, R., Manerba, D., Bruno, G., & Tadei, R. (2019). Synchromodal logistics: An overview of critical success factors, enabling technologies, and open research issues. Transportation Research Part E: Logistics and Transportation Review, 129, 92-110.

[33] Premkumar, P., Gopinath, S., & Mateen, A. (2021). Trends in third party logistics–the past, the present & the future. International Journal of Logistics Research and Applications, 24(6), 551-580.

[34] Chen, Z. S., Zhang, X., Govindan, K., Wang, X. J., & Chin, K. S. (2021). Third-party reverse logistics provider selection: A computational semantic analysis-based multi-

perspective multi-attribute decision-making approach. Expert Systems with Applications, 166, 114051.

[35] Benjelloun, A., & Crainic, T. G. (2008). Trends, challenges, and perspectives in city logistics. Transportation and land use interaction, proceedings TRANSLU, 8, 269-284.

[36] Hesse, M. (2002). City Logistics. Network Modelling and Intelligent Transport Systems. Journal of Transport Geography, 10, 158-159.

[37] Taniguchi, E. (2014). Concepts of city logistics for sustainable and liveable cities. Procedia-social and behavioral sciences, 151, 310-317.

[38] Bektas, T., Crainic, T. G., & Van Woensel, T. (2015). From managing urban freight to smart city logistics networks.

[39] Savelsbergh, M., & Van Woensel, T. (2016). 50th anniversary invited article—city logistics: Challenges and opportunities. Transportation Science, 50(2), 579-590.

[40] Holguín-Veras, J., Leal, J. A., Sánchez-Diaz, I., Browne, M., & Wojtowicz, J. (2020). State of the art and practice of urban freight management: Part I: Infrastructure, vehicle-related, and traffic operations. Transportation Research Part A: Policy and Practice, 137, 360-382.

[41] Holguín-Veras, J., Leal, J. A., Sanchez-Diaz, I., Browne, M., & Wojtowicz, J. (2020). State of the art and practice of urban freight management Part II: Financial approaches, logistics, and demand management. Transportation Research Part A: Policy and Practice, 137, 384-410.

[42] Crainic, T. G. (2008). City logistics. In State-of-the-art decision-making tools in the information-intensive age (pp. 181-212). INFORMS.

[43] Ghamlouche, I., Crainic, T. G., & Gendreau, M. (2003). Cycle-based neighbourhoods for fixed-charge capacitated multicommodity network design. Operations research, 51(4), 655-667.

[44] Ghamlouche, I., Crainic, T. G., & Gendreau, M. (2004). Path relinking, cycle-based neighbourhoods and capacitated multicommodity network design. Annals of Operations research, 131(1), 109-133.

[45] Crainic, T. G., Li, Y., & Toulouse, M. (2006). A first multi-level cooperative algorithm for capacitated multi-commodity network design. Computers & operations research, 33(9), 2602-2622.

[46] Crainic, T. G., & Gendreau, M. (2007). A scatter search heuristic for the fixed-charge capacitated network design problem. In Metaheuristics (pp. 25-40). Springer, Boston, MA.

[47] Paraskevopoulos, D. C., Bektaş, T., Crainic, T. G., & Potts, C. N. (2016). A cycle-based evolutionary algorithm for the fixed-charge capacitated multi-commodity network design problem. European Journal of Operational Research, 253(2), 265-279.

[48] Gendron, B., Hanafi, S., & Todosijević, R. (2018). Matheuristics based on iterative linear programming and slope scaling for multicommodity capacitated fixed charge network design. European Journal of Operational Research, 268(1), 70-81.

[49] Chouman, M., Crainic, T., & Gendron, B. (2018). The impact of filtering in a branch-and-cut algorithm for multicommodity capacitated fixed charge network design. EURO Journal on Computational Optimization, 6(2), 143-184.

[50] Sarayloo, F., Crainic, T. G., & Rei, W. (2021). A learning-based matheuristic for stochastic multicommodity network design. INFORMS Journal on Computing, 33(2), 643-656.

[51] Sarayloo, F., Crainic, T. G., & Rei, W. (2021). A reduced cost-based restriction and refinement matheuristic for stochastic network design problem. Journal of Heuristics, 27(3), 325-351.

[52] Kazemzadeh, M. R. A., Bektaş, T., Crainic, T. G., Frangioni, A., Gendron, B., & Gorgone, E. (2022). Node-based Lagrangian relaxations for multicommodity capacitated fixed-charge network design. Discrete Applied Mathematics, 308, 255-275.

[53] Agarwal, Y. K., Aneja, Y. P., & Jayaswal, S. (2022). Directed fixed charge multicommodity network design: A cutting plane approach using polar duality. European Journal of Operational Research, 299(1), 118-136.

[54] Pedersen, M. B., Crainic, T. G., & Madsen, O. B. (2009). Models and tabu search metaheuristics for service network design with asset-balance requirements. Transportation Science, 43(2), 158-177.

[55] Mara, S. T. W., Norcahyo, R., Jodiawan, P., Lusiantoro, L., & Rifai, A. P. (2022). A survey of adaptive large neighborhood search algorithms and applications. Computers & Operations Research, 105903.

[56] Dayarian, I., Crainic, T. G., Gendreau, M., & Rei, W. (2016). An adaptive large-neighborhood search heuristic for a multi-period vehicle routing problem. Transportation Research Part E: Logistics and Transportation Review, 95, 95-123.

[57] Kim, D., & Pardalos, P. M. (2000). Dynamic slope scaling and trust interval techniques for solving concave piecewise linear network flow problems. Networks: An International Journal, 35(3), 216-222.

[58] Mahey, P., & Souza, M. C. D. (2017). Multicommodity network flows with nonconvex arc costs. Pesquisa Operacional, 37, 571-595.

[59] Fortz, B., Gouveia, L., & Joyce-Moniz, M. (2017). Models for the piecewise linear unsplittable multicommodity flow problems. European Journal of Operational Research, 261(1), 30-42.

[60] Kim, D., Barnhart, C., Ware, K., & Reinhardt, G. (1999). Multimodal express package delivery: A service network design application. Transportation science, 33(4), 391-407.

[61] Armacost, A. P., Barnhart, C., & Ware, K. A. (2002). Composite variable formulations for express shipment service network design. Transportation science, 36(1), 1-20.

[62] Jarrah, A. I., Johnson, E., & Neubert, L. C. (2009). Large-scale, less-than-truckload service network design. Operations Research, 57(3), 609-625.

[63] Erera, A., Hewitt, M., Savelsbergh, M., & Zhang, Y. (2013). Improved load plan design through integer programming based local search. Transportation Science, 47(3), 412-427.

[64] Belieres, S., Hewitt, M., Jozefowiez, N., Semet, F., & Van Woensel, T. (2020). A Benders decomposition-based approach for logistics service network design. European Journal of Operational Research, 286(2), 523-537.