



Faculty of Computer Science and Information Technology

**Variable Neighbourhood Search Algorithm for Vehicle Routing Problem
with Backhaul**

Doreen Sek Siaw Ying

**Master of Science
2023**

Variable Neighbourhood Search Algorithm for Vehicle Routing Problem with Backhaul

Doreen Sek Siaw Ying

A thesis submitted

In fulfillment of the requirements for the degree of Master of Science

(Computational Mathematics)

Faculty of Computer Science and Information Technology

UNIVERSITI MALAYSIA SARAWAK

2023

DECLARATION

I declare that the work in this thesis was carried out in accordance with the regulations of Universiti Malaysia Sarawak. Except where due acknowledgements have been made, the work is that of the author alone. The thesis has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.



.....
Signature

Name: Doreen Sek Siaw Ying

Matric No.: 16020135

Faculty of Computer Science and Information Technology

Universiti Malaysia Sarawak

Date : April 2023

ACKNOWLEDGEMENT

First of all, I would like to express my profound gratitude and great appreciation to my supervisor, Associate Professor Dr. Sze San Nah for her valuable and constructive guidance and suggestions throughout this research project. A great appreciation for her patience, motivation, and willingness to share her knowledge throughout this research project which impressively assisted in this research project. Besides that, many thanks to my co-supervisors, Associate Professor Dr. Chiew Kang Leng and Dr. Sze Jeeu Fong for their encouragement and valuable time throughout the research planning and development process.

Apart from that, I also appreciate Faculty of Computer Science and Information Technology, Universiti Malaysia Sarawak for providing a comfortable learning environment and facilities. Moreover, I would like to thanks to all my friends for lending their precious time in giving their views, advices, ideas and supports throughout the completion of this research.

Last but not least, thanks to my family for their unconditional love, words of encouragement and the strength they gave me. This project would be impossible to be completed without them.

ABSTRACT

This research focuses on the Vehicle Routing Problem with Backhaul (VRPB). VRPB is one of the extended problems related to the usual Vehicle Routing Problem (VRP). VRPB consists of both linehaul customers and backhaul customers with known demand. There is only a single depot that can receive and supply the loads. The vehicles can only visit each customer once and serve all customers simultaneously by delivering or picking up the loads within a limited capacity. VRPB is a Non-deterministic Polynomial-time hard (NP-hard) problem. The huge data size has increased the difficulty of solving it using mathematical programming or combinatorial optimisation. Past literature showed that heuristic-based solutions could offer feasible solutions that are approximately accurate to the exact solution. It is one of the most popular solutions to solve a complex problem. Thus, a heuristic approach based on the Variable Neighbourhood Search (VNS) is proposed. In this research, 22 sets of benchmark instances introduced by Goetschalckx and Jacobs-Blecha are used to test the efficiency and effectiveness of the proposed algorithm. The proposed algorithm solution consists of two main phases. A simple priority heuristic rule is developed in the first phase to generate the initial dataset solution for each benchmark instance. The VNS algorithm is used to improve the obtained solutions in the improvement phase. A set of local search approaches and random shaking methods are proposed to conduct a list of neighbourhood solutions. Then, the most optimum solution among the neighbourhood solution in the improvement phase is selected as the final outcome of this research. The result is then compared with the best-known solution that can be found in past literature research. The relative percentage deviation shows a total of 14 out of 22 sets of datasets can achieve the same result as the best-known solution. The computational result shows that the proposed

heuristic algorithm is favourable in solving VRPB, and the generated solution is able to comply with VRPB constraints.

Keywords: Vehicle routing problem with backhaul, linehaul and backhaul customers, benchmark dataset, heuristic algorithm, variable neighbourhood search

Masalah Laluan Kenderaan Ambilan Balik dengan Cara Carian Kejiranan Berubah

ABSTRAK

Penyelidikan ini fokus kepada masalah laluan kenderaan ambilan balik (MLKB). MLKB adalah salah satu masalah lanjutan yang berkaitan dengan masalah laluan kenderaan. MLKB terdiri daripada pelanggan menerima dan pelanggan ambilan dengan permintaan yang diketahui. MLKB terdapat satu depoh yang boleh menerima dan membekalkan muatan. Kenderaan boleh melawat dan melayani setiap pelanggan dan memberikan khidmat kepada semua pelanggan secara serentak untuk menghantar atau mengambil muatan dalam kapasiti terhad. MLKB adalah masalah masa polinomial yang sukar dideterministik. Saiz data yang besar telah meningkatkan kesukaran untuk menyelesaikannya dengan menggunakan pengaturcaraan matematik atau cara pengoptimuman gabungan. Pengajian lepas telah menunjukkan bahawa penyelesaian berasaskan heuristik boleh menawarkan penyelesaian yang dekat dengan penyelesaian yang tepat. Ia adalah salah satu penyelesaian yang paling popular untuk menyelesaikan masalah ini. Oleh itu, pendekatan heuristik berdasarkan carian kejiranan berubah (CKB) telah dicadangkan. Dalam penyelidikan ini, 22 kombinasi contoh penanda aras yang diperkenalkan oleh Goetschalckx dan Jacobs-Blecha akan diguna untuk menguji kecekapan dan keberkesanan algoritma yang dicadangkan. Algoritma penyelesaian yang dicadangkan terdiri daripada dua fasa utama. Dalam fasa pertama, peraturan heuristik mudah dibangunkan untuk menjana penyelesaian bagi setiap kombinasi data. Algoritma CKB telah digunakan dekat fasa penambahbaikan untuk menambah baik penyelesaian. Senarai kaedah pendekatan carian tempatan dan kaedah goncangan rawak telah dicadangkan untuk menjalankan senarai penyelesaian kejiranan. Kemudian, penyelesaian yang paling baik antara penyelesaian kejiranan dalam fasa penambahbaikan akan dipilih sebagai hasil akhir dalam penyelidikan ini. Hasilnya

telah dibandingkan dengan penyelesaian yang paling terkenal yang didapati dalam penyelidikan literatur yang lalu. Sisihan peratusan relatif menunjukkan sejumlah 14 daripada 22 kombinasi data dapat mencapai hasil yang sama seperti penyelesaian yang paling terkenal. Keputusan tersebut menunjukkan bahawa algoritma heuristik yang dicadangkan adalah sesuai dalam menyelesaikan MKLB dan mampu mematuhi kekangan MKLB.

Kata kunci: Masalah laluan kenderaan ambilan balik, pelanggan penghantaran dan pelanggan menerima, kaedah heuristik, carian kejitiranan berubah

TABLE OF CONTENTS

	Page
DECLARATION	i
ACKNOWLEDGEMENT	ii
ABSTRACT	iii
ABSTRAK	v
TABLE OF CONTENTS	vii
LIST OF TABLES	xi
LIST OF FIGURES	xii
CHAPTER 1 INTRODUCTION	1
1.1 Background	1
1.2 Problem Statement	4
1.3 Research Objectives	6
1.4 Research Scope	6
1.5 Thesis Structure	6
CHAPTER 2 LITERATURE REVIEW	8
2.1 Overview of Vehicle Routing Problem	8
2.2 Vehicle Routing Problem with Backhaul	10
2.3 Existing Methods	12

2.3.1	Overview for Exact Solution	13
2.3.2	Heuristic and Metaheuristic Algorithm	14
2.3.2.1	Insertion Heuristic Algorithm	15
2.3.2.2	Tabu Search Algorithm	16
2.3.2.3	Ant Colony Algorithm	17
2.3.2.4	Variable Neighbourhood Search	18
2.4	Conclusion	20
CHAPTER 3 PROBLEM FORMULATION		22
3.1	Introduction	22
3.2	Problem Statement	22
3.2.1	Customers	23
3.2.2	Depot	24
3.2.3	Vehicles	25
3.3	Constraints	25
3.4	Objective Function	26
3.5	Mathematical Model	26
3.5.1	Notation	26
3.6	Conclusion	28

CHAPTER 4 VEHICLE ROUTING PROBLEM WITH BACKHAUL	30
4.1 Introduction	30
4.2 Benchmark Data	31
4.3 Data Pre-processing	33
4.3.1 Data Distance Calculation	34
4.4 Initial Solution	36
4.5 Variable Neighbourhood Search Algorithm	44
4.5.1 Local Search Techniques	49
4.5.1.1 Local 1-to-0 Swap	49
4.5.1.2 Local 2-to-1 Swap	51
4.5.1.3 Local 1-to-1 Swap	52
4.5.2 Shaking Methods	54
4.5.2.1 Random Multiple 1-to-0 Swap	54
4.5.2.2 Random Pick Solution	55
4.6 Result and Analysis	56
4.6.1 Computational Time	57
4.6.2 Computational Result	62
4.6.2.1 Analysis of the Final Solution	63
4.6.2.2 Analysis of Final Solution with Literature's Best-Known Solution	67
4.7 Conclusion	71

CHAPTER 5 MIXED VEHICLE ROUTING PROBLEM WITH BACKHAUL	72
5.1 Introduction	72
5.2 Model of m-VRPB	72
5.2.1 Initial Solution for m-VRPB	73
5.2.2 VNS for m-VRPB	79
5.2.2.1 Local Search Techniques for m-VRPB	80
5.2.2.2 Shaking Methods for m-VRPB	82
5.3 Result and Analysis for m-VRPB	83
5.3.1 Relaxed in Constraints	85
5.3.2 Minimise the Number of Vehicles	87
5.4 Conclusion	91
CHAPTER 6 CONCLUSION	93
6.1 Introduction	93
6.2 Conclusion of research	93
6.3 Research Contribution	94
6.4 Future Works	96
REFERENCES	97
APPENDICES	110
Appendix A Journal Publication	110
Appendix B Program Code	111
Appendix C Data output	117

LIST OF TABLES

	Page
Table 4.1: Customer size of benchmark datasets	32
Table 4.2: Lists of benchmark datasets	32
Table 4.3: Differences between local search and random shaking method	56
Table 4.4: Lists of compute time of VRPB	57
Table 4.5: Characteristics of each sub-problem sets (customer ratio)	62
Table 4.6: Initial and optimum solutions obtained for VRPB	64
Table 4.7: Summary of journals for Best-Known solution	68
Table 4.8: Lists of Best-Known result and final obtained result for VRPB	69
Table 5.1: Lists of constraints of VRPB and m-VRPB	72
Table 5.2: Benchmark dataset of m-VRPB	84
Table 5.3: Lists of Best-Known solution and m-VRPB solution	85
Table 5.4: Lists of Best-Known and m-VRPB (unfixed vehicle number) solution	89

LIST OF FIGURES

	Page
Figure 1.1: VRP diagram	4
Figure 1.2: VRPB diagram	5
Figure 3.1: Transportation network of VRPB	24
Figure 4.1: Distance matrix	34
Figure 4.2: Data of Problem Set A	35
Figure 4.3: Distance matrix of Problem Set A	36
Figure 4.4: Sorting structure for each route in VRPB	37
Figure 4.5: Vehicle travelling movement of VRPB	38
Figure 4.6: Node assignation Process (second node)	39
Figure 4.7: Node assignation (shortest travelled distance)	40
Figure 4.8: Data flow diagram of VRPB linehaul customer	42
Figure 4.9: Data flow diagram of VRPB backhaul customer	43
Figure 4.10: Initial solution for Dataset A1	44
Figure 4.11: Algorithm structure of VNS	45
Figure 4.12: Local search algorithm of VNS	47
Figure 4.13: Data flow diagram of VNS	48
Figure 4.14: Local 1-to-0 Swap	49
Figure 4.15: Lists of neighbourhood structure of node c_1	50
Figure 4.16: Process flow when better feasible solution has found	51
Figure 4.17: Local 2-to-1 Swap	52
Figure 4.18: Local 1-to-1 Swap	53
Figure 4.19: Lists of outcome for Dataset A1 by Local 1-to-1 Swap	53
Figure 4.20: Random Multiple 1-to-0 Swap	54

Figure 4.21: Outcomes of 3 times Random Multiple 1-to-0 Swap	55
Figure 4.22: Time taken of initial solution	59
Figure 4.23: Time taken of VNS	60
Figure 4.24: Total compute time of VRPB	61
Figure 4.25: Total travelled distance of initial and optimum solution	65
Figure 4.26: Total travelled distance of Best-Known and final obtained solutions	70
Figure 5.1: Route structure of m-VRPB	74
Figure 5.2: Vehicle travelling movement of m-VRPB	75
Figure 5.3: Sorting process of m-VRPB	76
Figure 5.4: Data flow diagram of initial solution of m-VRPB	78
Figure 5.5: Initial solution for Dataset A1 of m-VRPB	79
Figure 5.6: Local 1-to-1 Swap of m-VRPB	80
Figure 5.7: Local 2-to-1 Swap of m-VRPB	81
Figure 5.8: Local 1-to-Many Swap of m-VRPB	82
Figure 5.9: Random 1-to-0 Swap of m-VRPB	82
Figure 5.10: Random 2-to-2 Swap of m-VRPB	83
Figure 5.11: Random 1-to-1 Swap of m-VRPB	83
Figure 5.12: Total travelled distance of Best-Known and m-VRPB solution	86
Figure 5.13: Data flow diagram of initial solution (unfixed vehicle number)	88
Figure 5.14: Initial solution of Dataset A1 in m-VRPB (unfixed vehicle number)	89
Figure 5.15: Number of vehicles of Best-Known and final research outcomes	90
Figure 5.16: Total travelled distance of Best-Known, m-VRPB and m-VRPB (unfixed vehicle number)	91

CHAPTER 1

INTRODUCTION

1.1 Background

Logistic distribution is considered one of the core parts of our daily economy, production and transportation process. It plays an important role in global economic integration and information exchange operation. The logistic distribution has contributed a large portion of the total logistic costs (Crainic & Laporte, 1997). Among them, 85% of the logistic cost are transportation and stock costs (Zhao & Tang, 2009).

Logistic management is important not only at the firm level but also in the industry and globally economic competitiveness (Feng et al., 2012; Hwang et al., 2017). The study by Pishvae and Basiri (2009) stated that logistics costs had compromised a large ratio of the total gross domestic product (GDP). Besides, Wantanakomol (2021) also mentioned that the global average ratio of logistic cost to GDP is 10.7%. Therefore, well-planned and cost-effective logistic management helps reduce costs and increases profitability in economic activities.

After the outbreak of the COVID-19 pandemic, people are opting for contactless services such as e-commerce, online shopping, online food ordering and goods delivery services. The pandemic has caused great competition between companies all over the world in order to survive.

The Minister in the Prime Minister's Department (Economy) Datuk Seri Mustapa Mohamed has stated that the gross merchandise value (GMV) of Malaysia in the e-commerce and food delivery service showed a growth of RM29.28 billion in the year 2021

(“Tok Pa: E-commerce, Food Delivery Service Gross”,2022). In addition, food delivery services have expanded by 35% during that pandemic period. People are willing to pay for the online and delivery service to minimise their exposure to the virus and avoid the crowd and queues. As a result, the e-commerce and food delivery businesses such as, Grabfood, Foodpanda, fast food delivery services, Shopee, Lazada and Taobao were booming and accelerating globally.

Besides, this new normal has caused a rise in carriage volume and increased the demand for quality logistic service. Logistic service is said to be involved in the material transportation process, resources movement and goods distribution process (Laporte, 1992). Good transportation or logistic management can result in advantages over competitors and increase customer satisfaction (Sakhapov et al., 2018; Wantanacomol, 2021).

Based on the China Daily, the business volume of the express delivery companies in China has increased by over 70% with an increment of revenue of over 75% year on year in February 2022 (Yang, 2022). Moreover, Ninja Van, one of Malaysia’s courier and delivery services, has achieved 100 million parcel delivery as of July 2021 (Aufa, 2022). This parcel volume has shown the economic trends and the growth in e-commerce in Malaysia. According to the marketing forecast of Statista, the profit in the Platform-to-Consumer Delivery section is expected to show an annual growth rate of 7.78% with a market value of US\$302.50 billion by 2027 (Statista, 2022). These statistics have reflected the consumption growth in the surging delivery business.

However, Los Angeles Times has stated that the demand for grocery delivery has dropped 10.5% compared to its highest during the pandemic when the prices for the delivery service rose (Dee-Ann, 2022). One of the factors is that online prices are higher compared

to in-store prices, which include the extra charge to cover labour and transportation costs. This issue has raised pressure on logistics companies to provide better services at a much lower cost. The same tendency has happened in Malaysia; according to Soh Thian Lai, the president of the Federation of Malaysian Manufacturers (FMM), the price of daily goods is estimated to increase by 10% due to the increase in logistics costs, and the labour shortage (Reshna, 2022). Based on the survey conducted by FMM, most companies are expected to pass the increase in costs of between 1 to 10% to consumers. The increase in the prices would cause a multiplier impact on the economy unless proactive action was taken to control the rise in prices of the products.

Furthermore, according to Anbuudayasankar (2012), inefficient delivery management will cause a serious decline in service quality, a decrease in order volume and a loss of the loyalty of customers. Thus, additional attention needs to drag to the improvement of the logistic planning as well as the efficiency and service quality of the service (Yang, 2022).

In short, a well-planned logistic route can produce a cost-effective flow for the goods supply chain and a great customer experience. It is the main connecting link between the consumers and producers in the whole industry supply chain (Zhang et al., 2022) and the business network that joins the product producers and service providers associated globally (Farahani, 2011). Therefore, logistic distribution issues have been generalised to the linear optimisation issues and then transformed and modelled to the Vehicle Routing Problem (VRP) (Clarke & Wright, 1964). The main objective of VRP is to generate the optimal solution to serve a group of customers with a list of homogeneous vehicles under certain

constraints (Raff, 1983; Laporte & Nobert, 1987; Laporte, 1992). Therefore, VRP has become one of the popular topics among operation researchers in the past decade.

1.2 Problem Statement

In general, VRP is the flow of optimal routing routes between the initial point and a group of destination points under certain conditions. In a basic VRP, each customer can only be visited once and served by one vehicle. A vehicle should depart from an initial point, either a depot or warehouse and return to the same point after completing the tasks. Figure 1.1 represents a basic VRP routing diagram which involves a depot, a group of customers and the movement of a fleet of vehicles.

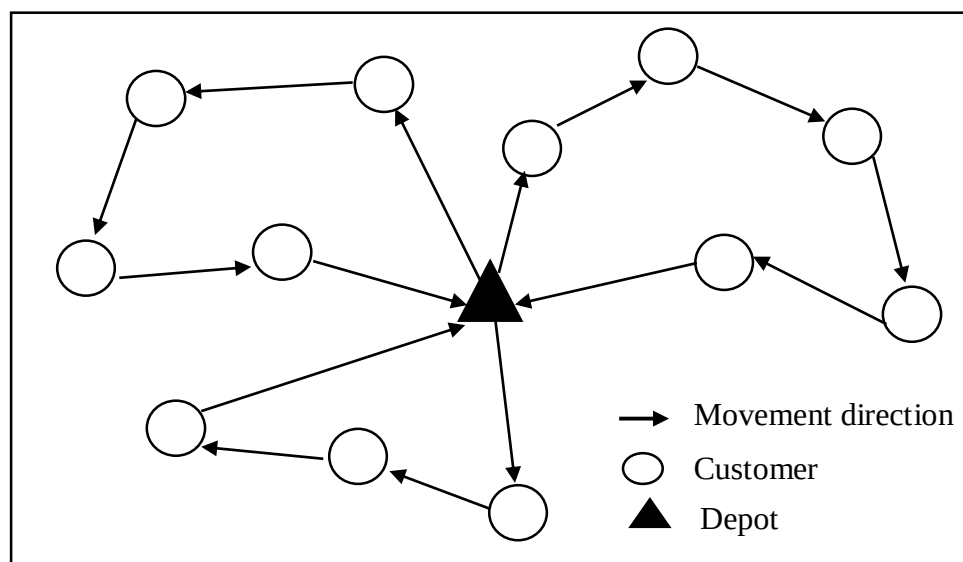


Figure 1.1: VRP diagram

In the past few years, VRP has widely extended with various constraints. Vehicle Routing Problem with Backhaul (VRPB) is one of the extension problems from the classical VRP (Goetschalckx & Jacobs-Blecha, 1989; Cuervo et al., 2014; Toth & Vigo, 1996; Nagy & Salhi, 2005).

VRPB consists of a group of customers, a depot and a fleet of homogeneous vehicles.

Unlike the basic VRP, VRPB consists of both delivery customers (linehaul customers) and pickup customers (backhaul customers). Delivery customers are the customers who require a delivery service, and the pickup customers are those who request a pickup service. The geometric location and the demand of each customer are different from each other. The vehicles need to move around the customers to either supply and collect sufficient loads for or from them.

In VRPB, a depot is a place that provides and stores stocks or goods. It can always supply a sufficient amount of the load to the delivery customers. Same as the basic VRP, in VRPB, the vehicle needs to depart and return to the depot in every logistic route. The vehicle will carry a certain amount of goods, depart from a depot, and distribute it to the delivery customers. It starts to approach the pickup customer along the way back to the depot after delivering all the goods to the customer. A feasible routing solution for a VRPB would be a list of transportation routes that can serve all the customer requirements under a fixed number fleet without violating the predefined capacity limit. Figure 1.2 below represents the vehicle movement in VRPB.

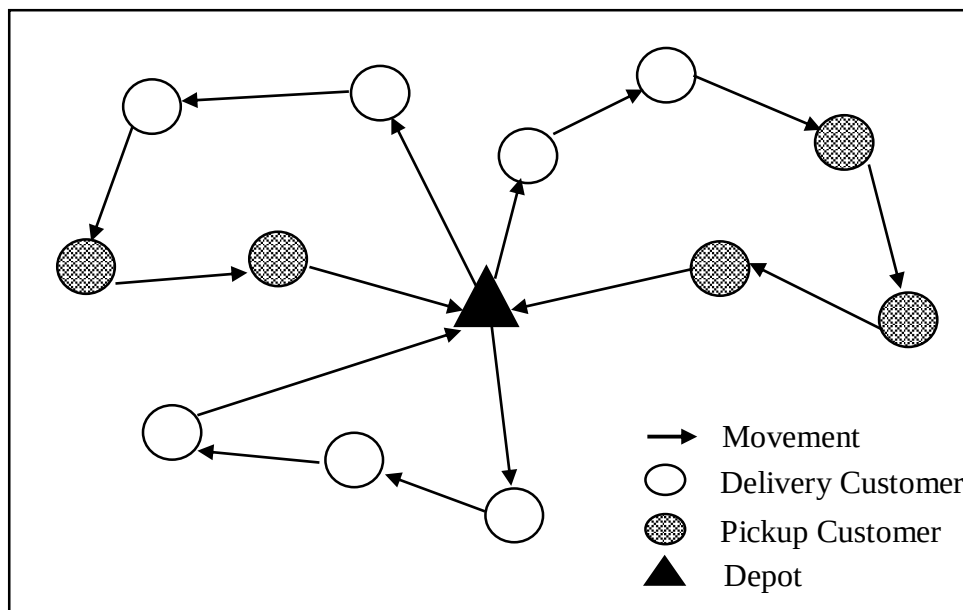


Figure 1.2: VRPB diagram

1.3 Research Objectives

The objectives of this research:

- i. To model a VRPB with extra operational constraints consideration.
- ii. To design and develop an effective Variable Neighbourhood Search (VNS) algorithm (Refer to Section 2.3.2.4) for the VRPB.
- iii. To conduct an extensive computational study and analysis using the benchmark data proposed by Goetschalckx and Jacobs-Blecha (1989).

1.4 Research Scope

The scope of this research is defined as follow:

- i. Focus on the metaheuristic Variable Neighbourhood Search (VNS) algorithm especially in algorithm development, to solve VRPB.
- ii. Conduct the computational and comparison study based on the 22 sets of selected benchmark data with the size from 25 to 113 customers proposed by Goetschalckx and Jacobs-Blecha (1989).

1.5 Thesis Structure

In Chapter 2, literature reviews on the overview of VRP and VRPB are studied and elaborated on in detail. Besides, the popular methodology approaches that have been used to solve VRPB-related issues in the past decade were presented in this chapter.

Chapter 3 presents the background and the constraint of the case study of this research. It included a detailed explanation of the customer, depot, vehicle, and problem

constraint of VRPB. A clear VRPB background condition and restriction are explained in this chapter.

In Chapter 4, the methodology to solve the VRPB of this research is discussed. The methodology included the data pre-processing steps used to prepare the input data, a simple priority rules heuristic algorithm used to generate the initial solution and the VNS heuristic algorithm used to produce an optimum solution. The proposed VNS algorithm with a set of local search techniques and random shaking methods is described in detail. The data analysis and result of the proposed algorithm on VRPB are discussed in this chapter.

In Chapter 5, a mixed Vehicle Routing Problem with Backhaul (m-VRPB) is proposed to examine the impact of the priority of the type of customer on the performance of the proposed algorithm. Note that m-VRPB is investigated using the same benchmark data but slight differences in the problem restriction. Two constraints are relaxed to simulate the real practical environment by removing some of the restrictions from the VRPB. The proposed VNS algorithm is transformed to cope with the new constraints. The new result obtained by m-VRPB is analysed and discussed in this chapter.

Lastly, Chapter 6 contains the conclusion remarks, the contribution of the project and some ideas for future works.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview of Vehicle Routing Problem

Vehicle Routing Problem (VRP) is a significant and widely studied logistics problem with a wide range of applicability. VRP is described as obtaining the optimal solution for transportation movement, which involves both delivery and collection processes from one or several depots to multiple geographical customer points under certain constraints (Raff, 1983; Laporte & Nobert, 1987; Laporte, 1992). It aims to develop suitable routes for a fleet of homogeneous vehicles that can serve a group of customers that start and end at a depot by minimising the travelling distance and transportation cost.

The first logistic issue, the “Truck Dispatching Problem”, was introduced by Dantzig and Ramser in 1959. Modelling of how a fleet of homogeneous gasoline delivery trucks travelled between a list of terminals with various supplier points has been conducted. It is targeted to obtain an optimum routing solution with minimum travel distance. However, the problem has been generalised to the linear optimisation issues commonly occurring in logistics and transport. Thus, the VRP was introduced (Clarke & Wright, 1964). Clarke and Wright (1964) were the first to have multiple corporate vehicles in VRP and also the first to introduce a new variant of VRP. It has begun to be applied in solving various kinds of real-life logistics problems.

Moreover, a study has started applying or combining VRP with different constraints and parameters in order to incorporate real-life situations. Researchers have simulated the transportation problem by varying the operational conditions, such as consideration of

different circumstances, different modes of serving, and various vehicle conditions (Clarke & Wright, 1964; Toth & Vigo, 1996; Nagy & Salhi, 2005; Braekers et al., 2016). Routing issues such as Capacitated Vehicle Routing Problem (CVRP) (Uchoa et al., 2017), Vehicle Routing Problem with Time Window (VRPTW) (Errico et al., 2016), Vehicle Routing Problem with Backhauls (VRPB) (Pinto et al., 2017; Wassan et al., 2017), Vehicle Routing Problem with Simultaneous Delivery and Pickup (VRPSDP) (Bianchessi & Righini, 2007; Wang et al., 2016), Green Vehicle Routing Problem (GVRP) (Koç & Karaoglan, 2016; Xiao & Konak, 2017), Open Vehicle Routing Problem (OVRP) (Lalla-Ruiz et al., 2016) and Split Delivery Vehicle Routing Problem (SDVRP) (Chen et al., 2017) are some of the common research topics of VRP that conducted in the past few years.

Furthermore, VRP software and applications development showed rapid growth in the industry and logistics fields. More and more companies have started to implement this technology into the logistics management process, for example, the support system used by an oil downstream company (Gayialis & Tatsiopoulos, 2004) and the scheduling model used by a sales company for flight tickets to arranging their free pickup and delivery services (Dong et al., 2011).

The study of Xiao and Konak (2017) has undertaken that an optimised routing solution is able to reduce transportation costs and the number of resources required. Besides that, good routing management can cut down fuel consumption (Norouzi et al., 2017) and reduce greenhouse gas emissions, such as carbon dioxide, especially in urban areas (Ehmke et al., 2016; Tian et al., 2018).

In short, a well-planned logistic route can produce a cost-effective flow for the goods supply chain, a great customer experience and a better optimisation plan within the limited

resources. Therefore, different methodologies and techniques were investigated to solve the VRP problem.

2.2 Vehicle Routing Problem with Backhaul

According to the literature study, VRP has been studied extensively by researchers worldwide using different approaches. Analysts and researchers are attracted to the complexity and the characteristic of VRP (Raff, 1983). Thus, VRP has become one of the popular operation research topics among the combinatorial optimization problem in logistic applications.

Among them, the logistics problem involving pickup and delivery service has received considerable attention worldwide. The study has proven that empty trips contribute a large part to the total travelled distance and to total logistic costs (Ubeda et al., 2011, Santos et. al., 2021). To avoid it, the backhauling strategy is implemented to reduce the empty trips by utilising the capacity of a vehicle to pick up some load on the return trips. Hence, the VRPB was introduced by Deif and Bodin (1984). Besides, VRPB has been successfully applied in real-life logistic problems (Yano et al., 1987; Koç & Laporte, 2018; Póvoa, 2020).

VRPB consists of a set of customers with two different characteristics. In VRPB, the customer is separated into linehaul and backhaul customers. According to Pinto et al. (2017), linehaul customers can be categorised as the source of demand, while backhaul customers are the suppliers in daily life. This kind of customer splitting pattern is common in the distribution problem (Koç & Laporte, 2018). Furthermore, it has happened extremely frequently in daily life and real-world scenarios (Crispim & Brandao, 2001).

Some common examples are door-to-door delivery service or dial-a-ride service in the grocery service, freight shipping industry and the supplier chain of industrial areas (Wilson et al., 1976; Desaulniers et al., 2002). For example, in the grocery industry, retail shops and supermarkets are the linehaul customer, whereas the manufacturing supplier is the backhaul customer (Pinto et al., 2020).

Although VRPB can serve linehaul and backhaul customers on the same route, it is practised to avoid a mix in pickup and delivery service. The constraint is set to avoid the re-handling and load rearrangement of the vehicle at the customer locations. Besides, in most practical applications, a vehicle is often rear-loaded after the delivery service due to the highest service priority of a linehaul customer (Toth & Vigo, 1999). An example of this is a drink distribution company delivering the full bottles to the customer before the process of collecting the empty bottle at the end of the route (Koç & Laporte, 2018). This also includes the example of a meat supplier which needs to supply the meat to the butcher before collecting the waste from the slaughterhouse due to hygiene reasons (Li et al., 2019).

Furthermore, due to the rapid growth of the logistic operation, effective transportation planning has been proven to reduce the cost of the logistic process (Yano et al., 1987; Santos et al., 2021). Based on Parragh et al. (2008), the cost of VRPB can be reduced by combining the linehaul and backhaul tours. This circumstance has proven valuable to research in the industry (Toth & Vigo, 1999).

Additionally, there are a lot of VRPB models and applications developed for real-life case studies. For example, the food or groceries seller delivers the food and goods to the customer and collects the goods from the supplier (Cheung & Hang, 2003; Ubeda et al., 2011). Anbuudayasankar et al. (2012) have developed a three-heuristic solution to optimise

the distribution process of a bank in the operation of collecting and reloading the money of the automated teller machines (ATM). Besides, Soleimani et al. (2018) have developed a model for Hamshahri, the Iranian newspaper. The model optimised the newspaper delivering and collecting process for the 36 newsstands located in District 11 and District 12 in the Tehran Metropolitan area.

Based on Povia (2020), backhauling is not well studied, yet it plays an important role in logistic optimisation. Besides, according to the research done by Braekers et al. (2016) of a total of 277 VRP articles published between the years 2009 to mid-year 2015, the total of studies related to VRP with backhaul only consists of 18.65% out of 100%. Hence, this project is focused on the VRPB.

2.3 Existing Methods

VRP has widely transformed into various extended logistic problems, as mentioned in Section 2.1. The increase in constraints and data size has boosted the difficulty of a VRP to be solved using mathematical programming or combinatorial optimisation method.

Based on the review of VRP conducted by Zhang et al. (2022), the simple heuristic method, such as the greedy algorithm and local improvement heuristic, were the popular solution from 1960 to 1970. Next, it was followed by mathematical planning-based heuristics such as assignment, set segmentation and coverage (Zhang et al., 2022). At present, various rigorous heuristic and artificial intelligence methods are implemented to solve VRP-related problems.

There are a huge number of methods and solutions that can be found in published academic literature. Different methodologies and techniques were investigated to solve the

VRP-related problem. The solution methods can be divided into exact algorithms, heuristics, and metaheuristics (Fisher, 1995; Koç & Laporte, 2018; Zhang et al., 2022). The common exact algorithms used to solve VRP include integer linear programming, precision algorithms and tree search method. In contrast, the popular heuristic and metaheuristics include the saving method, simulated or deterministic annealing method, tabu search method, genetic algorithm, ant colonization algorithm, neighbourhood method, etc.

2.3.1 Overview for Exact Solution

At the beginning stage, an exact algorithm is used to solve VRPB. In 1987, Laporte and Nobert (1987) classified the exact algorithm into three categories, thus, the direct tree search method, dynamic programming and integer linear programming. They divided the exact algorithm into set partitioning formulation, vehicle route formulation, and commodity flow formulation (Laporte & Nobert, 1987).

Among them, precision algorithms include branch and bound methods and branch-cutting methods. In this algorithm, every branch of the tree is represented as an arc in the solution. Besides, this method performed better with less constraint or a tight problem as the number of branches to be explored is reduced (Laporte, 1992). However, for dynamic programming, the feasible and dominance criterion is needed in order to reduce the number of feasible options.

The first exact branch and the bound algorithm used to solve VRPB are proposed by Toth and Vigo (1997). The proposed solution solved the instances of Goetschalckx and Jacobs-Blecha (1989) with 25 and 68 customers. They have computed an effective Lagrangian bound, generating a set of new instances between 21 to 100 customers (Toth & Vigo, 1997).

Mingozi et al. (1999) have proposed an exact method based on the partitioning formulation for VRPB. The feasible solution is obtained using linear programming relaxation and two different bounding procedures. The proposed algorithm has solved all instances published by Goetschalckx and Jacobs-Blecha (1989) and Toth and Vigo (1997).

Furthermore, the exact solution algorithms, such as additive bounding, branch and bound algorithm, branch and cut algorithm, branch, price algorithm, and set coverage methods, can be found in Barnhart et al. (1998), Carpaneto et al. (1989), Toth and Vigo (2002), Naddef and Rinaldi (2002), Fischetti and Toth (1989), Padberg and Rinaldi (1991) and Zhang et al. (2022).

2.3.2 Heuristic and Metaheuristic Algorithm

VRP is a Non-deterministic Polynomial-time hard (NP-hard) problem (Dantzig & Ramser, 1959; Garey & Johnson, 1979; Laporte & Nobert, 1987; Nagy & Salhi, 2005). Besides, VRP can be widely extended with various constraints and conditions, such as distance restriction, capacity or time window. The growth of the data size and constraints have increased the process jobs and time needed for the VRP to represent the full scale of feasible options (Raff, 1983; Laporte, 1992). Hence, solving the VRP-related problems to the optima is difficult using integer programming or an exact algorithm.

Since VRPB is extended from VRP, VRPB is also NP-complete. The huge data size of NP-hard problems has increased the difficulty of solving them using the exact solution. The size of VRP that can be solved using exact solutions may be limited (Nobert, 1987). From the literature study, the exact algorithms have been used to solve problems with small-scale data in a reasonable computing time. However, the number of computations grows exponentially with the size of the VRP problem. According to the study of Goetschalckx and

Jacobs-Blecha (1989), their exact algorithms can only resolve the VRP up to about 25 customers, which was a time-consuming process. Besides, Dell'Amico et al. (2006) developed the first branch and price approach with exact dynamic programming and some relaxation procedure. The proposed method can only solve instances of up to 40 customers with high compute time.

According to Kytöjoki et al. (2007), the heuristic methodology is able to yield a better and more appropriate solution, especially in the large data size and huge constraints issues. Moreover, Raff (1983) has stated that the heuristic methods can offer the approximate solution for the set covering and partitioning problem. Although it is not guaranteed to reach the exact solution, it can provide a feasible solution for a complex, large, and specific dependent problem under a reasonable computing time.

The analysis of Braekers et al. (2016) shows that between 2009 and 2015, the metaheuristic was the most common solution used, with an overall percentage of 71.25%. They stated that the exact methods and heuristic solution showed less portion compared to the metaheuristic, probably due to the high chances of getting stuck in an optimum local. Furthermore, from the study of Santos et al. (2020) throughout a total of 107 VRPB literature, the metaheuristic method has covered a large portion with a value of 58%, followed by the heuristic method gets up to 35%, and lastly, the exact method with only 14%. Thus, the metaheuristic is more suitable and powerful for complex and large data instances. Some of the popular heuristic and metaheuristics methods are discussed below.

2.3.2.1 Insertion Heuristic Algorithm

The first heuristic used to solve VRPB is proposed by Deif and Bodin (1984). They extended the concept of Clarke and Wright (1964), a single customer route is generated first,

then the remaining route is built by sequential insertion of customers using some consideration based on the cost-saving constraints (Wilson et al., 1976; Deif & Bodin, 1984). The proposed heuristic was tested on instances with up to 300 customers.

Based on Solomon (1987), the insertion method is known as the effective approximation method of VRP. Insertion procedure built the feasible configuration by inserting and comparison between a list of possibly infeasible configurations by some criterion function such as total cost, large saving or demanded entity (Raff, 1983).

In the year 1988, Casco et al. used this method in the delivery and pickup second theory. The cost is added to the load-based insertion procedure to get the shortest path solution. Moreover, the cheapest insertion heuristic is introduced, and the performance of both methods that were used are tested with various sizes of the simulated problem (Casco et al., 1988). Salhi and Nagy (1999) have proposed a new solution by extending the method published by Casco et al. (1988). They achieved slight improvement after applying the backhaul customer to the clustering method.

Besides, the insertion heuristic plays an important role in solving a VRP-related problem. The problem can be resolved in a short time by this method. Hence, it is usually used to obtain the feasible solution for an improvement method and as an advanced improvement procedure to obtain the optimal solution. It can be found in Rochat and Semet (1994), Lin and Chou (2012), Kuo et al. (2016) and Wang et al. (2020).

2.3.2.2 Tabu Search Algorithm

Tabu Search (TS) algorithm is a metaheuristic approach for VRP (Battiti & Tecchiolli, 1994; Barbarosoglu & Ozgur, 1999). It is able to solve VRP in a reasonable computing time with an acceptable degree of precision. The basic theory of TS is to get out

from the local optimal by providing a guiding strategy to explore the solution from the local search procedure. The strategy moves to the best solution in the neighbourhood even though the objective value is not the best. The acceptance solution is then stored in tabu list attributes (Barbarosoglu & Ozgur, 1999).

One of the first tabu search approaches used to solve VRPB is proposed by Osman and Wassan (2002) and called Reactive Tabu Search (RTS). The saving insertion and saving assignment methods are used as a two-route construction to generate the initial solution. RTS has provided a new diversity strategy from the original approach proposed by Battiti and Tecchiolli (1994). Besides, Wassan et al. (2008) have further improved RTS and applied it to solve the VRPB. Then, it was implemented by Nagy et al. (2013) to solve VRP with mixed pickup and delivery.

Furthermore, some studies have proved that the tabu search algorithm can yield a better solution when combined with other metaheuristic methods (Crispim & Brandão, 2005). In 2007, Wassan introduced a hybrid metaheuristic method which is the extended study of the RTS methodology by Osman and Wassan (2002). The output of the new methodology has shown effectiveness and efficiency in solving VRP-related problems.

2.3.2.3 Ant Colony Algorithm

The ant colony algorithm (ACO) is originally named Ant System (Dorigo et al., 1996; Wade & Salhi, 2003; Dorigo et al., 2006). ACO metaheuristics are then introduced by Dorigo et al. (1999). It is a simulated optimisation algorithm that simulates the biological finding on the real ant foraging behaviour and counterparts.

According to Dorigo et al. (2006), ant colony optimisation is an iterative algorithm. At each iteration, each of the artificial ants builds the route by working from vertex to vertex

stochastically with the constraint of not visiting any vertex that has been visited. At the end of each iteration, the pheromone value is changed to bias the ants in future iterations based on the quality of the constructed solution.

Wade and Salhi (2003) implemented some enhancements to the general characteristics of the ant system. In this study, the construction method on the candidate list is guided by the visibility function's look-ahead scheme and some rules that deal with local and global structuring updates.

ACO is the second most popular methodology used to solve VRP. For example, Wassan et al. (2013), Mazzeo and Loiseau (2004), Balseiro et al. (2011) and Gao et al. (2016) are some studies that applied the ACO to solve the capacitated VRP, VRP with a time window, VRP with dynamic location, etc.

Wu et. al. (2016) proposed a Label-based Ant Colony System (LACS) algorithm and solved the China Post Office postal shuttle service problem with a total of 84 nodes located in Guangzhou. The result shows the improvement in cost savings, and the proposed algorithm is able to cope with different customer demands.

2.3.2.4 Variable Neighbourhood Search

The Variable Neighbourhood Search (VNS) is a metaheuristic method (Hansen et al., 2010). The VNS algorithm aims to improve the solution by systematically exploring several neighbourhood structures in a local search (Uchoa et al., 2017). It is a framework to yield a list of feasible solutions by a systematic change along the neighbourhood structure. The solutions are generated using some local search approach until a local optimal has been reached (Mladenović & Hansen, 1999; Hansen et al., 2008).

The basic model of VNS was first proposed by Mladenović and Hansen (1999). Unlike other metaheuristic methods such as tabu search and ant colony, VNS required only a simple or no parameter to extend its schemes to solve combinatorial and global optimization problems. Therefore, VNS is a simple and easy method to adapt different schemes, leading to more effective and sophisticated operations (Hansen et al., 2008). Besides, VNS does not follow a trajectory like other metaheuristics; it first explores the optimal solution from the nearest neighbourhoods and only jumps to a new structure if an improvement is made (Mladenović & Hansen, 1999).

VNS solves the standard VRP or practical applications of routing problems (Hansen et al., 2008). Ochi et al. (2001) solved the travelling consumer problem using parallel VNS methodology. Parallel VNS is proposed to solve the p-Median problem by parallelising the process of local search until the best solution is found (García-López et al., 2002). In parallel VNS, each neighbourhood structure is handled by a single processor; the process is only interrupted when an improved solution is found (Crainic et al., 2004).

Moreover, Pinto et al. (2020) have proposed three variants of the VNS algorithm with 10 neighbourhood structures to solve a VRP with mixed linehaul and backhaul customers with two-dimensional loading constraints. The algorithms have been tested by a list of benchmark instances with a max number of 225 customers. The result shows a significant improvement for the instances with large data sizes. Kalayci and Kaya (2016) have developed a hybrid metaheuristic algorithm based on VNS and ant colony search (ASC) algorithm to test the performance of combining two different algorithm schemes. The proposed method was tested by a set of benchmark instances with a customer size of 50 to

200. The computerised result showed that ACS could help in utilising the memory mechanism of VNS.

Variable Neighbourhood Decomposition Search (VNDS) is proposed by Hansen et al. (2001). In VNDS, the basic VNS is decomposed into two levels VNS scheme by adding a new parameter to represent the running time for the decomposed problem. Besides, Crispim and Brandao (2001) and Rousseau et al. (2002) applied VNDS in VRP with backhauls.

Moreover, Melechovsky et al. (2005) proposed a hybrid metaheuristic which merged the tabu search with VNS to resolve the location routing problem with nonlinear costs. Kytöjoki et al. (2007) have also proposed a VNS methodology for a very large-scale vehicle routing problem. The application of VNS for the open VRP can be found in the studies of Fleszar et al. (2009), VNS in solving VRP with divisible delivery and pickup can be found in Polat (2017) and VPR with time window can be found in the research of Bräysy (2003).

2.4 Conclusion

From the literature study, the popularity of the VRP in the logistic study has been proven by a huge amount of academic studies on it and the various number of extended problems under different constraints. It has been considered one of the core processes to optimise the logistic distribution process. It has contributed a huge portion to vehicle route planning and, in the meantime, to reduce logistic expenses.

However, among the extended logistic studies of VRP, the VRPB only received little attention compared to others. The statistics of Braekers et al. (2016) have shown that VRPB is not well studied but is efficient in reducing logistic costs. Therefore, this research is focused on VRPB. Based on the study, VRPB can reduce the empty route by utilising the

vehicle's capacity in every route using backhauling strategy. Besides, it consists of a group of linehaul and backhaul customers, and this kind of customer pattern is very common in a real-world scenario. Lastly, the literature study has shown that the VRPB model has been applied and successfully solved real-life problems. In short, the strength of VRPB in optimising the logistic route has been proven in past academic studies.

VRPB has been proven an NP-hard problem. As the extended problem from VRP, the VRPB model may need to cope with more extra constraints. These strategies have increased the complexity of VRPB to an optimum. From the literature study, most researchers tend to solve the NP-hard problem with a heuristics-based solution. A heuristic-based solution can easily cope with many constraints. Besides, the heuristic algorithm can handle the problem with large data sizes. It is able to offer feasible solutions which approximate exact solutions. Thus, in this research, a heuristic-based algorithm is used to solve VRPB.

Based on the studies from the present method, VNS is selected to be used in this research. VRPB has been considered a complex problem consisting of both line and backhaul customers with multiple vehicles and constraints. VNS is suitable to solve VRPB because it is easy and simple to cope with a complex problem under specific schemes or constraints. Moreover, VNS has the ability to solve problems with large data sizes. Additionally, VNS does not follow a trajectory like other metaheuristics. The principle of VNS is that a solution should not be limited by the first local findings. The solution will not be trapped in a local optimum and can search through the nearest neighbourhoods for a better solution. Furthermore, the proposed VRPB model is extended to achieve multi-objective to achieve minimum travelled distance and minimise the number of vehicles used.

CHAPTER 3

PROBLEM FORMULATION

3.1 Introduction

In this chapter, the Vehicle Routing Problem with Backhaul (VRPB) issues will be discussed and analysed. Besides, the problem formulation is presented. The objective function and the benchmark data used will be determined. Then, a sample model is simulated based on the problem and the parameters.

3.2 Problem Statement

VRPB is one of the variant problems from Vehicle Routing Problem (VRP), which involves the delivery and pickup service. VRPB aims to optimise cost-effective and efficient routing plans for both linehaul and backhaul customers. A linehaul customer needs the delivery service, whereas the backhaul customer seeks the pickup service. Every customer is assigned a known demand (either a delivery or pick-up demand), and each customer should be served exactly once by a vehicle.

In VRPB, only one type of vehicle will be taken into consideration. The size and the capacity of every vehicle are the same. The vehicles can only visit one customer at a time and serve the customer by either delivering or picking up the loads. The total load carried in the vehicle cannot exceed the total capacity. Besides, only a single depot is considered in VRPB. A depot is a warehouse used to receive and supply loads. The vehicle should always start and end the trip at the depot.

In this research, a list of benchmark data instances with different customer sizes, customer locations, service demand, vehicle numbers and vehicle capacity is used to

simulate the different situations of the real-life problem. The objective is to develop an efficient routing plan that can complete all jobs within the minimum distance travelled. The detail and information of each substance are described in the sub-section below.

3.2.1 Customers

VRPB consists of two types of customers, linehaul customers and backhaul customers. Each customer can either request pickup or delivery services, but not both. Based on Goetschalckx and Jacobs-Blecha (1989), linehaul customers are the individuals that will receive the goods from a depot, whereas backhaul customers are the pickup points that send the goods back to the depot. Besides, Pinto et al. (2017) have stated that linehaul customers can be categorised as the source of demand, while backhaul customers are denoted as the suppliers in our daily life.

Moreover, the number of demands to be delivered or picked up from each customer is fixed in advance. The required service of each customer should be served or completed in one-off as the total loads of a particular customer cannot be split in any movement. In VRPB, every customer can only be visited and served by only one vehicle.

The customers are located in a transportation network with different coordinates and positions. There are no movement restrictions among the customers. The vehicle can move freely between linehaul to linehaul customer, linehaul to backhaul customer or backhaul to backhaul customer. However, movement from backhaul customer to linehaul customer is not considered in the VRPB modal. The linehaul customers should be served before backhaul customers to avoid rearranging the load on the vehicle (Queiroga et al., 2020). Besides, this problem does not allow any route with only backhaul customers. However, it allows serving only linehaul customers in a single route (Goetschalckx & Jacobs-Blecha, 1989). This

constraint is made under the assumption of prioritising the main profitable source of the companies, which are the consumers or the linehaul customers (Cuervo et al., 2014).

3.2.2 Depot

Only a single depot is considered in VRPB. A depot is used to receive and supply the goods. Each route should always start and end at the depot. The depot's capacity has no limit and can provide sufficient goods for the linehaul customer. A vehicle will carry and deliver the goods from the depot to each linehaul customer. On the other hand, the depot also plays as a central storing place for backhaul customers. The vehicle will pick up and return the goods from every backhaul customer to the depot.

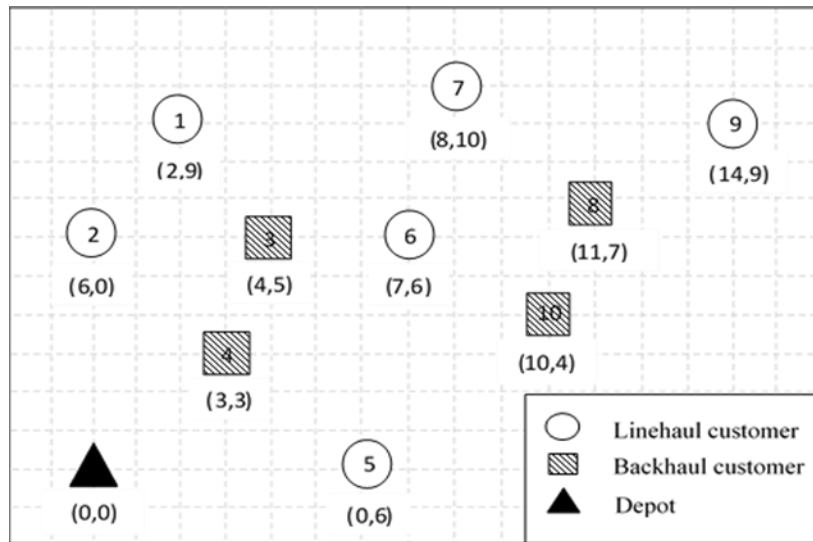


Figure 3.1: Transportation network of VRPB

Figure 3.1 illustrates the transportation network of the depot with a group of customers. The position of each component is determined by the coordinate plane. A coordinate helps to locate a point in space. For example, in Figure 3.1, the depot is located at the origin (0,0), and linehaul customer 1 is located at the point (2,9) with an x-axis value of 2 and a y-axis with a value of 9.

3.2.3 Vehicles

There is a homogeneous fleet of vehicles in VRPB. A vehicle with the same capacity and same size is considered. The capacity of vehicles used in each data instance is predefined. Yet the number of customers served in a single route is not restricted. A vehicle can serve any customer as long as its total loads are under the maximum capacity limit with no violation occurring. The maximum capacity (Q_k) (Refer to Section 3.5.1) is the same for all vehicles.

In every transportation movement, the vehicle should start and end the journey at the depot. A vehicle should not return to the depot during the journey. The vehicle can travel back to the depot only when the maximum capacity is reached or all the customers have been served. Besides, multiple vehicles are allowed in this environment, while a vehicle can have multiple trips in a single route arrangement.

3.3 Constraints

A set of constraints need to be satisfied to develop a suitable algorithm that can produce a feasible solution. The list of constraints for this research is listed below:

- i. The vehicle departs and ends at the origin (depot) for every route.
- ii. Each customer can only be served once.
- iii. The customer is visited by exactly one vehicle only.
- iv. The demand of customers must be fulfilled.
- v. All vehicles will have the same predefined capacity.
- vi. The total capacity cannot exceed the maximum capacity of the vehicle.
- vii. Routes are not allowed to involve only backhaul customers.
- viii. Linehaul customers are served before any backhaul customers.

3.4 Objective Function

The objective function is to minimise the total transportation distance by maximising the utility of the capacity.

3.5 Mathematical Model

The mathematical problem formulation that can describe the problem statement and the relationship of each component is presented in this section.

3.5.1 Notation

I	Set of customers
I_0	Set of all customers and depot (origin)
n	Number of customers
c_{ij}	Distance between customer i and customer j ; $i, j = 1, 2, 3, \dots, n$
c_{0j}	Distance between the depot (origin) and customer j ; $j = 1, 2, 3, \dots, n$
m	Number of vehicles
K	Set of vehicle $K = \{1, 2, \dots, m\}$
Q_k	Maximum capacity of vehicle k
p_i	Pickup demand of customer i ; $i = 1, 2, 3, \dots, n$, $p = 0$ for a linehaul customer
d_i	Delivery demand of customer i ; $i = 1, 2, 3, \dots, n$, $d = 0$ for a backhaul customer
y_{ij}	Load picked up by vehicle k while travelling from customer i to customer j , $i \neq j$
z_{ij}	Load to be delivered by vehicle k while travelling from customer i to customer j , $i \neq j$

Decision variable:

$$x_{ij}^k = \begin{cases} 1, & \text{If vehicle from } i \text{ to } j \text{ by vehicle } k \\ 0, & \text{Otherwise} \end{cases}$$

Equation 3.1 represents the objective function of this research. The corresponding model is shown below

$$\text{Minimise } \sum_{k=1}^M \sum_{i=1}^n \sum_{j=1}^n C_{ij} x_{ij}^k \quad \text{Equation 3.1}$$

subject to

$$\sum_{k=1}^m x_{ij}^k = 1, \quad \forall i, j = 1, 2, 3, \dots, n \quad \text{Equation 3.2}$$

$$\sum_{i=1}^n x_{ij}^k = \sum_{i=1}^n x_{ji}^k, \quad \forall k = 1, 2, 3, \dots, m, \forall j \in i \quad \text{Equation 3.3}$$

$$y_{ij}^k = \sum_{k=1}^M \sum_{i=1}^n \sum_{j=1}^n x_{ij}^k p_j, \quad \forall i, j = 1, 2, 3, \dots, n \quad \text{Equation 3.4}$$

$$z_{ij}^k = \sum_{k=1}^M \sum_{i=1}^n \sum_{j=1}^n x_{ij}^k d_j, \quad \forall i, j = 1, 2, 3, \dots, n \quad \text{Equation 3.5}$$

$$\sum_{k=1}^m \sum_{i=1}^n y_{ji}^k x_{ji}^k - \sum_{k=1}^m \sum_{i=1}^n y_{ij}^k x_{ij}^k = p_j, \quad j \neq 0 \quad \text{Equation 3.6}$$

$$\sum_{k=1}^m \sum_{i=1}^n z_{ij}^k x_{ij}^k - \sum_{k=1}^m \sum_{i=1}^n z_{ji}^k x_{ji}^k = d_j, \quad j \neq 0 \quad \text{Equation 3.7}$$

$$(y_{ij} + z_{ij}) \leq Q_k \sum_{k=1}^m x_{ij}^k, \quad \forall i, j = 0, 1, 2, 3, \dots, n \quad \text{Equation 3.8}$$

$$y_{0j} = 0, \quad j = 1, 2, 3, \dots, n \quad \text{Equation 3.9}$$

$$z_{j0} = 0, \quad j = 1, 2, 3, \dots, n \quad \text{Equation 3.10}$$

The model above represents the problem statement of VRPB. The objective function is to minimise the total travelling distance. Equation 3.2 ensures that each customer is served by exactly one vehicle. Equation 3.3 shows that the vehicle that arrived and departed from a customer is the same. Equation 3.4 represents that the total picked-up goods are the same as the sum of the total pickup demand. Equation 3.5 shows that the sum of the goods loaded on the vehicles at the starting point equals the total delivery demand. Notation y_{ij} and z_{ij} are defined to calculate the current load for pickup and delivery, respectively. This ensures the total load loaded does not exceed the maximum capacity set for all vehicles. Equation 3.6 and Equation 3.7 are the flow equations regarding the amount of pickup and delivery loads. Equation 3.8 ensures the total amount of goods will not exceed the maximum capacity. Equation 3.9 means the total amount of pickup goods starting from the depot is zero, and Equation 3.10 means that at the end of the journey, the total amount of delivery goods loaded must equal zero.

3.6 Conclusion

The problem statement of the VRPB has been discussed in this chapter. This chapter discusses and defines the parameters, benchmark data, constraints, and objective functions.

VRPB is extended from the classical VRP by separating the customer into linehaul and backhaul customer types. VRP has become more comparable to the practical environment after the involvement of linehaul and backhaul customers. Thus, VRPB is considered more valued than VRP in terms of cost-saving in shortening the total transportation distance.

Moreover, VRPB has various constraints. A feasible solution should serve all customers while complying with all the rules. However, the problem has become more

complex when involved with a list of factors. An algorithm that can cope with all the constraints is needed to yield an efficient solution.

CHAPTER 4

VEHICLE ROUTING PROBLEM WITH BACKHAUL

4.1 Introduction

The proposed Vehicle Routing Problem with Backhaul (VRPB) model is discussed in this chapter. VRPB is an extended problem from the Vehicle Routing Problem (VRP) by partitioning a set of customers into linehaul customers and backhaul customers. VRPB can be reduced to a classical VRP when only linehaul customers are considered. Thus, as a generalisation of VRP, VRPB is a Non-deterministic Polynomial-time hard (NP-hard) combinatorial optimisation problem (Zachariadis & Kiranoudis, 2012; Elshaer & Awad, 2020).

VRPB can be solved exactly for small-scale instances. When the scale size increases, it cannot solve using an exact solution (Zachariadis et al., 2009). As the huge size instances are difficult to solve efficiently by mathematical programming or the exact solution, researchers are attracted to solving VRPB by the approximate methods (Osman & Wassan, 2002). Lately, heuristic and metaheuristic-based solutions are more attractive in solving large-scale and practical applications (Parragh et al., 2008; Braekers et al., 2016; Koç & Laporte, 2018). Many heuristic techniques have been developed and applied. Although a heuristic-based algorithm is not able to reach an exact solution, it is suitable to cope with large-scale and complicated problems. Therefore, a heuristics-based algorithm will be proposed and applied in this project.

4.2 Benchmark Data

A set of benchmark instances are used to test the efficiency and effectiveness of the proposed algorithm. The benchmark datasets used in this research are instances introduced by Goetschalckx and Jacobs-Blecha in the paper “The Vehicle Routing Problem with Backhauls” in 1989 (Goetschalckx & Jacobs-Blecha, 1989). It is one of the popular benchmark data used in past literature research related to VRPB (Osman & Wassan, 2002; Gajpal & Abad, 2009; Anbuudayasankar et al., 2014; García-Nájera et al., 2015; Wassan et.al., 2016; Koç & Laporte, 2018).

The benchmark instance consists of the location of every factor (depot, linehaul customers, backhaul customers), vehicle information and the prefixed demand of each customer. The depot is located in the middle position throughout all benchmark instances, with a fixed coordinate in [12000,16000]. However, the number of available vehicles, the vehicle's maximum capacity, the service of each customer type and demand are different. The demand amount is selected from a normal distribution with a means of 500 units and a standard deviation of 200 units (Goetschalckx & Jacobs-Blecha, 1989).

The benchmark datasets are composed of different ratios of linehaul customers and backhaul customers, which are 50:50, 33:67 and 20:80, respectively, with all different demands. On top of that, the x-coordinate of customers is randomly distributed within the interval [0,24000], whereas the y-coordinates are generated within the interval [0,32000].

The benchmark instances used in this research are obtained from Marc Goetschalckx's website (Goetschalckx, 2014) and the VRP-REP website (Mendoza et al., 2014). VRP-REP is a collaborative open data platform to share benchmark instances and solution related to the vehicle routing problem.

This research chooses five sets of small-scale datasets and one large-scale dataset from Goetschalckx and Jacobs-Blecha's benchmark. A total of 22 sets of benchmark instances consisting of a data size from 25 to 113 are selected for use. The benchmark data is formed up of different customer sizes, customer demands, customer positions, vehicle capacity size and the number of vehicles. The vehicle's size range from 3 to 12 is to measure and compare the differences in the total travelled distance to the vehicle involved.

The chosen benchmark datasets can be categorised into six sub-problems, sets A, B, C, D, E, and K. Problem Set A has the smallest customer size, which only involved 25 customers, and problem set K is the largest sub-problem set, which has 113 number of customers in the total. Table 4.1 shows the summary of customer size for each sub-problem set that has been used in this project. Furthermore, the 22 sets of benchmark data used in this research are shown in Table 4.2.

Table 4.1: Customer size of benchmark datasets

Problem Set	Customer Size	Number of Linehaul	Number of Backhaul
A	25	20	5
B	30	20	10
C	40	20	20
D	38	30	8
E	45	30	15
K	113	75	38

Table 4.2: List of benchmark datasets

Dataset	Customer size	No. of linehaul	No. of backhaul	Number of vehicle	Capacity size of vehicle
A1	25	20	5	8	1550
A2	25	20	5	5	2550
A3	25	20	5	4	4050
A4	25	20	5	3	4050
B1	30	20	10	7	1600
B2	30	20	10	5	2600

Table 4.2: continued

B3	30	20	10	3	4000
C1	40	20	20	7	1800
C2	40	20	20	5	2600
C3	40	20	20	5	4150
C4	40	20	20	4	4150
D1	38	30	8	12	1700
D2	38	30	8	11	1700
D3	38	30	8	7	2750
D4	38	30	8	5	4075
E1	45	30	15	7	2650
E2	45	30	15	4	4300
E3	45	30	15	4	5225
K1	113	75	38	10	4100
K2	113	75	38	8	5200
K3	113	75	38	9	5200
K4	113	75	38	7	6200

4.3 Data Pre-processing

This research uses data pre-processing to produce a list of the feasible initial solution. The list of feasible solutions is generated initially before proceeding to the improvement stage.

In this process, the benchmark instances shown in Table 4.2 generate the initial solution. The feasible solutions are required to adapt to all the factors and conditions. The factors such as the size of the customer, location of the customer, customer demand, and vehicle capacity are all taken into consideration.

Besides, all the constraints should be accomplished in this stage. Generally, the distance among all the customer locations is calculated and presented in a distance matrix. The detailed workflow of the step is discussed in the following section.

4.3.1 Data Distance Calculation

Each benchmark instance has provided a coordinated location for the depot and the customers. The distance between each node (customer and depot) is calculated in this step. The depot has a fixed coordinate of [12000, 16000], whereas each customer is randomly located around the x intervals and y intervals between the coordinate position of [24000, 0] and [0, 32000].

The distance between two customer nodes is calculated using the Euclidean Distance formula given in Equation 4.1. Here, (x_1, y_1) are the coordinates of Customer 1, c_1 , whereas (x_2, y_2) are the coordinates for Customer 2, c_2 . Moreover, d_{12} is the distance between (x_1, y_1) and (x_2, y_2) .

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \quad \text{Equation 4.1}$$

Each coordinate can only be allocated by one customer. Overlapping the coordinate location between customers will not happen in the selected benchmark instance. The distances are calculated and recorded in a distance matrix. The distance matrix below is used to illustrate the distance-storing pattern which is used in this project.

$$\begin{array}{c} c_0 \quad c_1 \quad c_2 \quad \cdots \quad c_n \\ \begin{array}{c} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_n \end{array} \left(\begin{array}{ccccc} 0 & d_{01} & d_{02} & \cdots & d_{0j} \\ d_{10} & 0 & d_{12} & \cdots & d_{1j} \\ d_{20} & d_{21} & 0 & \cdots & d_{2j} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ d_{i0} & d_{i1} & d_{i2} & \cdots & 0 \end{array} \right) \end{array}$$

Figure 4.1: Distance matrix

Figure 4.1 is the distance $n \times n$ distance matrix used to represent the distance between the data set with n number of customers and a depot. Element c_0 represents the

depot, and all customers are presented by element c_1 until c_n . Matrix value d_{01} is the distance between the depot and Customer 1, whereas d_{12} is the distance between Customer 1 and Customer 2.

Theoretically, the distance of an object to or from itself is always equal to zero. Therefore, the value of the main diagonal is all zeros. Besides, the value of the upper triangular part of the distance matrix is a mirror of its lower triangular. Meanwhile, the distance travelled from Customer 1 to Customer 2, d_{12} is the same as the distance travelled from Customer 2 to Customer 1, d_{21} .

Figure 4.2 below shows the data of Problem Set A that was recorded and used in this project. Columns customer_id, x-axis, and y-axis represented the customer nodes, the x-interval values, and the y-interval values of the customer respectively.

customer_id	x-axis	y-axis	customer_type	demand	
1	21,524	24,879	1	483	Linehaul customer
2	22,337	17,915	1	389	
3	7,642	9,722	1	435	
4	3,707	20,036	1	452	
5	15,492	2,721	1	343	
6	12,430	10,638	1	444	
7	13,595	1,918	1	350	
8	4,290	2,721	1	307	
9	18,684	5,268	1	810	
10	8,686	9,546	1	1,002	
11	23,140	10,989	1	476	
12	5,263	14,289	1	674	
13	19,557	17,564	1	691	
14	23,281	29,684	1	675	
15	374	22,683	1	410	
16	18,804	9,082	1	351	
17	5,444	25,004	1	786	
18	20,390	25,745	1	59	
19	4,310	21,692	1	362	
20	8,526	2,495	1	550	Backhaul customer
21	9,795	21,510	2	549	
22	7,758	10,895	2	558	
23	12,927	2,464	2	851	
24	23,286	5,538	2	388	
25	7,416	15,826	2	194	

Figure 4.2: Data of Problem Set A

Customers with the customer type of value 1 denoted the linehaul customers, and the backhaul customers are the customers with customer_type which is equal to 2. The demand of each customer has been listed in the demand column.

Figure 4.3 below shows the sample of the distance matrix for Problem Set A generated in this project. customer_id 0 denoted the depot and customer_id 1 to 10 are Customer 1 to Customer 10. Columns d_0 to d_10 represented the distance between two respective location points. For example, for the row in which customer_id equals 1, d_0 is the distance between Customer 1 and the depot, and d_2 is the travel distance between Customer 1 and Customer 2.

customer_id	d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_10
0	0.00	13,020.88	10,512.89	7,642.35	9,222.97	13,730.47	5,379.21	14,172.04	15,355.00	12,643.25	7,255.12
1	13,020.88	0.00	7,011.30	20,553.46	18,463.48	22,964.36	16,896.95	24,291.49	28,071.12	19,815.57	19,997.88
2	10,512.89	7,011.30	0.00	16,824.63	18,750.35	16,664.68	12,292.41	18,229.83	23,591.35	13,164.00	16,012.18
3	7,642.35	20,553.46	16,824.63	0.00	11,039.15	10,518.39	4,874.83	9,815.33	7,762.08	11,906.46	1,058.73
4	9,222.97	18,463.48	18,750.35	11,039.15	0.00	20,945.06	12,822.37	20,640.60	17,324.81	21,033.41	11,611.66
5	13,730.47	22,964.36	16,664.68	10,518.39	20,945.06	0.00	8,488.51	2,059.96	11,202.00	4,083.63	9,638.58
6	5,379.21	16,896.95	12,292.41	4,874.83	12,822.37	8,488.51	0.00	8,797.48	11,355.11	8,243.14	3,900.00
7	14,172.04	24,291.49	18,229.83	9,815.33	20,640.60	2,059.96	8,797.48	0.00	9,339.58	6,092.65	9,071.09
8	15,355.00	28,071.12	23,591.35	7,762.08	17,324.81	11,202.00	11,355.11	9,339.58	0.00	14,617.61	8,118.22
9	12,643.25	19,815.57	13,164.00	11,906.46	21,033.41	4,083.63	8,243.14	6,092.65	14,617.61	0.00	10,874.80
10	7,255.12	19,997.88	16,012.18	1,058.73	11,611.66	9,638.58	3,900.00	9,071.09	8,118.22	10,874.80	0.00

Figure 4.3: Distance matrix of Problem Set A

4.4 Initial Solution

In this stage, a simple priority rule heuristic is applied to generate the initial feasible solution. In order to fulfil the constraints of VRPB, some pre-arrangement is carried out before executing the sorting process.

First, the number of routes is dependent on the number of vehicles given for each benchmark dataset. For example, 8 different routes are generated in dataset A1 since the predefined vehicle for it is 8. Then, as a vehicle should depart from the depot, the depot will be the first and the last node for every single route. Next, in VRPB, the solution with only

backhaul customers is not allowed. Meanwhile, the number of linehaul customers in each route should equal or more than one. In order to cope with this constraint, at least one linehaul customer needs to be reserved for every route. Hence, in this study, the second node always starts with a linehaul customer. Figure 4.4 represents the structure of each route in the initial solution.

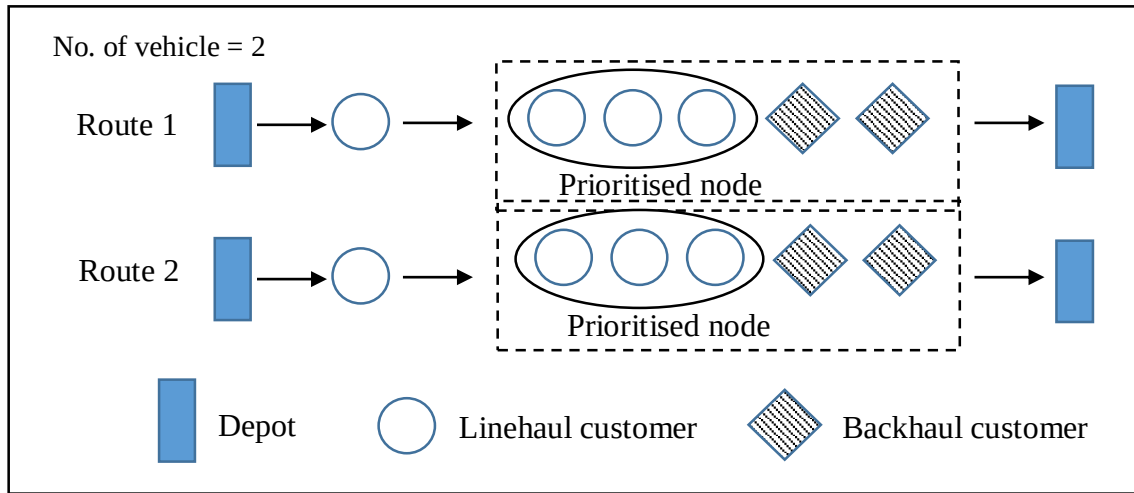


Figure 4.4: Sorting structure for each route in VRPB

Moreover, as observed from Figure 4.4, the linehaul customers are prioritised to be selected as the next serving customer before any backhaul customer. The linehaul customers are considered first due to the constraint in which linehaul customers are served before backhaul customers.

Figure 4.5 below represents the vehicle travelling movement of this research in a sample transportation network. The serving sequence for linehaul customers is determined before the backhaul customers. Besides, the vehicle always starts and ends at the depot.

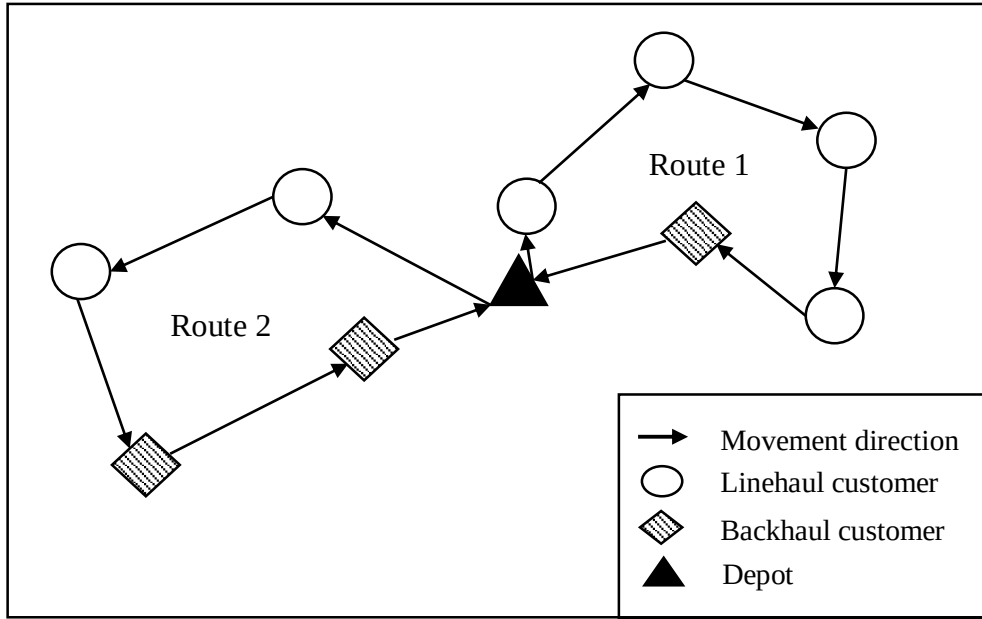


Figure 4.5: Vehicle travelling movement of VRPB

The route is generated based on the priority rule heuristic. The initial solution is formed based on the following consideration:

- i. Set the nearest linehaul customer as the first node to be visited in each route.
- ii. Seek the next linehaul customer with the nearest distance. Check on the total loaded capacity size.
 - If the total capacity is under the limit, set the customer as served and searched for the next node.
 - Else, seek the second nearest linehaul customer from the last node and check the total capacity loaded.
- iii. Look for the backhaul customer with the nearest travel distance from the last node of the route and check on the capacity after no possible linehaul customer can be served.
 - If the total capacity is under the capacity limit, set the customer as served and searched for the next node.

- Else, seek the second nearest backhaul customer from the last node and check the total capacity loaded.

In short, the linehaul customers with the shortest travel distance from the depot will be considered and selected first before any backhaul customer. Figure 4.6 represents assigning the second node (linehaul customer) for each route based on the distance travelled.

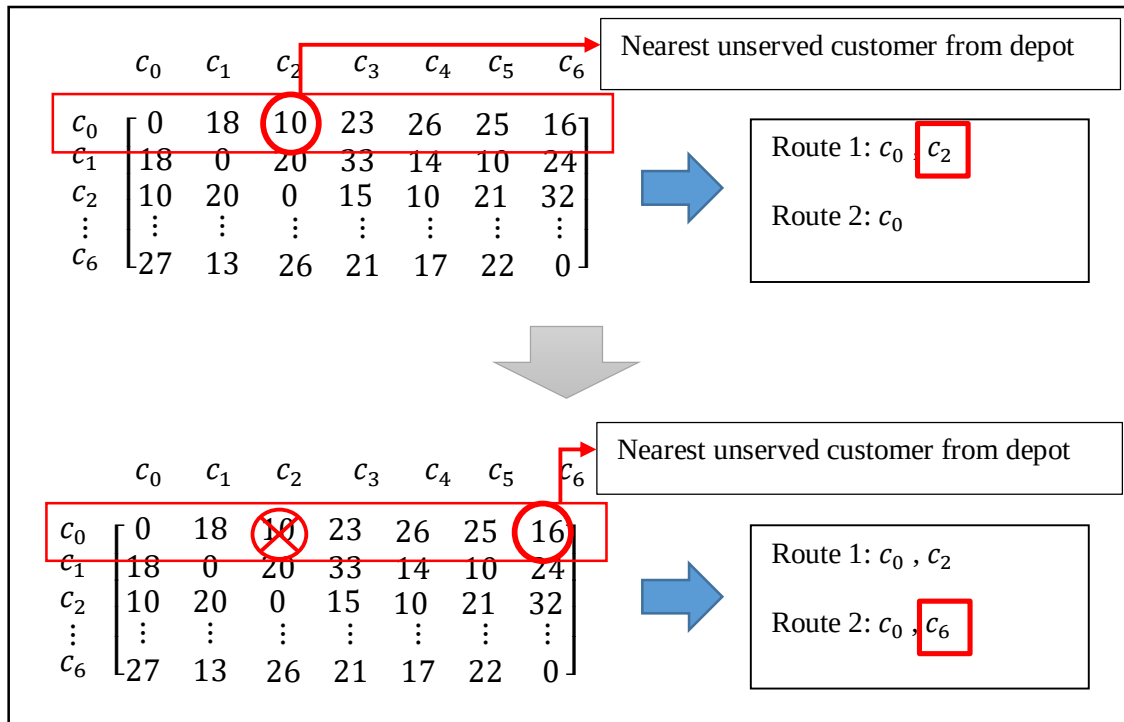


Figure 4.6: Node assignment (Second node)

Then, starting with the first route, the next linehaul customer with the nearest distance is assigned when going through the priority algorithm. Figure 4.7 represents the assignment process depending on the travel distance.

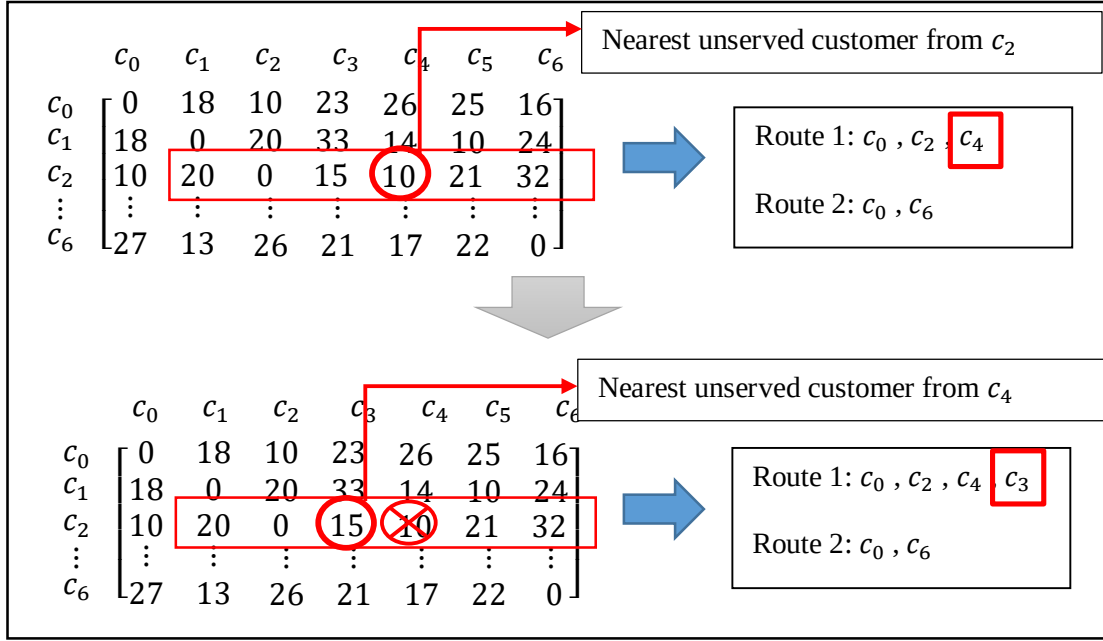


Figure 4.7: Node assignment (shortest travelled distance)

The capacity constraint is determined to make sure no violation has occurred. The vehicle needs to carry and supply the load from a depot. This checking has ensured the total load carried will not exceed the maximum capacity of the vehicle. This process is repeated until there is no suitable linehaul customer. The algorithm for the priority rule for linehaul customers starting from the depot is defined as in Algorithm 4.1.

Algorithm 4.1: Priority Rule Heuristic for Linehaul Customer

n = total number of unserved linehaul customer,

Q = maximum capacity of vehicle,

D = Summation of load deliver by the vehicle for customer, x_{i-1} ,

d = delivery demand for x

WHILE $n > 0$

$x_0 = 0$

FOR $i = 1$ to n

Assume that x_i is the nearest linehaul customer from the previous node, x_{i-1}

IF $(D + d_i) \leq Q$

$x_i = i$; $D = D + d_i$

ELSE

Skip x_i

ENDIF

ENDFOR

Next, the backhaul customers are considered after all linehaul customers have been served. The backhaul customer with the shortest travelling distance from the last node is selected, and capacity checking is executed. A backhaul customer is a customer required for the pickup service. Hence, capacity checking is now focused on the total loads collected along the way before the vehicle returns to the depot. The algorithm for the priority rule for backhaul customers and end at the depot is defined in Algorithm 4.2.

Algorithm 4.2: Priority Rule Heuristic for Backhaul Customer

n = total number of unserved backhaul customer,
 Q = maximum capacity of vehicle,
 P = Summation of load pickup by the vehicle for customer, x_{i-1} ,
 p = pickup demand for x
 FOR $i = 1$ to n , where $n > 0$
 Assume that x_i is the nearest backhaul customer from the previous node, x_{i-1}
 IF $(P + p_i) \leq Q$
 $x_i = i$; $P = P + p_i$
 ELSE
 Skip x_i
 ENDIF
 ENDFOR
 $x_{i+1} = 0$

The vehicle is returned to the depot when the maximum capacity loaded has been reached, or no suitable customer can be found. The process of forming a single route stops here and starts to form the next route. The data pre-processing is terminated until all the customers are served exactly once. Lastly, the total travel distance of each route is calculated and summed up. The final solution yields throughout the initial solution and has become the feasible solution used in the improvement stage. Figures 4.8 and 4.9 below show the flow charts of the overall algorithm used to produce the initial solutions. Figure 4.8 represents the data flow diagram for linehaul customers, and Figure 4.9 illustrates the data flow diagram for the backhaul customers.

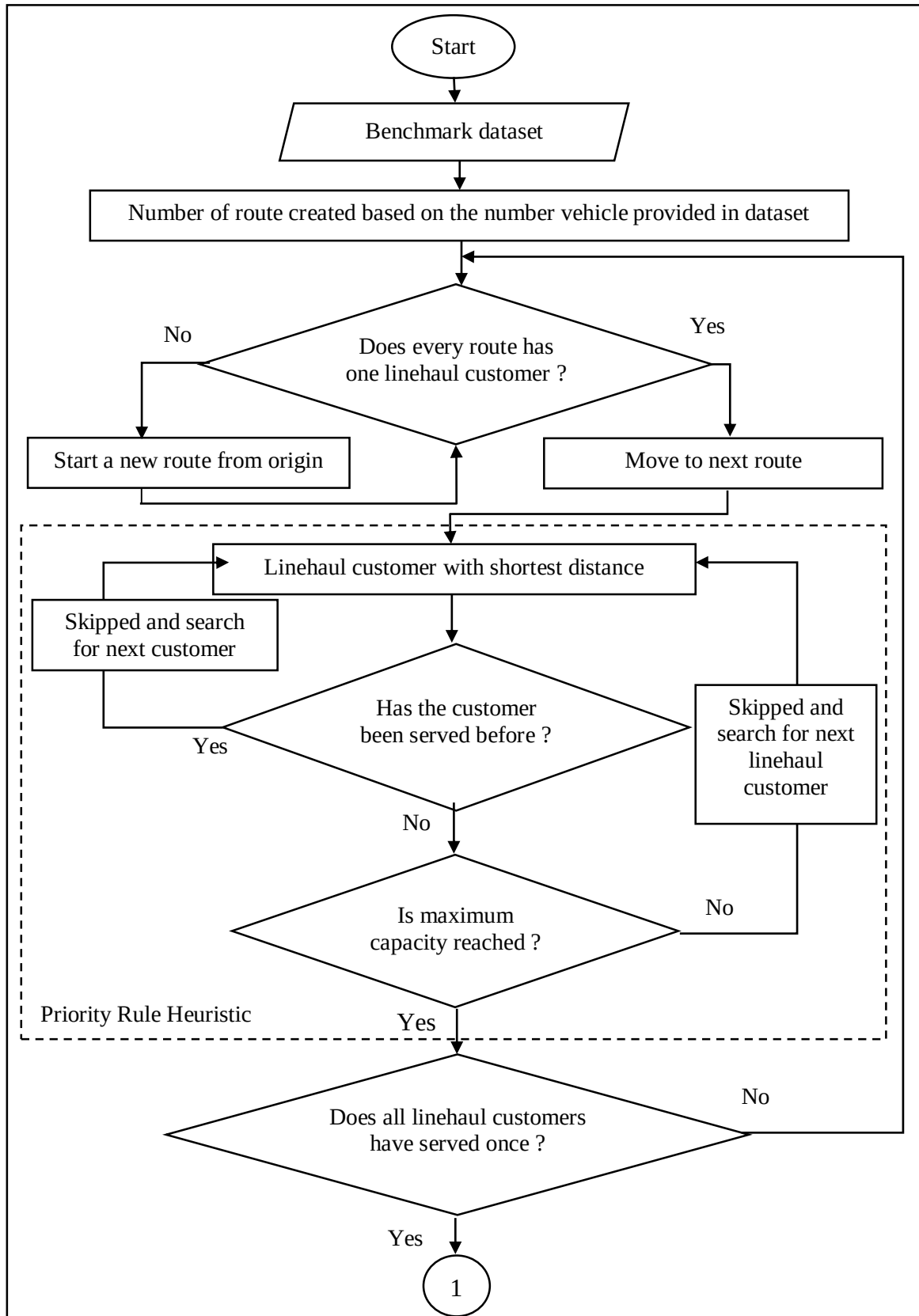


Figure 4.8: Data flow diagram of VRPB linehaul customer

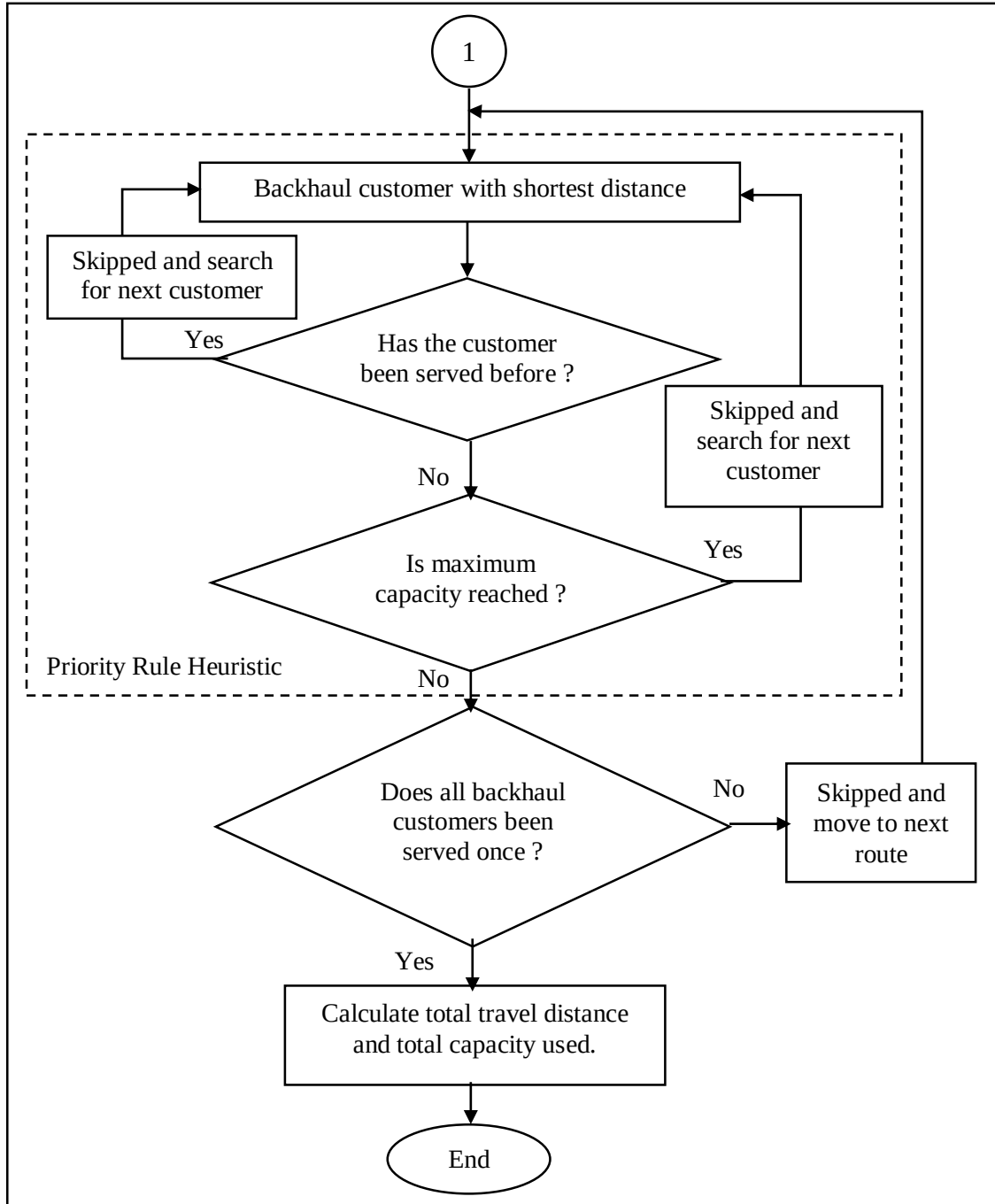


Figure 4.9: Data flow diagram of VRPB backhaul customer

Figure 4.10 below shows the initial solution for Dataset A1, generated by the proposed priority rule heuristic obtained in this project. Dataset A1 has a predefined total of 8 vehicles with a maximum capacity of 1550. Therefore, the generated solution has 8 routes and the total loads in each route are always under 1550. In Figure 4.10, each route_no

represents one vehicle, and the data in the route_list column shows the travel movement of each particular vehicle. Based on Figure 4.10, in Route 1, the vehicle travels from depot 0 to serve Customers 6, 9, 18, 23, and 22 consecutively, then returns to the depot. The total_travelled_distance column represents the total distance travelled in each route, and the total_delivery_amount and total_pickup_amount are the summaries of the vehicle's capacity in each route. As shown in Figure 4.10, backhaul customers are served after the linehaul customers.

route_no	route_list	total_travelled_distance	total_delivery_amount	total_pickup_amount
1	0 6 9 18 23 22 0	75145.074724577	1313	1409
2	0 12 17 21 24 25 0	67655.135708622	1460	1131
3	0 10 1 0	40273.870711847	1485	0
4	0 3 14 15 0	70363.871555988	1520	0
5	0 13 20 8 0	45989.224125895	1548	0
6	0 4 11 2 0	48143.960651076	1317	0
7	0 19 7 5 0	47203.238421383	1055	0
8	0 16 0	19406.508193344	351	0

Figure 4.10: Initial solution for Dataset A1

4.5 Variable Neighbourhood Search Algorithm

This stage is considered the improvement stage of this research. The Variable Neighbourhood Search (VNS) is applied to improve the initial solution obtained in the data pre-processing stage. VNS constructed by a list of local search procedures and shaking methods. Three local search techniques and two random shaking methods have been proposed in this research:

- i. Local Search Techniques
 - a. Local 1-to-0 Swap
 - b. Local 2-to-1 Swap
 - c. Local 1-to-1 Swap

ii. Shaking Methods

- a. Random Multiple 1-to-0 Swap
- b. Random Pick Solution

In this research, the local search techniques are arranged in a static structure. The proposed three local search techniques are grouped as local improvement processes. The first local search is the Local 1-to-0 Swap, followed by the Local 2-to-1 Swap, and the last is the Local 1-to-1 Swap. Each local search method will be run exactly once in each improvement iteration. While the shaking methods are random pick among the Random Pick Solution and Random Multiple 1-to-0 Swap. Figure 4.11 shows the two possible algorithm structures that can take place in the proposed VNS.

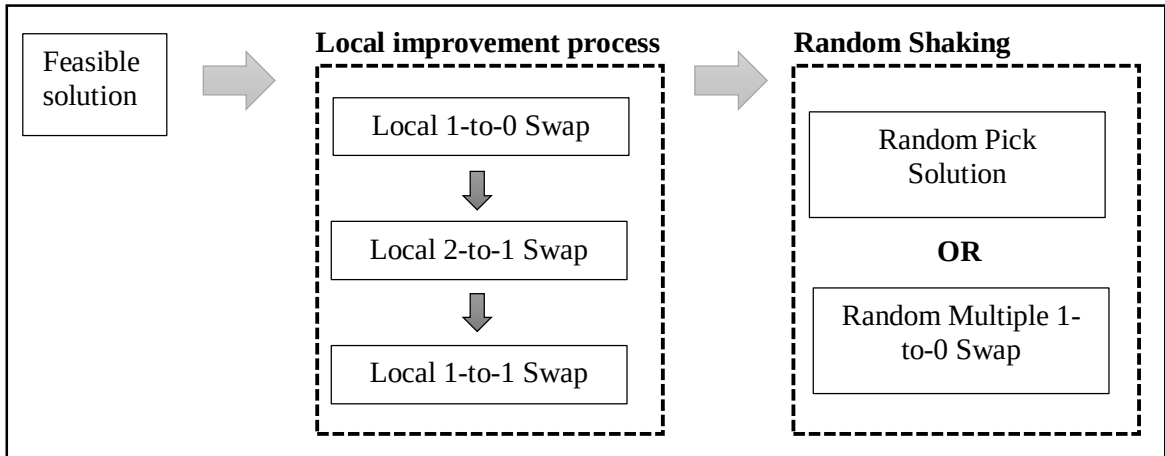


Figure 4.11: Algorithm structure of VNS

Local search techniques are used to obtain the possible neighbourhood combinations in order to increase the chances of reaching the optimum. The structure of the feasible solution produced in the initial stage is rearranged systematically, and a list of neighbourhood solutions is obtained through the local search algorithm.

In a local improvement process, the current best solution is kept updated if the newly yielded solution is able to achieve a better result. In this project, the total travelled distance

is used to weigh the quality of the yielded solution. The total travelling distance of each feasible solution is calculated, and the capacity constraint is checked. Hence, the new feasible solution with the shortest total travelled distance will replace the current best solution structure. The process flow used to determine the best solution structure in the local search process is defined in Algorithm 4.3.

Algorithm 4.3: Determine the best solution structure

n = Total number of node in the solution route,

d = Distance travel from x_i to x_{i+1} ,

S_{best} = Current best feasible solution structure with shortest travelled distance

D_{best} = Current shortest travelled distance

S_{new} = New feasible solution structure

D_{new} = New total travelled distance

Assume that S_{new} has complied with all constraints

FOR $i = 1$ to n

$D_{new} = D_{new} + d_i$

ENDFOR

IF $D_{new} < D_{best}$

$D_{best} \leftarrow D_{new}$

$S_{best} \leftarrow S_{new}$

ELSE

Skip S_{new}

ENDIF

The rearrangement process is executed repeatedly until a local optimum is reached. This local improvement process ends when no new improvement structure can be generated. Then, the next local search process is launched using the same process flow. However, the feasible solution that is discharged is saved and kept in a solution list.

In short, the best solution obtained in a local search process will become the input solution for the next local improvement process. Subsequently, once a new feasible solution has been obtained, new local improvement processes will be carried out again, starting from the first local search. Along the way, the current best solution is updated when an

improvement is met. The local improvement processes are executed repeatedly until no further improvement. In this case, the process is terminated, and the number of improvement iterations is increased by one. Figure 4.12 provides the flow of the local search algorithm of this research.

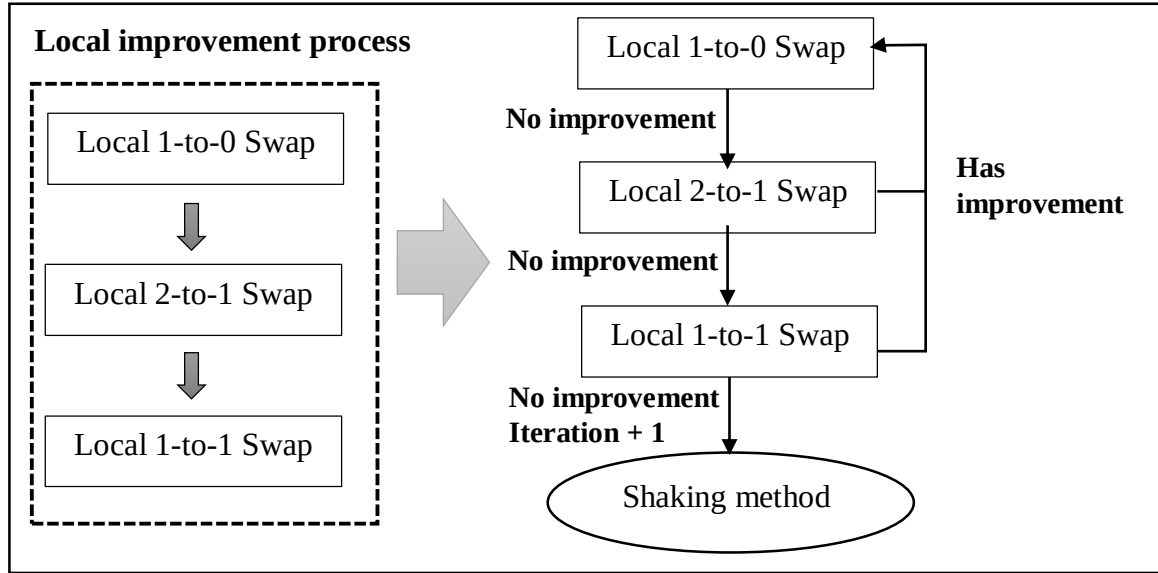


Figure 4.12: Local search algorithm of VNS

Next, the random shaking method is applied to get the new neighbour structure. Each shaking method contains stopping criteria. If no feasible solution can be generated and the stopping criteria have been reached, another shake method will be applied.

Shaking iteration ended if a feasible solution had been obtained. On the other hand, if no feasible solution can be found after all shake methods have been tried, the shaking process will be skipped. In this condition, the last improvement solution will be treated as the new feasible solution and applied to the local search process. Moreover, VNS ends when the maximum iteration number of improvement is reached. The maximum iteration set for this project is 30 iterations. The data flow diagram for the improvement stage is shown in Figure 4.13. The shaking and local search algorithm details are discussed in the subsection below.

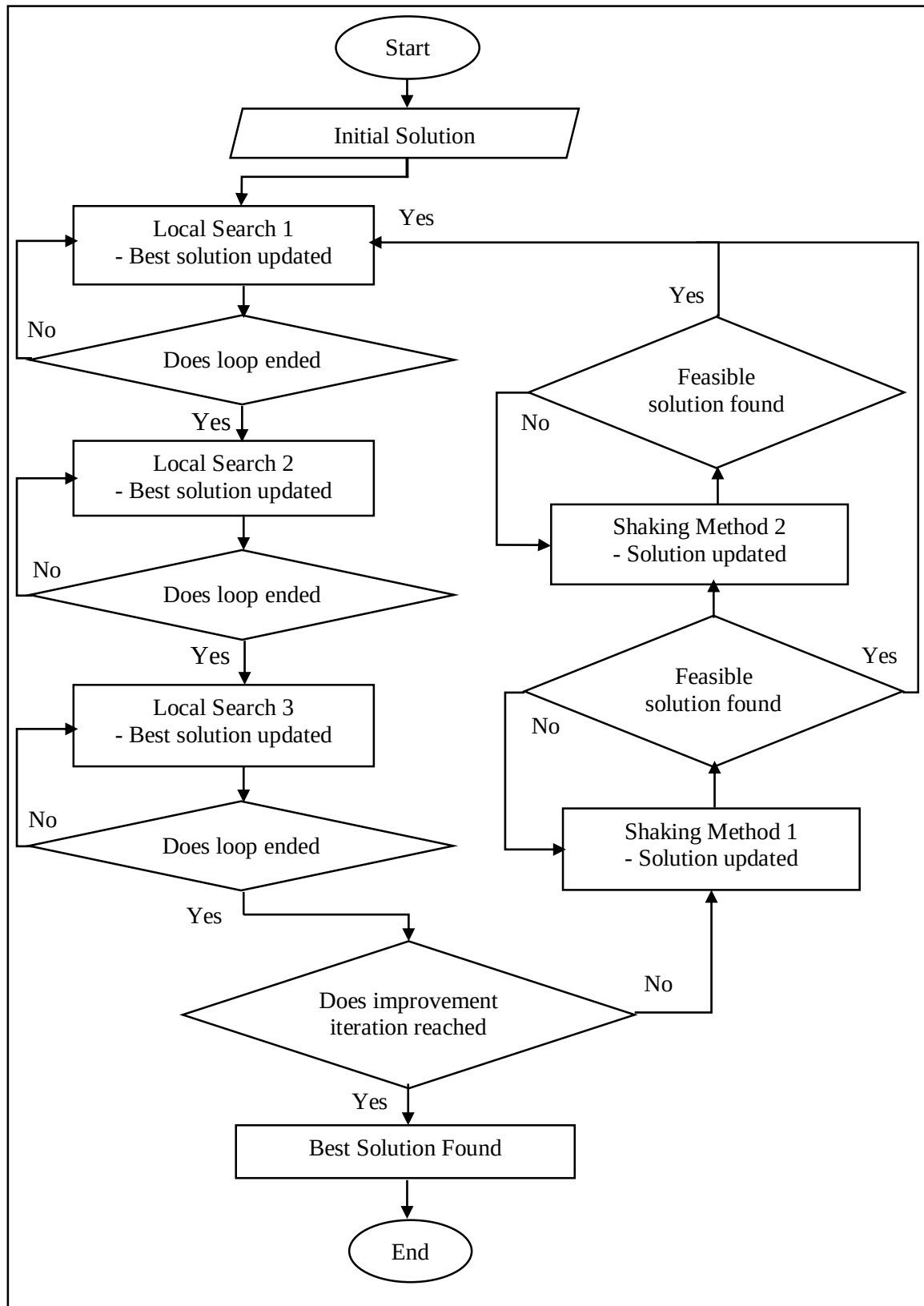


Figure 4.13: Data flow diagram of VNS

4.5.1 Local Search Techniques

4.5.1.1 Local 1-to-0 Swap

A customer (linehaul or backhaul) from a route is shifted to an alternative route until all customers have been selected once. The feasible outcome must comply with the constraint that the backhaul is served after the linehaul customer, and no route should contain only the backhaul customer. Figure 4.14 shows the schematic flow of the Local 1-to-0 Swap.

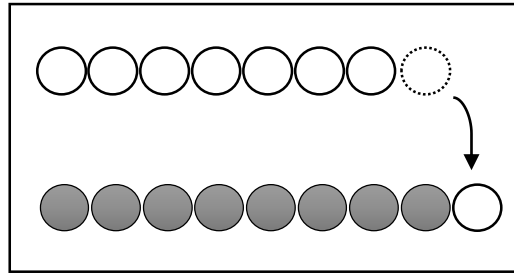


Figure 4.14: Local 1-to-0 Swap

Figure 4.15 below demonstrates the Local 1-to-0 Swap performed in the customer node, c_1 , where the structure is rearranged systematically to obtain a list of feasible neighbourhood solutions. Assume that customers, $c = \{1, 2, 3, \dots, 7\}$ are linehaul customers whereas c_8 and c_9 are backhaul customers, while c_0 represented the depot. Generally, the Local 1-to-0 Swap moves a node from its current position to another location in a different route. The position of the first customer node, c_1 is changed systematically to test out all the possible solutions that can be formed. As shown in Figure 4.15, solutions 1, 2, 3, 4, and 5 are the new neighbourhood structures of the current best solution.

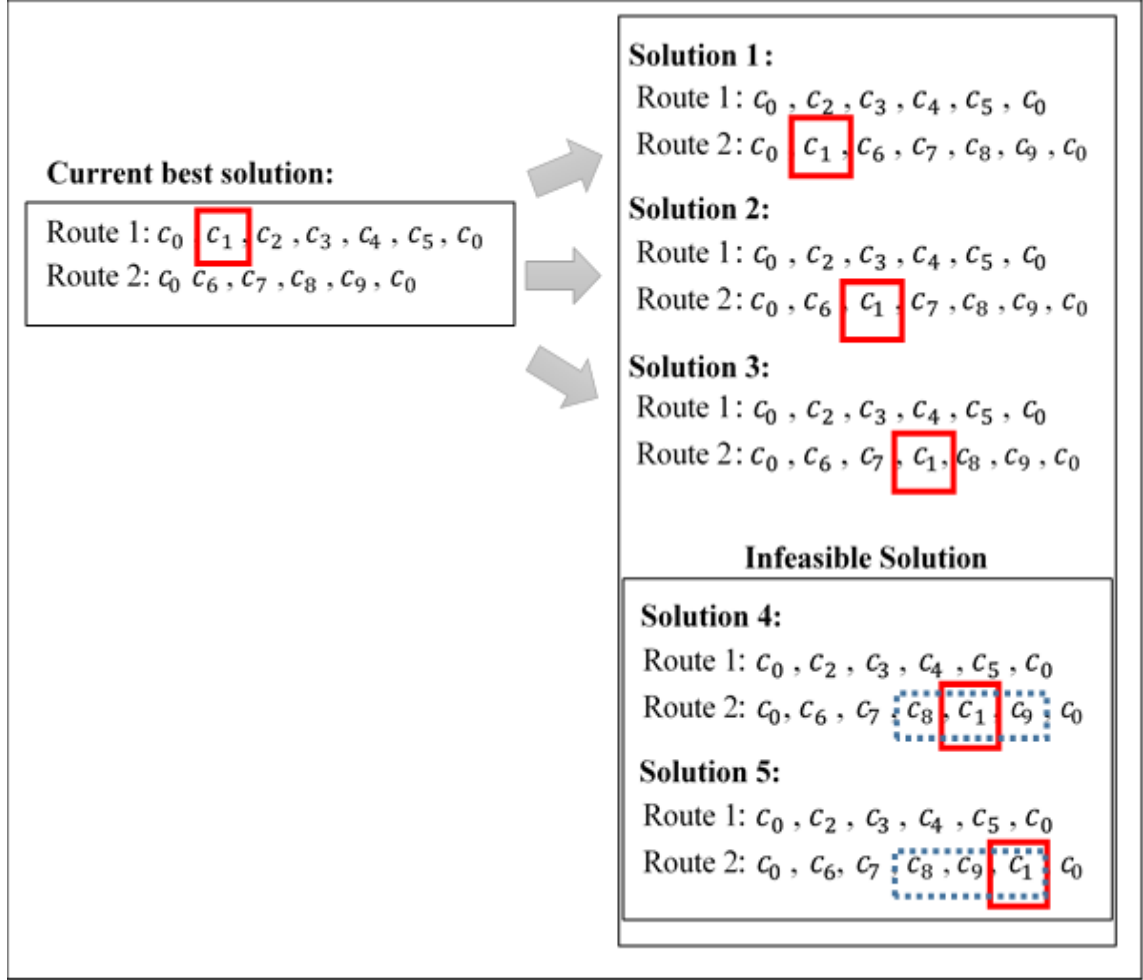


Figure 4.15: Lists of neighbourhood structure of node c_1

Solutions 1, 2 and 3 are feasible. However, solutions 4 and 5 are infeasible. In this scenario both c_8 and c_9 are backhaul customers, and based on the constraint condition, a linehaul customer should be served before the backhaul customers. Therefore, the linehaul customer, c_1 cannot be relocated behind c_8 and c_9 . Thus, solutions 4 and 5 are not taken into consideration in this example.

Once a feasible structure is obtained, the summation of the total travelling distance of each route of that structure is calculated. The total travelled distance is used to determine the performance of the solution. The solution structure with the shorter travelled distance is denominated as the new current best solution structure, as shown in Algorithm 4.3. If found

that solution 2 is able to produce a better solution than the current best solution, solution 2 will be recorded as the new current best solution, and the local search procedure will restart again with the new current best solution. In this circumstance, solution 3, shown in Figure 4.16, will not be generated. When no improved solution can be found after all the customer nodes are tested with the Local 1-to-0 Swap, the algorithm will move to the next local search method.

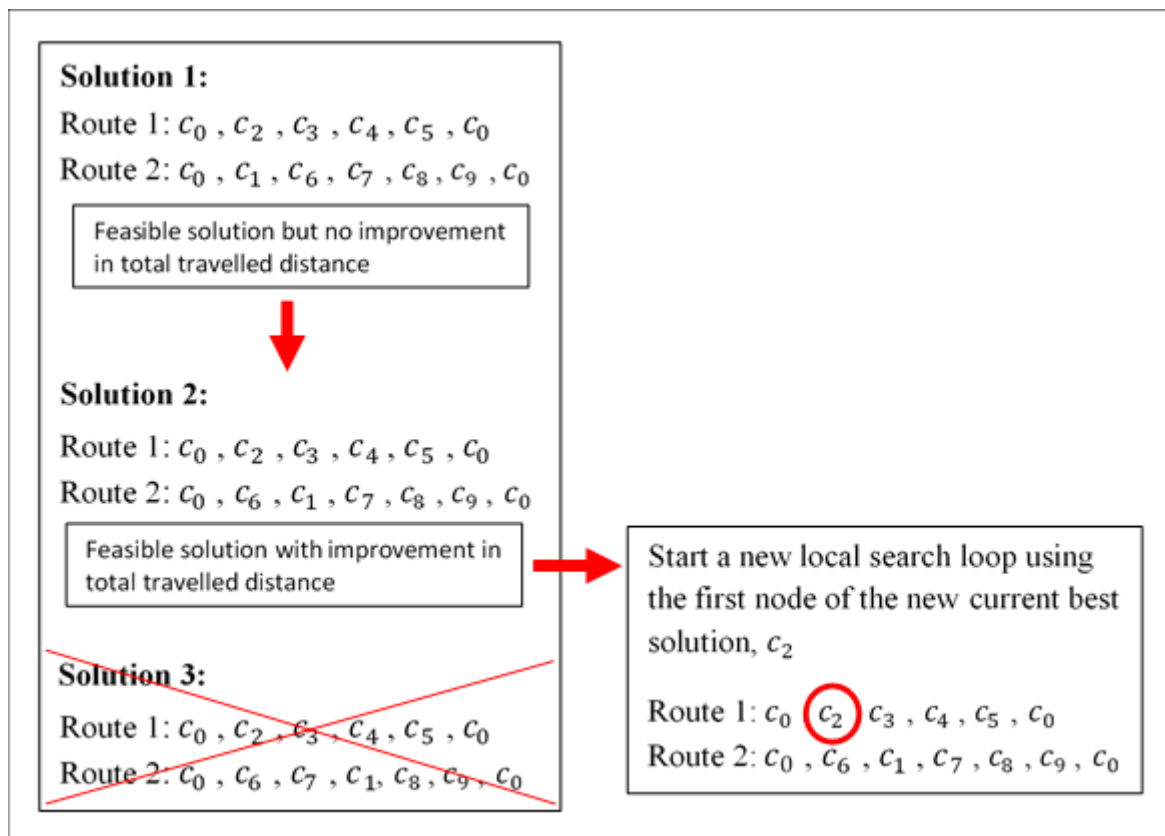


Figure 4.16: Process flow when better feasible solution has found

4.5.1.2 Local 2-to-1 Swap

Two consecutively arranged customers (both linehaul or backhauls or mixed linehaul and backhaul) in a certain location are swapped iteratively with a customer (linehaul or backhaul) from a different route until all feasible structures are tested. Figure 4.17 shows the pattern of Local 2-to-1 Swap.

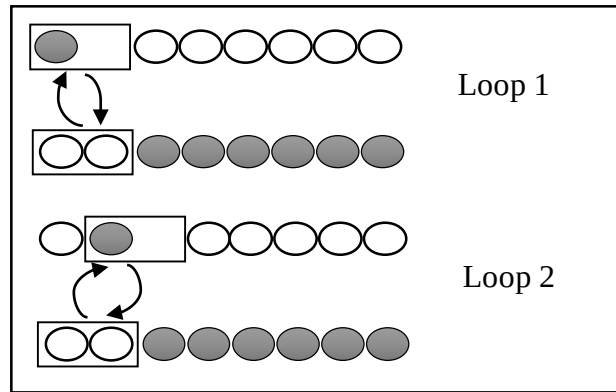


Figure 4.17: Local 2-to-1 Swap

The possible 2-to-1 swapping combinations are:

- i. Two linehaul customers are swapped with a linehaul customer from a different route.
- ii. Two backhaul customers are swapped with a backhaul customer from a different route.
- iii. A linehaul and a backhaul customer that line up one after the other are swapped with the first or last linehaul customer from a different route.

4.5.1.3 Local 1-to-1 Swap

A customer's location (linehaul or backhaul) is swapped iteratively with another customer (linehaul or backhaul) from a different route until all feasible structures are tested. Customers who choose to swap should have the same customer type. The only swap that can happen between a linehaul and a backhaul customer is when the last linehaul customer in a route is swapped with the first backhaul customer in another route and vice versa. The possible swapping pattern of the Local 1-to-1 Swap is shown in Figure 4.18.

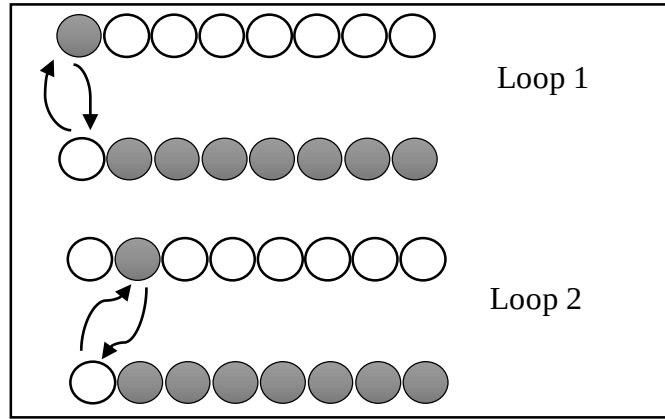


Figure 4.18: Local 1-to-1 Swap

Figure 4.19 below shows the sample of two feasible structure sets for Dataset A1 obtained from undergoing Local-1-to-1 Swap. Route numbers 1 to 8 are the initial solution of Dataset A1. Route numbers 9 to 16 are the feasible neighbourhood solution obtained by swapping customer node 6 and customer node 1. However, routes number 17 until number 24 are the second feasible set of Dataset A1 that was found after swapping customer node 6 with customer 16. In this case, the second solution with the shorter travelled distance is selected to be the current local optimum solution.

route_no	route_list	exchange_values	total_delivery_amount	total_pickup_amount	total_travelled_distance	
1	0 6 9 18 23 22 0	Initial Solution	1313	1409		
2	0 12 17 21 24 25 0	Initial Solution	1460	1131		
3	0 10 1 0	Initial Solution	1485	0		
4	0 3 14 15 0	Initial Solution	1520	0		
5	0 13 20 8 0	Initial Solution	1548	0		
6	0 4 11 2 0	Initial Solution	1317	0		
7	0 19 7 5 0	Initial Solution	1055	0		
8	0 16 0	Initial Solution	351	0	414180.8841	Initial solution
9	0 1 9 18 23 22 0	6 AND 1	1352	1409		
10	0 12 17 21 24 25 0	6 AND 1	1460	1131		
11	0 10 6 0	6 AND 1	1446	0		
12	0 3 14 15 0	6 AND 1	1520	0		
13	0 13 20 8 0	6 AND 1	1548	0		
14	0 4 11 2 0	6 AND 1	1317	0		
15	0 19 7 5 0	6 AND 1	1055	0		
16	0 16 0	6 AND 1	351	0	409655.4349	1 st Local 1-to-1 Swap outcome
17	0 16 9 18 23 22 0	6 AND 16	1220	1409		
18	0 12 17 21 24 25 0	6 AND 16	1460	1131		
19	0 10 1 0	6 AND 16	1485	0		
20	0 3 14 15 0	6 AND 16	1520	0		
21	0 13 20 8 0	6 AND 16	1548	0		
22	0 4 11 2 0	6 AND 16	1317	0		
23	0 19 7 5 0	6 AND 16	1055	0		
24	0 6 0	6 AND 16	444	0	405429.5877	2 nd Local 1-to-1 Swap outcome

Figure 4.19: Lists of outcome for Dataset A1 by Local 1-to-1 Swap

4.5.2 Shaking Methods

4.5.2.1 Random Multiple 1-to-0 Swap

One customer (linehaul or backhaul) is selected randomly from a randomly selected route and shifted to an alternative random route. The feasible outcome must comply with the constraint that backhaul is served after the linehaul customer. This method is looped for a random iteration from 3 to 10 times. The outcome produced in the first loop is the input for the second loop. Figure 4.20 shows the flow of Random Multiple 1-to-0 Swap with 2 loop iteration.

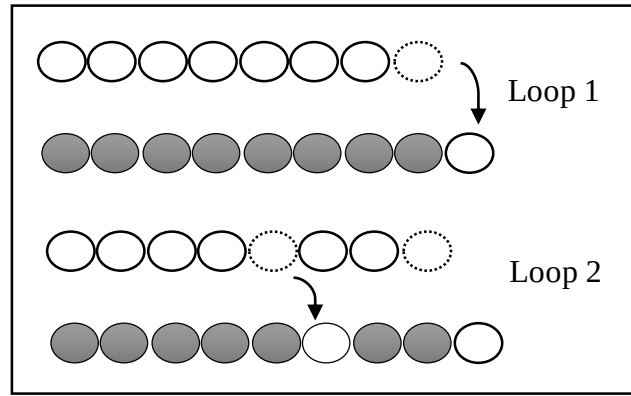


Figure 4.20: Random Multiple 1-to-0 Swap

Figure 4.21 shows the sampling process for the Random Multiple 1-to-0 Swaps. Assume that customers, $c = \{1, 2, 3, \dots, 7\}$ are linehaul customers, c_8 and c_9 are backhaul customers, c_0 represented the depot, and the randomly generated iteration number for this shaking method is equal to 3 times.

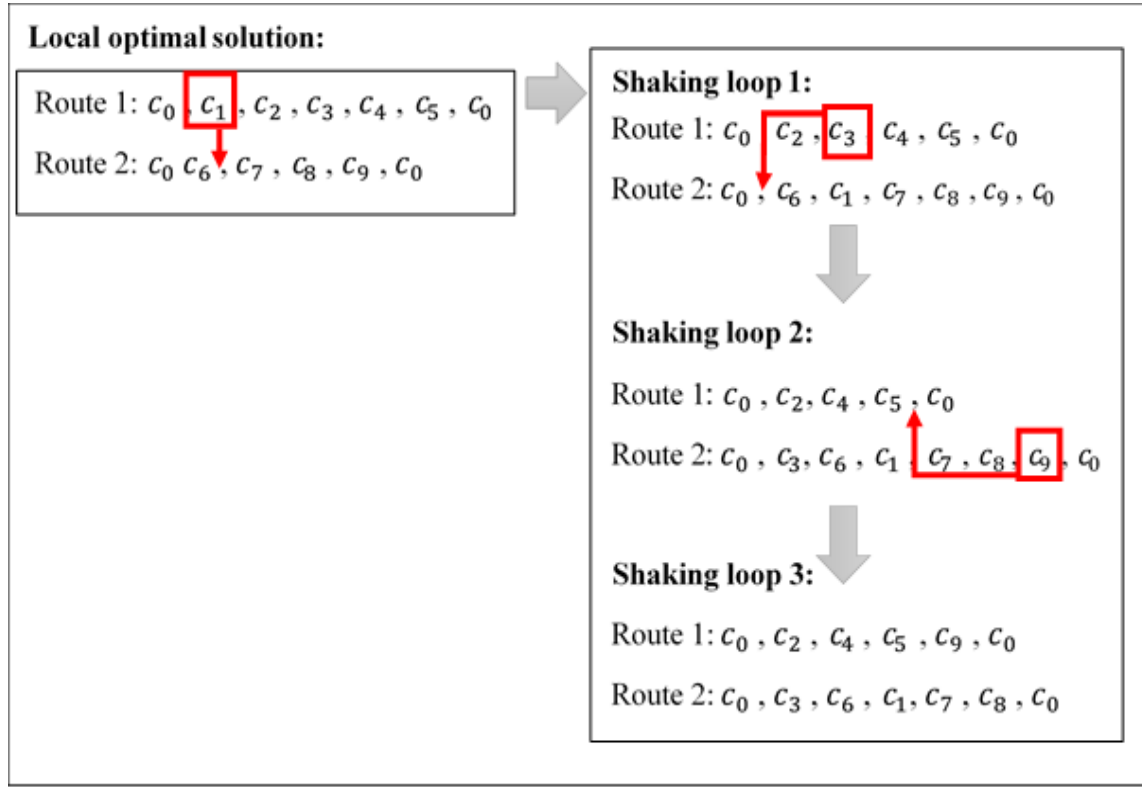


Figure 4.21: Outcomes of 3 times Random Multiple 1-to-0 Swap

As shown in Figure 4.21, the customer c_1 is randomly selected to be swapped from a locally optimal solution. Different from the process in the local search, the swapping process for the shaking methods can be performed randomly and allocated to a random position. In Figure 4.21, the customer c_1 is selected to re-allocate in between c_0 and c_6 in Route 2, as long as it is under the constraint condition. Next, in shaking loop 1, the customer c_3 is randomly selected to be the customer to be swapped. The selected customer can be either a linehaul or backhaul customer. Lastly, the feasible solution produced after undergoing the shaking method is updated to be the new solution. The new solution is then improved in the new local search cycle.

4.5.2.2 Random Pick Solution

A set of feasible solution sets is randomly selected from the solution list. The solution list is the list of the feasible solutions generated along the VNS process. This method is

proposed in case no feasible solution can be generated in Random Multiple 1-to-0 Swap. This method ensures that there is always a feasible solution that can be improved in the next local search techniques.

In summary, the algorithm for local search is developed to always refer to and start the swapping process from the current best solution. The final solution must be better than the current best solution with all constraints met. On the other hand, random shaking will always perform the swapping operation, starting with the last swapped combination obtained. The final outcome may not be the best, but it needs to be feasible.

Furthermore, compared to the common practice in VNS, a random shaking solution is randomly picked from the generated k th neighbourhood solutions (Hansen et al., 2008). The Random Multiple 1-to-0 Swap is proposed to be incorporated into the proposed VNS to gain a more significant shaking solution. Table 4.3 shows the comparison between local search and random shaking methods.

Table 4.3: Differences between local search and random shaking method

Comparison Item	Local Search	Random Shaking
Solution be undergoing swapping operation	Always refer to the current best solution	Always refer to the last swapped combination solution
Final solution	Improved and feasible solution	Last and feasible solution

4.6 Result and Analysis

In this research, the experiments are run using a computer at 2.5GHz on an Intel Core i5 processor with 4GB RAM. The operating system is 64-bit Windows 7 Premium. Furthermore, the algorithm is written in php, and the xampp web server is used for debugging

and compiling the system. The database of the system is connected and managed using phpMyAdmin.

The 22 sets of benchmark instances with the various number of linehaul and backhaul customers, numerous vehicles, and different vehicle capacities are used to test the performance of the proposed solution. The algorithm performance and computational result obtained by the proposed heuristic for each data instance are presented. Then, the best solution obtained in this research is compared with the best-known solution from the literature research discussed. The gap between both solutions is calculated and analysed.

4.6.1 Computational Time

The total computing time used to generate the initial solution is presented in Table 4.4. In the meantime, the total compute time spent in VNS to reach an optimum solution is recorded. The average time used to generate one set of feasible solutions is calculated by dividing the total time spent by the total looping number, which takes place in the improvement stage before it reaches the optimum.

Table 4.4: Lists of compute time of VRPB

Dataset	Customer Size	Number of Vehicles	Initial Solution (s)	VNS (hrs)	Total compute time (hrs)	Number of feasible solution (loop)	Average runtime for each looping (s)
A1	25	8	56	2.998	3.014	434543	0.0249
A2	25	5	55	0.877	0.892	178667	0.0179
A3	25	4	34	2.414	2.424	450937	0.0194
A4	25	3	33	2.053	2.062	281626	0.0264
B1	30	7	141	0.472	0.511	10743	0.1712
B2	30	5	61	0.544	0.561	94808	0.0213
B3	30	3	15	0.211	0.216	40695	0.0191
C1	40	7	130	2.084	2.120	399482	0.0191
C2	40	5	76	1.484	1.506	278674	0.0194
C3	40	5	57	4.030	4.046	890977	0.0163

Table 4.4: continued

C4	40	4	56	7.870	7.885	1381268	0.0206
D1	38	12	125	4.080	4.114	289065	0.0512
D2	38	11	114	3.045	3.077	522696	0.0212
D3	38	7	83	10.21	10.24	716549	0.0514
D4	38	5	77	5.293	5.314	1045095	0.0183
E1	45	7	100	8.134	8.162	311844	0.0942
E2	45	4	77	11.98	12.00	1187229	0.0364
E3	45	4	146	6.149	6.189	277613	0.0802
K1	113	10	898	22.75	23.00	6064912	0.0137
K2	113	8	802	25.23	25.45	5998831	0.0153
K3	113	9	756	20.37	20.58	5095408	0.0145
K4	113	7	732	24.93	25.131	5766870	0.0157

According to Table 4.4, the overall trend shows that the total compute time increases when the size of the customer or the number of vehicles increases. The time used to get an optimal solution is increased exponentially from Problem Set A to Problem Set K. Based on the table, Problem Sets A, B, C, and D, which with a customer size below 40, are mostly able to complete the process within 10 hours. When the size of the customer increased to 40, the longest time needed became 12 hours. Then, for Problem Set K, the time required is approximately one day.

Based on the results, VNS needed a longer time in the neighbourhood restructuring process for a large data instance. The total compute time for Problem Set B is the shortest, possibly because the customer size of Problem Set B is small and the number of vehicles for Set B is lesser. From the Table 4.4, Problem Set B has fewer vehicles than Problem Sets A, C, and D. Thus, the time needed for VNS to restructure the neighbourhood solution will be shorter than other datasets.

Besides, although the average runtime for Problem Set A and Problem Set K are close to each other, which only took 0.02 seconds per loop, their total time to reach the optimal solutions shows a big gap. This is because the data size for Problem Sets A, B, C,

D, and E are comparatively smaller. Thus, the time needed to serve all customers until the optimum is shorter compared to the total time needed for Problem Set K with 113 customers.

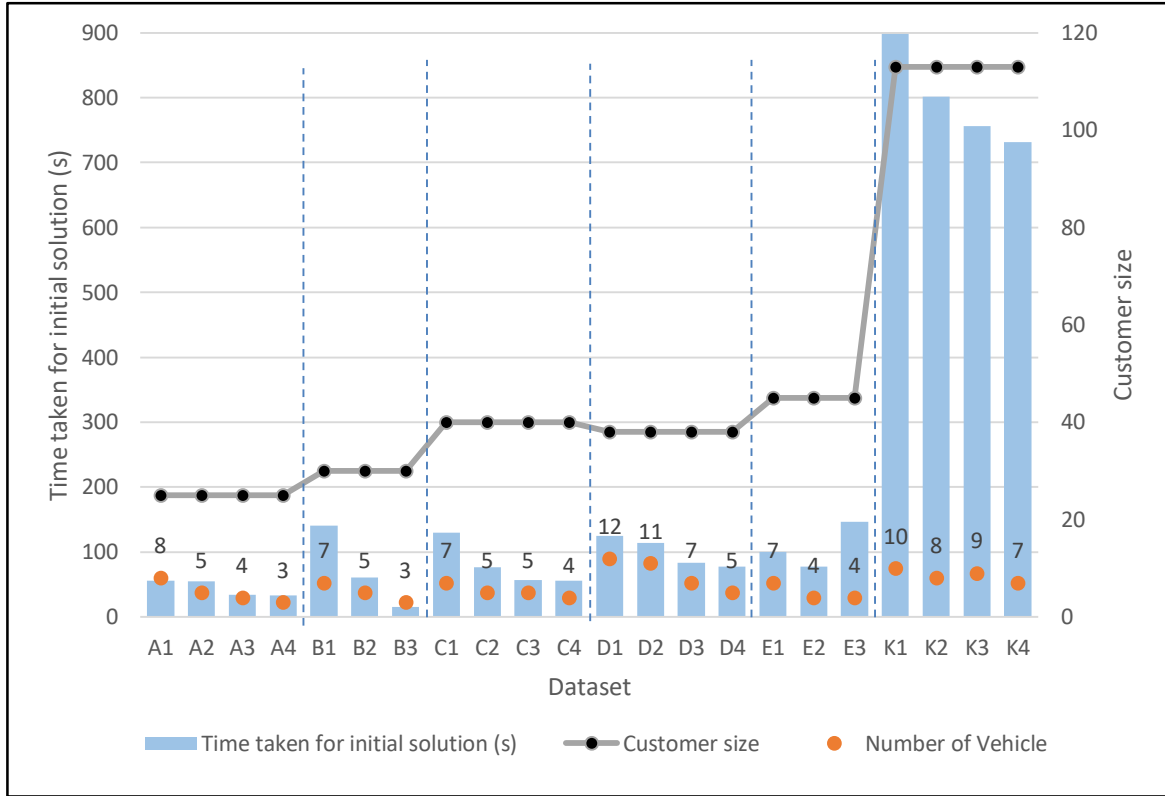


Figure 4.22: Time taken of initial solution

Figure 4.22 above represents the total time consumed to generate the initial solution and the number of vehicles used in each tested instance. According to Figure 4.22, the total time taken in the initial stage shows an increment when the number of vehicles increases. As observed from the group perspective for Problem Sets A, B, C, D and K, the time taken to generate the initial solution decreased as the total number of predefined vehicles decreased. The outcomes have implied within the instances with the same data size that the time used to form the initial solution with a higher vehicle usage is longer than that with fewer vehicles. Figure 4.23 below represents the total time consumed in VNS.

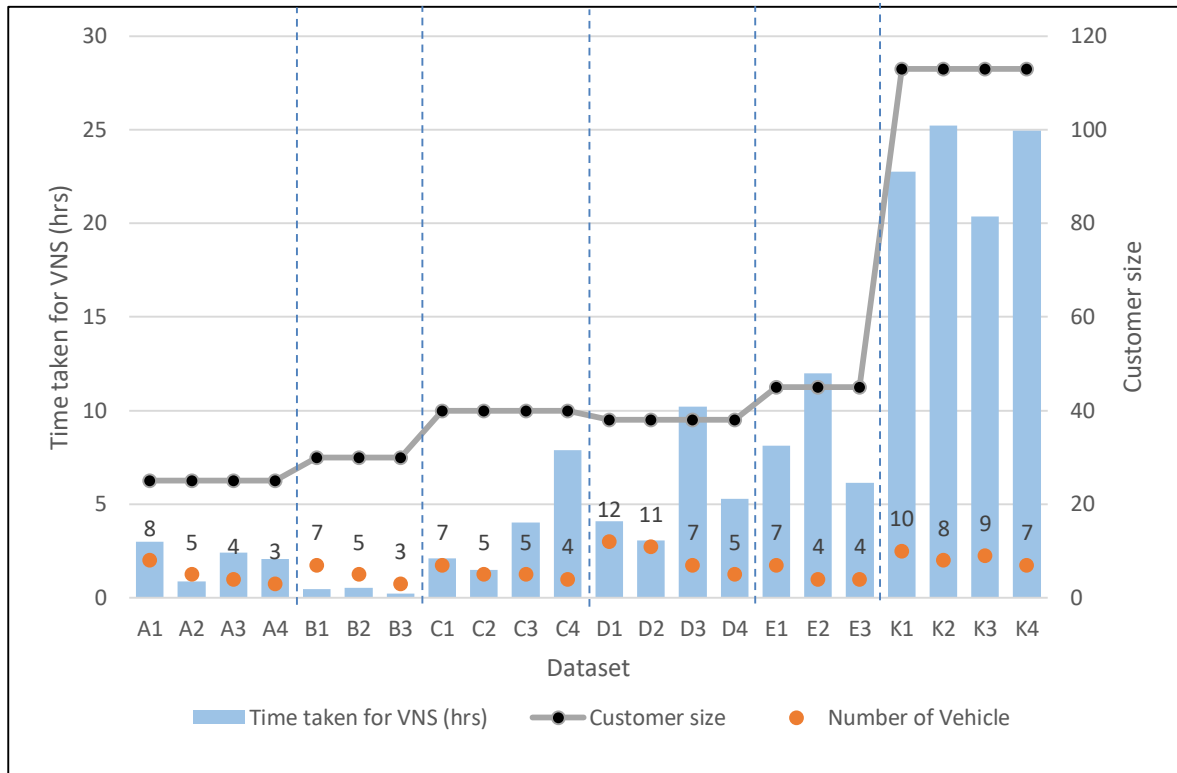


Figure 4.23: Time Taken of VNS

According to Figure 4.23, time taken for VNS to reach optimum is not consistent. This is due to the number of loops executed in VNS depending on the improvement of the solution. In VNS, the improvement procedure is repeatedly performed until no improvement exists. The number of loops and feasible solutions that can be found and formed is uncontrollable. The local search procedure can only be ended when no further improvement can be found. Moreover, in VNS, the shaking method is applied randomly. The number of times to execute the shaking process is unpredictable. All of these factors could lead to a longer processing time.

In contrast, A2, B1, C1, C2, D1, D2, E1, and K1 can reach optimum with a shorter executing time even with the large number of vehicles used. These have presented that local

search and shaking methods has the possibility to obtain a better solution by escaping from the trap of local optima in a short time.

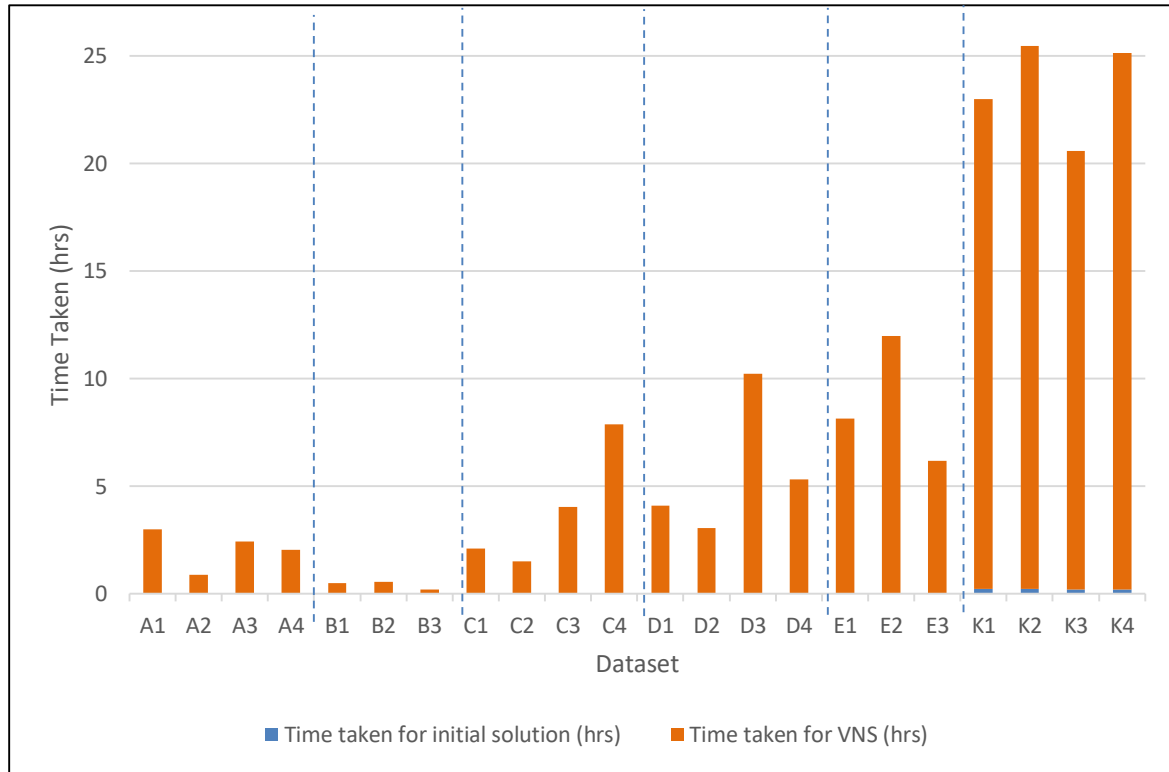


Figure 4.24: Total compute time of VRPB

Furthermore, Figure 4.24 above shows that the compute time of the VNS is longer than the priority rules heuristic. According to the data, the total time used to execute the improvement stage has occupied more than 95% of the overall time consumed. The results indicated that a meta-heuristic solution is more time-consuming than a heuristic solution. VNS needs to undergo a few loops until all possibilities are tried, and the global optimum solution is reached.

Generally, the results indicate that the total time used to retrieve the final solution for a smaller customer data size is shorter than the benchmark data with a large scale size. Moreover, the total time taken to reach a final solution for the benchmark data with a large vehicle number is shorter than the dataset with fewer vehicle numbers. In VRPB, the number

of predefined vehicles must be used to complete the task. Therefore, the number of vehicles represented the number of routes used to serve the customers. Thus, fewer vehicles denoted that the number of routes is lesser; the fewer the number of routes, the fewer the number of neighbourhood structures that can be formed and the shorter the time required.

Lastly, most benchmark datasets have a similar reading on the average looping time. The readings indicated that the proposed algorithm could generate a feasible solution with a consistent speed, which is around 0.036 seconds.

4.6.2 Computational Result

The selected benchmark datasets consist of different ratios of linehaul customers to backhaul customers, which are 50:50, 33:67, 67:33, 80:20 and 20:80. Table 4.5 shows the characteristic of the sub-problem set used in this research. Benchmark datasets with different customer ratios are used to test the robustness of the proposed method under different scenario types with varying amounts of customer types.

Table 4.5: Characteristics of each sub-problem sets (customer ratio)

Problem Set	Customer Size	Number of Linehaul	Number of Backhaul	Percentage of linehaul (%)	Percentage of backhaul (%)
A	25	5	20	20	80
B	30	10	20	33	67
C	40	20	20	50	50
D	38	30	8	80	20
E	45	30	15	67	33
K	113	75	38	67	33

Sub-problems A and D are the sets that hold the biggest gap between the number of linehaul and backhaul customers. Set A and D groups under the same ratio of 20:80 with the customers' sizes are 25 and 38, respectively. Although sets A and D are categories under the same ratio group, the major customer group in set A is backhaul customers. In contrast, the

major customer group in set D is linehaul customers. Since they have the same ratio, sets A and D can be used to test the performance of the proposed method in solving the following scenarios:

Scenario I: Number of backhaul customers is four times bigger than the number of linehaul customers

Scenario II: Number of linehaul customers is four times larger than the backhaul customers

Besides, sub-problem sets B, E, and K have the customer with a ratio of 33:67. The diversity between the two customer types of set B, E and K is relatively smaller compared to sets A and D. In sub-problem B, the major customer is backhaul customers. In contrast, in sub-problem E and K, the number of linehaul customers is greater than backhaul customers. Hence, the scenarios can be compared as follows:

Scenario III: Number of backhauls customers is twice of linehaul customers

Scenario IV: Number of linehaul customers is twice of backhaul customers

Lastly, the customer ratio for sub-problem set C is 50:50. Sub-problem set C has 40 customers, and the number of linehaul customers is exactly equal to the number of backhaul customers, which are 20 customers for each. The last scenario that can be investigated is:

Scenario V: Number of backhaul customers is equal to the number of linehaul customers

4.6.2.1 Analysis of the Final Solution

The computational analysis of the final solution of each benchmark data is carried out in this section. It included the improvement analysis for the solution obtained by the priority rule heuristic algorithm and the final output obtained by the VNS method. Besides, the final solutions obtained in this research are compared with the best-known solution found

in the past literature study. The improvement percentage rate among the solutions is calculated and analysed. In this section, the recorded and calculated values are rounded to the nearest two decimal places.

Table 4.6 below illustrates the outputs of the computational results of the proposed algorithm. In contrast, the values for the computational result are plotted using bar chart graph format and shown in Figure 4.25. The system outputs for the optimum solution (final result obtained) are appended in Appendix C, Table A1.

Table 4.6: Initial and optimum solutions obtained for VRPB

Dataset	Total Travelled Distance		IMP (%)	Average IMP (%)
	Initial Solution	Optimum Solution (Final Result Obtained)		
A1	414181	229886	-44.50	-55.69
A2	397237	180119	-54.66	
A3	420513	163405	-61.14	
A4	415037	155796	-62.46	
B1	270218	243860	-9.75	-42.29
B2	444467	198048	-55.44	
B3	441865	169372	-61.67	
C1	615416	250557	-59.29	-64.94
C2	590353	215020	-63.58	
C3	621872	199346	-67.94	
C4	628917	195367	-68.94	
D1	660491	322740	-51.14	-54.22
D2	612869	316709	-48.32	
D3	535760	239479	-55.30	
D4	543500	205832	-62.13	
E1	647168	243469	-62.38	-65.18
E2	612442	212263	-65.34	
E3	648756	208837	-67.81	
K1	1742066	429667	-75.34	-76.09
K2	1697729	407477	-76.00	
K3	1715204	416642	-75.71	
K4	1667285	378513	-77.30	

The total distance for the initial solution and VNS solution of each data instance are shown in Table 4.6 above. The initial solution was the total travelled distance of the feasible

solution generated in the initial stage by applying the simple priority rule heuristic. However, the final solution is the total distance travelled obtained after applying the VNS algorithms. The improvement percentage, IMP, between the initial and final solutions is calculated by Equation 4.2 (Polat, 2017). IMP is calculated to determine the distance improvement of the initial solution compared to the final outcome generated by VNS. The increase in the average IMP value represented the reduction percentage of the total travelled distance.

$$IMP = \frac{(Optimum\ solution - Initial\ solution)}{Initial\ solution} \times 100 \quad \text{Equation 4.2}$$

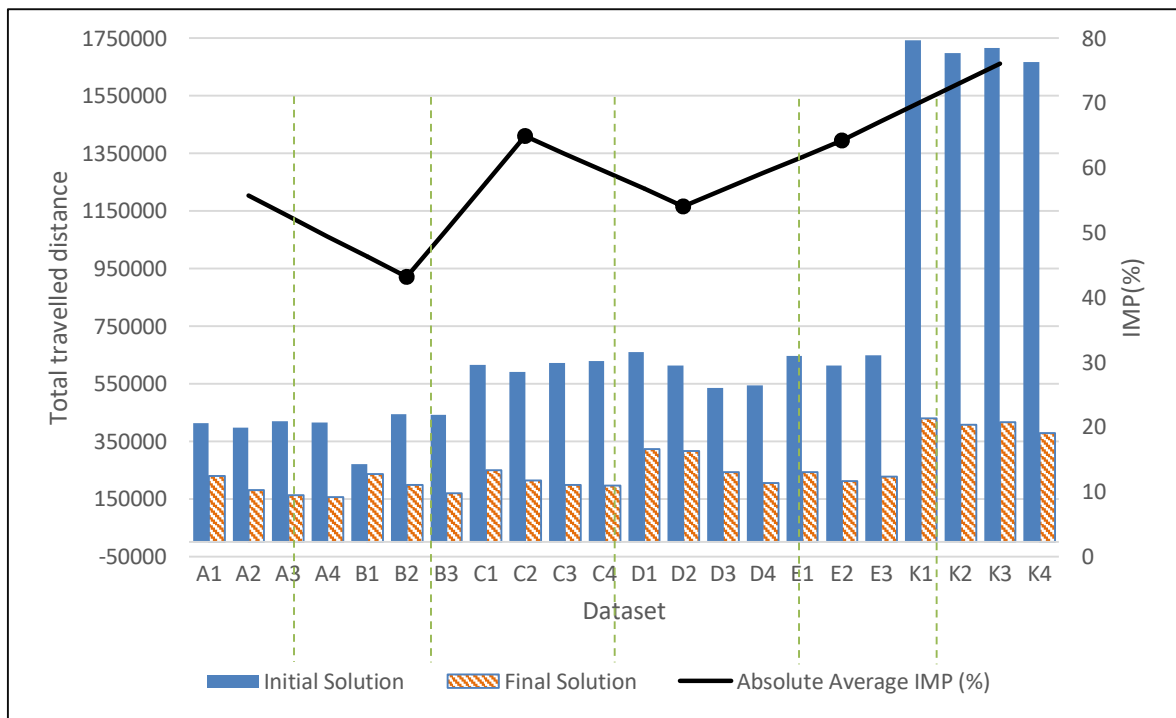


Figure 4.25: Total travelled distance of initial and optimum solution

There is a significant improvement in the results before and after the VNS is applied. As observed from Figure 4.25, the final total travelled distance decreases extremely after undergoing the VNS. The analytic result shows that the highest improvement rate after VNS has been applied is 77.30%, which results obtained in K4, and the lowest rate, 9.75%, is

obtained from B1. The overall result has shown that the proposed method provided an increment of at least 9.75% compared to the initial solution.

According to the average IMP values, Problem Set K shows the highest average IMP with a value of 76.09%. The great improvement is due to the simplicity of the proposed priority heuristic. The vehicle will serve as many customers as before it reaches its maximum capacity. Therefore, it produces a long list of trips before it starts hauling back to the depot. Besides, the increase in the data size has increased the number of possible offspring patterns found in the improvement stage. Hence, a small shift in the position among the customer can cause a bigger chance to minimise the total travel distance.

On the other hand, Problem Set K shows a superiority improvement among the datasets, followed by Problem Sets C and E, which show a substantial increment of average IMP of 64.94% and 65.18%, respectively. The results indicated that the proposed VNS algorithm is good at solving Scenario IV (Problem Sets E and K), in which the scenario has an amount of linehaul and backhaul customers with a ratio of 67:33. Moreover, the results also show the effectiveness of the proposed algorithm in solving Scenario V (Problem Set C), in which the number of one of the customer's types is exactly equal to each other.

However, Problem Set B has an IMP value of only 42.29%, which is the problem set with the lowest increment. Although Problem Sets B, E, and K all with the customer with the ratio of 67:33, the average IMP value for Problem Set B did not show a degree of satisfaction as Problem Sets E and K. Differ from Problem Sets E and K, Problem Set B has categorised under Scenario III which the backhaul customers are twice than the linehaul customers. This indicated that the proposed algorithm is less effective in solving the scenario with more backhaul customers.

Furthermore, as shown in Table 4.6, even though Problem Set B shows the least increment in average IMP value, the total distance of the initial solution for subset B1 is very close to the optimum solution. Therefore, the gap between the initial and final solutions for B1 is small. This denotes that the initial solution generated by the proposed priority rules heuristic is close to the final optimum solution. Thus, the proposed simple priority rule heuristic algorithm is able to provide a fine solution for a small-scale problem.

In summary, the overall result shows significant improvement from the initial solution. The results reflect that the total distance travelled can be minimised in the shake and implement steps. The optimum solution has a much shorter total distance travelled than the initial solution. This is because the repeated local search process in the VNS structure can ensure a local minimum is reached. The analytic result from Table 4.6 and Figure 4.25 shows that VNS can help the simple heuristic escape from the trap at the local optimum.

4.6.2.2 Analysis of Final Solution with Literature's Best-Known Solution

This section compares the final solution obtained with the current best solution obtained from past literature studies. The summary of the list of referred journals is stated in Table 4.7 (Osman & Wassan, 2002; Gajpal & Abad, 2009; Anbuudayasankar et al., 2014; Wassan et.al., 2017; Koç & Laporte, 2018).

These five chosen papers are the key papers many VRPB researchers referred to as references. The best-known solution for the selected 22 set benchmark datasets, which Osman and Wassan published in the year 2002, has been referred to and used to compare as the best-known until the present. The best-known solutions retrieved from these five journals are exactly the same. It means that all the authors from these five papers have proposed different techniques but achieved the same minimum total travelled distance, which is the

best-known solution here. Thus, the solutions can be accepted as the finest solution generated in these ten years. The list of best-known solutions has been listed in Table 4.8.

Table 4.7: Summary of journals for Best-Known solution

Year	Title	Journal	Author
2002	The multiple trip vehicle routing problem with backhauls: Formulation and a two-level variable neighbourhood search	Computers & Operations Research	Osman & Wassan
2009	Multi-ant colony system (MACS) for a vehicle routing problem with backhauls	European Journal of Operational Research	Gajpal & Abad
2014	Models for practical routing problems in logistics	Springer International Pu.	Anbuudayasankar, Ganesh & Mohapatra
2017	The multiple trip vehicle routing problem with backhauls: Formulation and a two-level variable neighbourhood search	Computers & Operations Research	Wassan et.al
2018	Vehicle routing with backhauls: Review and research perspectives	Computers & Operations Research	Koç & Laporte

In order to evaluate the proposed methodology, the Relative Percentage Deviation, RD , is computed for each solution obtained (Anbuudayasankar et al., 2014). Therefore, equation 4.3 is the equation of RD used in this section.

$$RD = \frac{(\text{Heuristic solution} - \text{Best known solution})}{\text{Best known solution}} \times 100 \quad \text{Equation 4.3}$$

Table 4.8 below shows the total distance of the best-known solution and the final result obtained in this research. RD value is calculated by Equation 4.3, and then an average

RD is calculated based on the set grouping stated in Table 4.1. The result is represented in Figure 4.26.

Table 4.8: Lists of Best-Known result and final obtained result for VRPB

Dataset	Number of Vehicles	Total Travelled Distance		RD (%)	Average RD (%)
		Best-Known Solution (*)	Final Result Obtained		
A1	8	229886	229886	0	0
A2	5	180119	180119	0	
A3	4	163405	163405	0	
A4	3	155796	155796	0	
B1	7	239080	243860	2.00	0.67
B2	5	198048	198048	0	
B3	3	169372	169372	0	
C1	7	250557	250557	0	0
C2	5	215020	215020	0	
C3	5	199346	199346	0	
C4	4	195367	195367	0	
D1	12	322530	322740	0.07	0.02
D2	11	316709	316709	0	
D3	7	239479	239479	0	
D4	5	205832	205832	0	
E1	7	238880	243469	1.92	0.99
E2	4	212263	212263	0	
E3	4	206659	208837	1.05	
K1	10	394637	429667	8.88	10.23
K2	8	362360	407477	12.45	
K3	9	365693	416642	13.93	
K4	7	358308	378513	5.64	

(*) The best-known solution is referred to from journal papers listed in Table 4.7.

Based on Table 4.8, Benchmark dataset A1, A2, A3, A4, B2, B3, C1, C2, C3, C4, D2, D3, D4 and E2 has reached the optimum solution. Besides, the *RD* for datasets B1, D1, E1 and E3 is less than 5%. This represents that the value of the final solution obtained is very close to the best-known published solution.

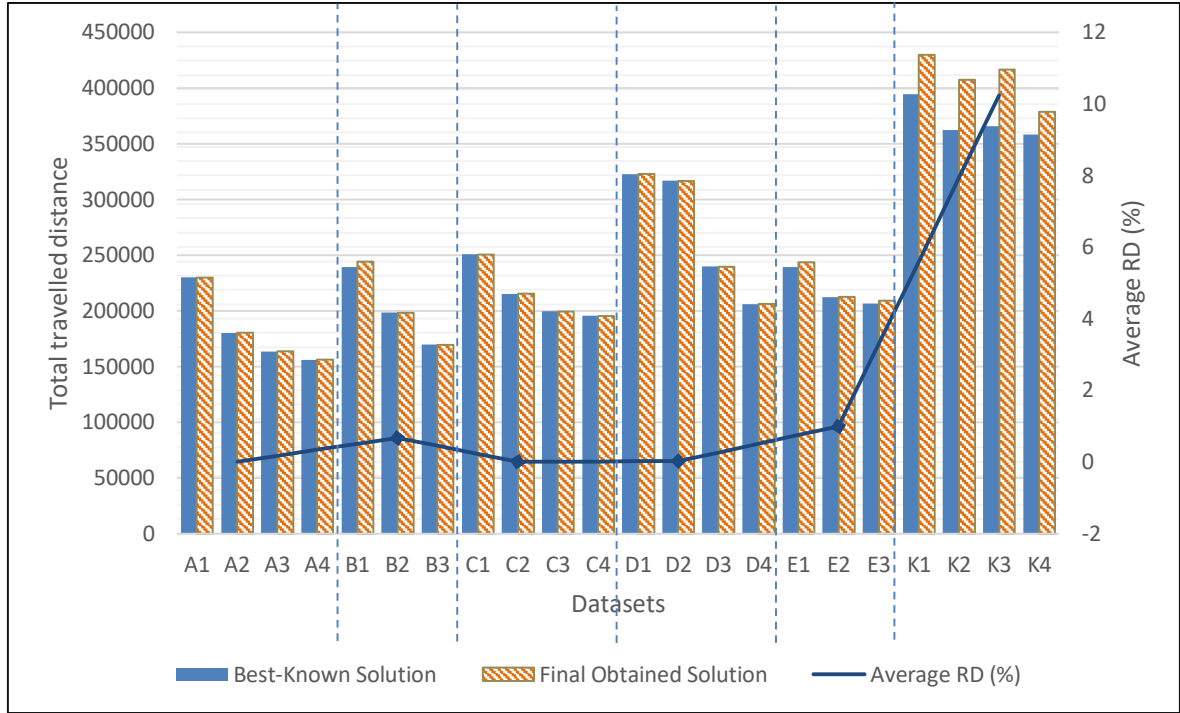


Figure 4.26: Total travelled distance of Best-Known and final obtained solutions

The bar chart shown in Figure 4.26 shows only a slight discrepancy between the solution obtained in this research and the best-known solution. The chart has represented that the final solutions obtained in this research are equal to or closer to the best-known solutions.

According to Figure 4.26, Problem Set A, B, C, D and E with a small-scale problem have lower average *RD* values, whereas the average *RD* for Problem Set K is the highest. The result represents that the proposed algorithm is better at solving, Problem Sets A and C, the problem sets with 25 and 40 customers. For the data with a custom size of equal and fewer than 45 customers, an average *RD* of less than 1% is obtained. The average *RD* value increases to 10.23% when the customer size increases to 113. The gap in *RD* increases when the number of customers increases. The trends represent that the proposed algorithm is less effective in solving a huge-scale solution.

In short, 14 datasets obtained the same solution as the best-known solution, 4 datasets had *RD* values that were less than 5%, 2 datasets with *RD* values that fell between 5% to 10%, and only 2 out of the total 22 sets of them have the differences which are more than 10%. The largest gap between the two solutions is 13.93%. The overall data has shown that the proposed method is able to compete with other methods.

4.7 Conclusion

This chapter compares the best solution obtained by the proposed algorithm with the best-known solution found in the past literature study. The computational time shows that the heuristic method can easily reach a feasible solution in a short time compared to VNS. The average looping time of around 0.036 seconds per looping shows the consistency of the proposed VNS algorithm in solving various data with different data sizes and customer types.

Besides, the computational results have shown the capability of the proposed algorithm in solving the VRPB model. A VNS is able to improve the feasible solution that is generated in the initial stage. A significant improvement is found before and after a VNS application. Moreover, the final optimum solution obtained by VNS has proven that the proposed algorithm is as effective as other methods. The final results obtained in this research are the same or closer to the best-known solution. Although some of the results obtained are not good enough compared to the current best research result, the proposed method is able to improve the initial solution.

CHAPTER 5

MIXED VEHICLE ROUTING PROBLEM WITH BACKHAUL

5.1 Introduction

In this chapter, an extension study is conducted to test the performance of the proposed Variable Neighbourhood Search (VNS) algorithm. A practical pickup and delivery scenario simulation is studied, integrating both pickup and delivery with the routing planning. This study aims to examine the impact of customer-type priority on the performance of the proposed algorithm.

5.2 Model of m-VRPB

The mixed Vehicle Routing Problem with Backhaul (VRPB) (m-VRPB) is an extended study conducted to observe the impact of routing arrangement on the final solution. Here, m-VRPB is studying using the same benchmark data and problem background. However, the vehicle can serve linehaul and backhaul customers alternately in the model m-VRPB. Besides, m-VRPB allows the route with only backhaul customers.

In this chapter, constraint (vii) and constraint (viii), stated in Section 3.3, are relaxed. However, the rest of the constraints, as per discussed in Chapter 3, remain unchanged (as referred Chapter 3, Section 3.3 Constraints). Table 5.1 below shows the comparison of the constraints for VRPB and m-VRPB.

Table 5.1: Lists of constraints of VRPB and m-VRPB

Constraints	VRPB	m-VRPB
Vehicles depart and end at the origin of every route	Applied	Applied
Each customer can only be served once	Applied	Applied

Table 5.1: continued

The customer is visited by exactly one vehicle only	Applied	Applied
The demand of customers must be fulfilled	Applied	Applied
All vehicles will have the same predefined capacity	Applied	Applied
The total capacity cannot exceed the maximum capacity of the vehicle	Applied	Applied
Routes are not allowed to involve only backhaul customer	Applied	Relaxed*
Linehaul customers are served before any backhaul customer	Applied	Relaxed*

In summary, m-VRPB is generated based on the new constraints stated in the table above. m-VRPB is investigating using the same benchmark data but slight differences in the problem restriction. The vehicle can serve linehaul customers and backhaul customers alternately in a route. The route design can violate the rule that all linehaul customers should serve before the backhaul customer. Meanwhile, the customer to be assigned next is decided based on the quantity of customer demand and the remaining capacity of the vehicle, not the customer type. Therefore, the route with only backhaul customers is acceptable in the extended study. m-VRPB has expected to have more liberty on the number of feasible routes compared to VRPB as it is carried out to simulate the practical environment by removing some of the restrictions from the VRPB.

This chapter applies a new priority rule in the initial stage. Furthermore, one new local search procedure and one new shaking method are added to the VNS improvement process.

5.2.1 Initial Solution for m-VRPB

A new simple priority rule heuristic for m-VRPB is proposed. The subsequence customers in m-VRPB are sorted based on the distance, and the type of customer is

neglected. In the new simple priority rule heuristic, the customer with the shortest distance is picked to be served as long as no capacity violation happens.

First, as discussed in Section 4.4, the number of routes for each dataset is created based on the predefined vehicle number, and also, the depot is the first and last node of the route. Then, the customer with the closest customer distance is selected as the next subsequence node. The selected node can be either a linehaul or a backhaul customer. Figure 5.1 below shows the new sorting structure of each route for m-VRPB.

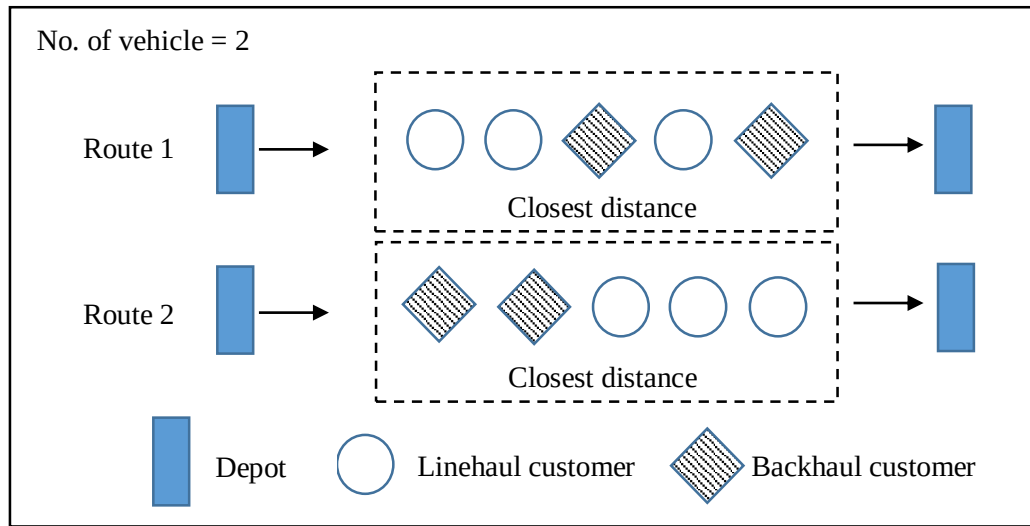


Figure 5.1: Route structure of m-VRPB

As shown in Figure 5.1, Routes 1 and 2 represented the sample of feasible routes in m-VRPB. Differing from VRPB (Figure 4.4), the sequence of the customer can start from a backhaul customer and end with a linehaul customer. The time used to rearrange the load on the vehicle is ignored in m-VRPB.

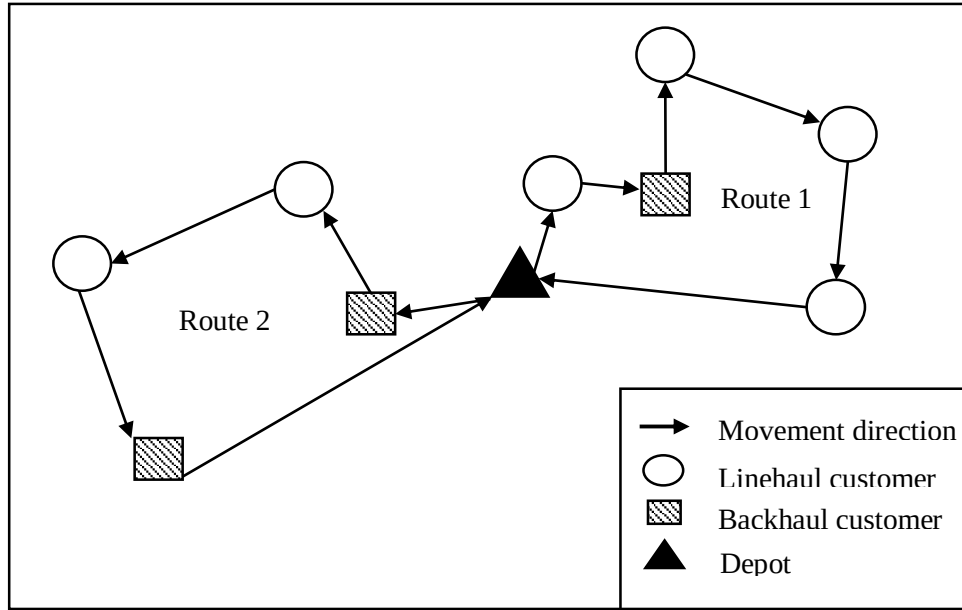


Figure 5.2: Vehicle travelling movement of m-VRPB

Figure 5.2 represents the vehicle movement of m-VRPB. The route is generated using a list of new priority rule heuristics. The priority rule is executed based on the following consideration

- i. The closest unassigned customer is the customer to serve. Check on the total loaded capacity size.
 - If the total capacity is under the capacity limit, set the customer as served and searched for the next node.
 - Else, seek the second nearest linehaul customer from the last node and check the total capacity loaded.

In short, any customers with the shortest travel distance from the depot will be considered and selected. Figure 5.3 below shows the customer sorting process in m-VRPB for each route based on the data distancing.

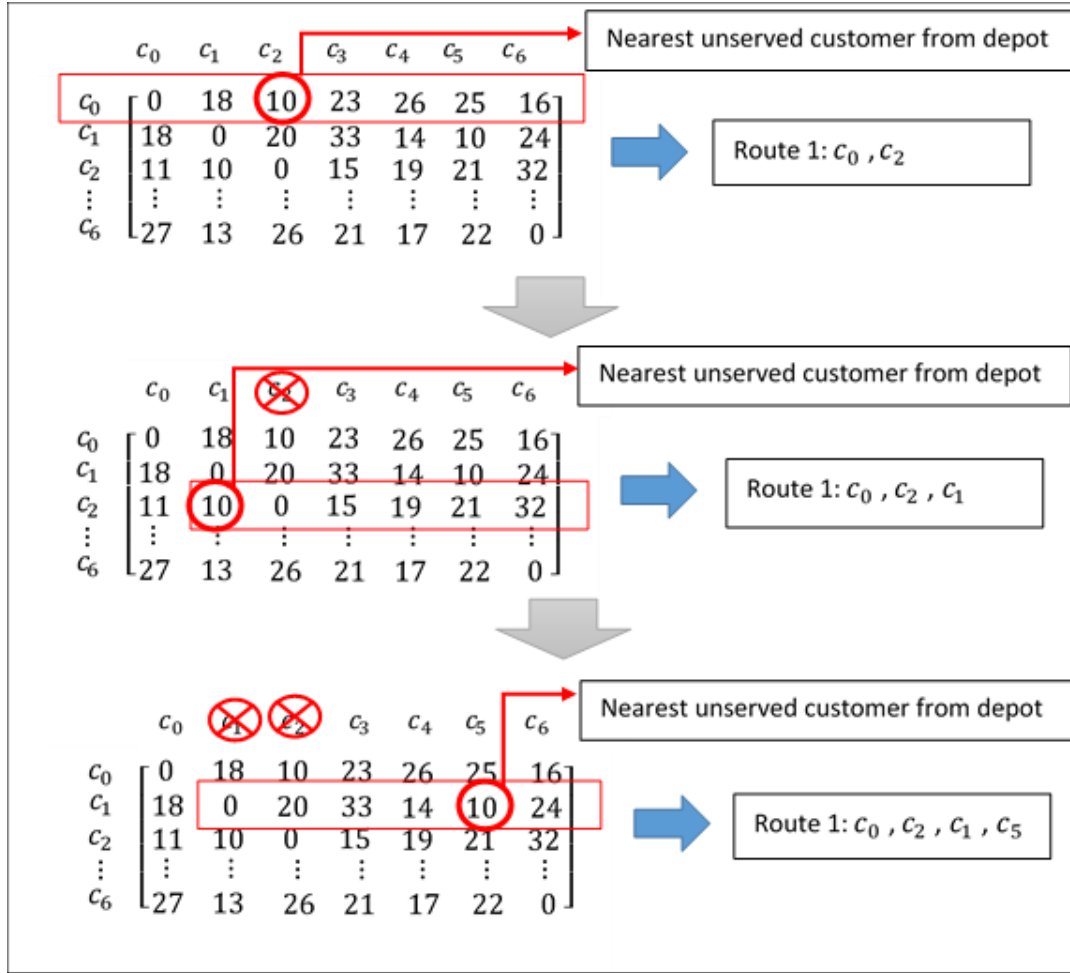


Figure 5.3: Sorting process of m-VRPB

Then, once the node with the shortest path is determined, the vehicle's total capacity is calculated and checked. Again, suppose the vehicle has exceeded the maximum capacity. In that case, the customer is skipped, and then the subsequent customer with a shorter distance is considered and becomes the next subsequent node.

The load delivered to a linehaul customer is loaded on the vehicle at the depot, whereas the vehicle will collect the load from the backhaul customers and carry it back to the depot. In m-VRPB, the fluctuating loads on the vehicle are the most challenging part.

A capacity-checking algorithm is proposed to check the total load carried on the vehicle. Algorithm 5.1 shows the algorithm used to check and update the vehicle's capacity.

The algorithm has ensured the capacity left on the spot the vehicle arrived at customer i is enough to deliver or collect the total demand, d_i of customer i . The free space of the vehicle and the remaining loaded quantity, L are checked from time to time after every serving process. The maximum occupied capacity, C is updated when the highest sum of occupied capacity is found.

Algorithm 5.1: Customer's Capacity Checking

```

 $n$  = Total number of customer
 $T$  = Customer type
 $C$  = The most capacity occupied in the vehicle
 $L$  = Total load carried by vehicle from the depot

 $C = 0$ 
 $L = 0$ 
FOR  $i = 1$  to  $n$ 
    IF( $T_i = linehaul$ )
         $C_i = C_{i-1} + d_i$ 
         $L_i = L_{i-1} + d_i$ 
         $C = C_i$ 
    ELSE
         $C_i = C_{i-1} + d_i$ 
        IF( $C_i > C$ ) THEN  $C = C_i$ 
    ENDIF
ENDIF
ENDFOR

```

When a limited capacity is reached, the vehicle will return to the depot, and a new vehicle will continue to serve the remaining customers. The process will be terminated if all customers have been served exactly once.

Lastly, the total travel distance is calculated based on the feasible solution generated. The data flow diagram of the initial stage for m-VRPB-m is shown in Figure 5.4. Step A represents the process flow to form the initial solution structure for m-VRPB. The number of routes of the generated outcome depends on the number of vehicles assigned.

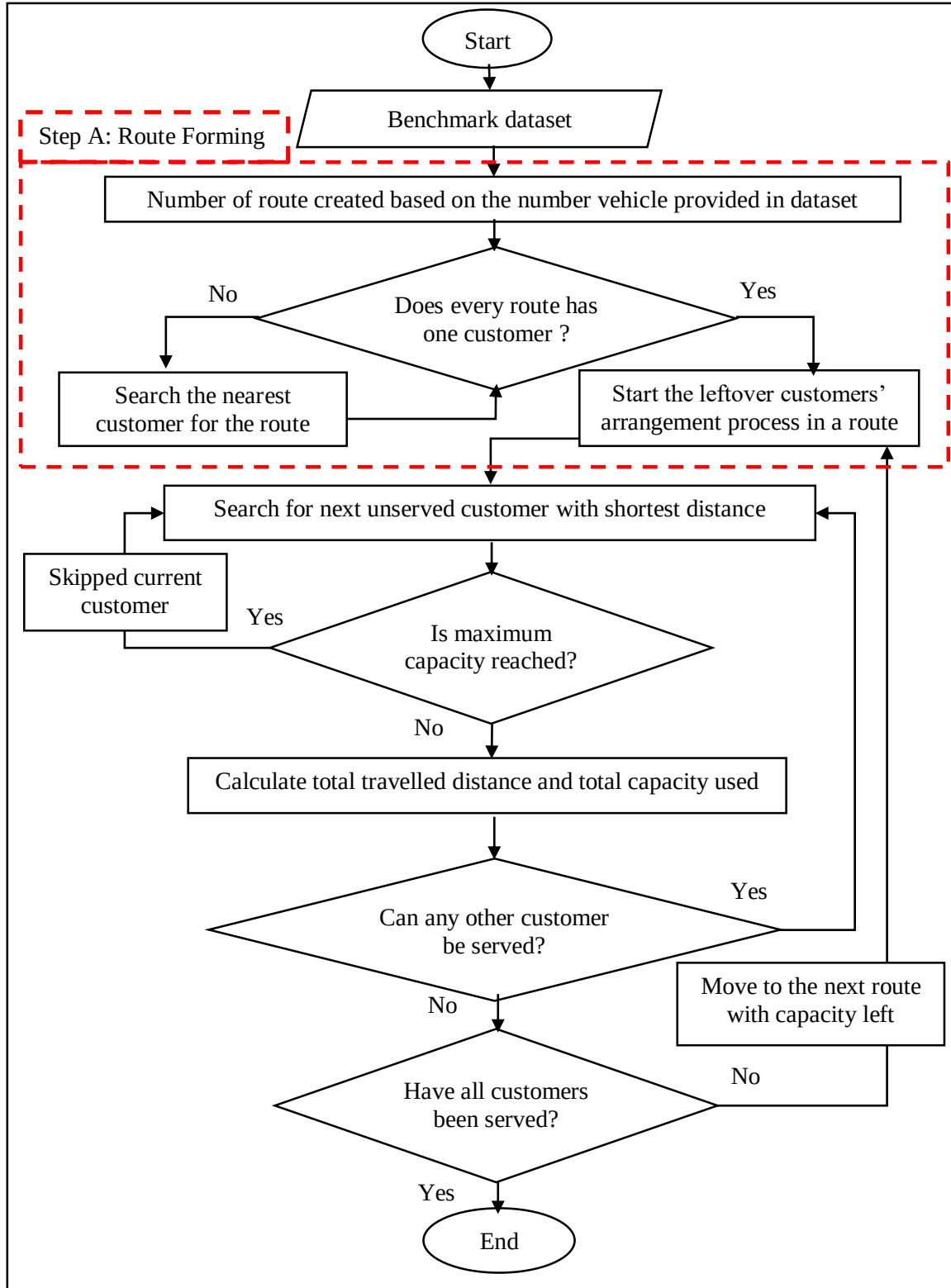


Figure 5.4: Data flow diagram of initial solution of m-VRPB

Figure 5.5 below shows the initial solution for Dataset A1 generated by the flow of the new priority rule heuristic, shown in Figure 5.4. Dataset A1 in m-VRPB has the same predefined number in vehicles and the maximum capacity as Dataset A1 in VRPB. Also, the columns in Figure 5.5 have the same definition as explained in Figure 4.10. Unlike the solution routes presented in Figure 4.10, the backhaul customers in Figure 5.5 can be served in any position in the route.

route_no	route_list	total_travelled_distance	total_delivery_amount	total_pickup_amount
1	0 25 12 22 3 21 18 0	48875.9273	1168	1301
2	0 6 10 23 24 0	43728.6289	1446	1239
3	0 13 2 16 0	29735.829	1431	0
4	0 4 19 15 8 0	50741.6543	1531	0
5	0 17 1 0	40239.2771	1269	0
6	0 11 9 0	32109.9946	1286	0
7	0 5 7 20 0	34836.8291	1243	0
8	0 14 0	35469.0184	675	0

Figure 5.5: Initial solution for Dataset A1 of m-VRPB

5.2.2 VNS for m-VRPB

Note that m-VRPB has relaxed some constraints and has more flexibility compared to VRPB. In m-VRPB, swapping any nodes in VNS's operator is allowed. Thus, a list of new local search procedures and shaking methods enhances the VNS improvement process.

To test the impact of the solution and final result yields before and after the relaxation of the consideration of customer type and serving sequence, the proposed VRPB model discussed in Chapter 4 is amended. Below are the local search techniques and shaking methods proposed in this research.

5.2.2.1 Local Search Techniques for m-VRPB

i. Local 1-to-1 Swap

The location of a customer (linehaul or backhaul) is swapped iteratively with another customer (linehaul or backhaul) from a different route until all feasible structures are tested. The steps of the local 1-to-1 swap are shown in Figure 5.6. For example, Loops 1 and 2 are two different neighbourhood structures. In Loop 1, the first node in Route 2 is exchanged with the first node in Route 1, whereas the same node in Route 2 is exchanged with the second node in Route 1 to form another feasible structure.

The Local 1-to-1 Swap for VRPB stated in Section 4.5.1.3, can only exchange among the node with the same customer type. However, in m-VRPB, the two nodes can be different customer types, as long as all constraints are fulfilled.

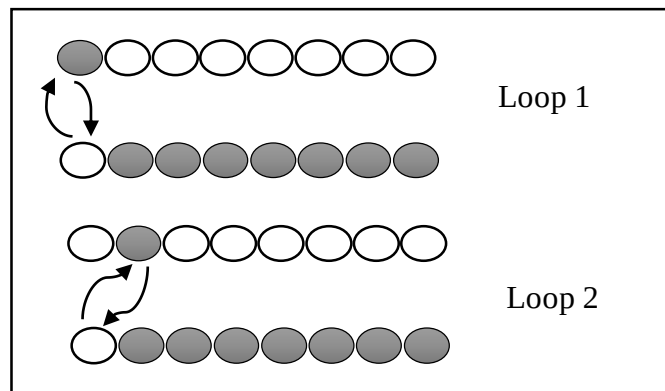


Figure 5.6: Local 1-to-1 Swap of m-VRPB

ii. Local 2-to-1 Swap

Two consecutively arranged customers (linehaul or backhaul) in a certain location are swapped iteratively with a customer (linehaul or backhaul) from a different route until all feasible structures are tested. The steps of the local 2-to-1 swap are shown in Figure 5.7.

Loops 1 and 2 represented two different neighbourhood structures after applying the Local 2-to-1 Swap. For example, in Loop 1, two nodes in Route 1 are swapped with a node in Route 2. The two nodes in the first route can be either the mix of linehaul and backhaul customers or both with the same customer type. The feasible swapping pattern for Local 2-to-1 Swap of m-VRPB increased as m-VRPB allowed the mixing of linehaul and backhaul customers in any route.

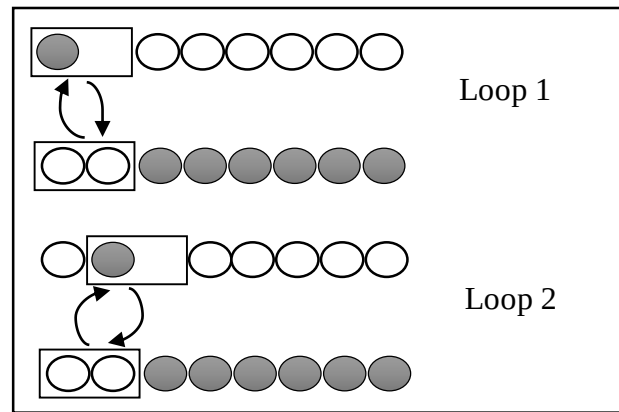


Figure 5.7: Local 2-to-1 Swap of m-VRPB

iii. Local 1-to-Many Swap

The location of a customer (linehaul or backhaul) is swapped iteratively with n customers (linehaul or backhaul) from the different route that is arranged side by side consecutively. The number of n depends on the cumulative capacity for n customers. The $n+1$ customer can be included if the cumulative capacity does not exceed the capacity limit. Other than that, the process is repeated iteratively until all feasible structures are tested. The local 1-to-Many swapping steps are shown in Figure 5.8.

Loop 1 represented the example when one node was selected to swap with three nodes from another route. Then, Loop 2 demonstrated the scenario in which one node is selected to be swapped with another 2 nodes from a different route.

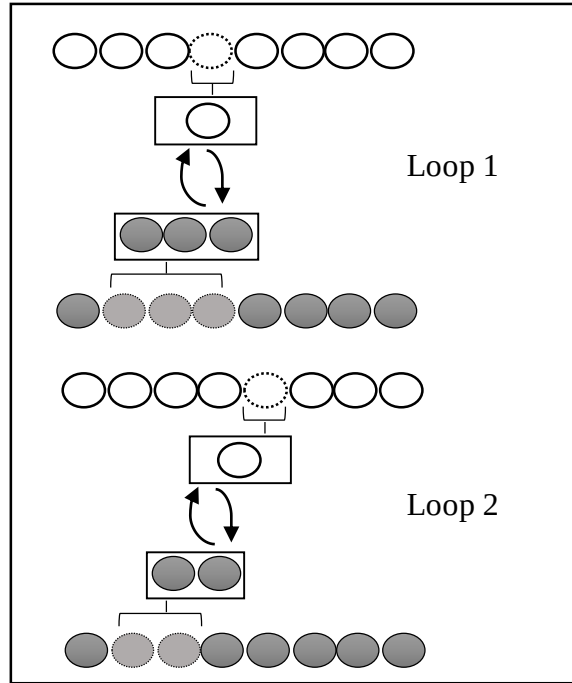


Figure 5.8: Local 1-to-Many Swap of m-VRPB

5.2.2.2 Shaking Methods for m-VRPB

i. Random 1-to-0 Swap

One customer (linehaul or backhaul) is selected at random from a randomly selected route and shifted to an alternative random route. Figure 5.9 represents the Random 1-to-0 swapping method.

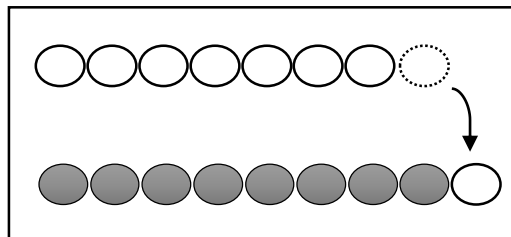


Figure 5.9: Random 1-to-0 Swap of m-VRPB

ii. Random 2-to-2 Swap

Two randomly picked customers (linehaul or backhaul) from the same randomly selected route are swapped with another two randomly picked customers (linehaul or

backhaul) from another route. These two customers can either be two customers selected from two different routes or the customers from the same route. Figure 5.10 denotes the random 2-to-2 swapping method.

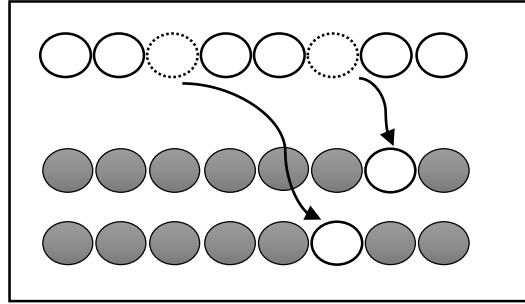


Figure 5.10: Random 2-to-2 Swap of m-VRPB

iii. Random 1-to-1 Swap

A randomly picked customer (linehaul or backhaul) from a randomly selected route is swapped with another one randomly picked customer (linehaul or backhaul) from another route. Figure 5.11 denotes the random 1-to-1 swapping method.

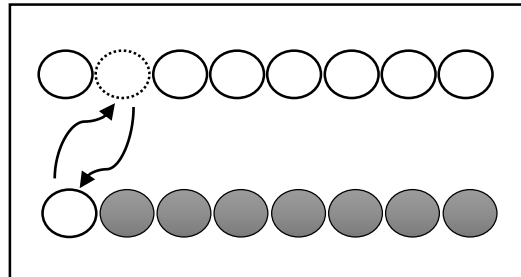


Figure 5.11: Random 1-to-1 Swap of m-VRPB

5.3 Result and Analysis for m-VRPB

m-VRPB is an extended model conducted to observe the impact of routing arrangement on the final solution in the classical VRPB model. The elimination of the constraints has affected the model structure of the VRPB. Thus, an amended algorithm with the new checking process is proposed to fit the new model tested.

m-VRPB is executed under the same benchmark data and problem background as the VRPB background used in this research. Moreover, since the proposed algorithm for VRPB has shown effectiveness in solving a small-scale dataset, only Problem Sets A, B, C, D, and E are tested in this section. However, some subtests from Problem Sets A, C, and D are eliminated in order to retain the uniqueness of each sub-dataset on the feature of the number of the vehicle and volume of the given capacity. For example, as shown in Table 4.2, the capacity size for datasets A3 and A4 equal 4050. Therefore, A4 is eliminated. Thus, benchmark datasets A4, C4, and D2 are removed since the capacity size is the same as A3, C3, and D1 respectively.

Besides, Problem Set K is excluded because the time required to solve Problem Set K is estimated to be longer than the time used in VRPB. The number of combinations of Problem Set K neighbourhood structures is expected to be greater than VRPB. The benchmark instances used to test m-VRPB are shown in Table 5.2 below.

Table 5.2: Benchmark dataset of m-VRPB

Dataset	Customer size	No. of linehaul	No. of backhaul	Number of vehicles	Capacity size of vehicle
A1	25	20	5	8	1550
A2	25	20	5	5	2550
A3	25	20	5	4	4050
B1	30	20	10	7	1600
B2	30	20	10	5	2600
B3	30	20	10	3	4000
C1	40	20	20	7	1800
C2	40	20	20	5	2600
C3	40	20	20	5	4150
D1	38	30	8	12	1700
D3	38	30	8	7	2750
D4	38	30	8	5	4075
E1	45	30	15	7	2650
E2	45	30	15	4	4300
E3	45	30	15	4	5225

5.3.1 Relaxed in Constraints

Classical VRPB needs to obey the constraints on a linehaul customer needs to serve before any backhaul customer and is not allowed to involve only the backhaul customer in a particular route. However, these constraints are relaxed in this section to test the capability of the proposed algorithm in solving a more practical environment.

In this section, m-VRPB is executed using the same datasets instance and problem background used by VRPB. Table 5.4 and Figure 5.12 below compare the best-known results retrieved from past journal studies with the final result obtained by m-VRPB in a free-serving sequence. The system outputs for the m-VRPB solutions are appended in Appendix C, Table A2.

Table 5.3: Lists of Best-Known solution and m-VRPB solution

Dataset	Total Travelled Distance		RD (%)
	Best-known for VRPB	m-VRPB solution	
A1	229886	228316	-0.68*
A2	180119	175412	-2.61*
A3	163405	167004	2.20
B1	239080	235243	-1.60*
B2	198048	198963	0.46
B3	169372	173516	2.45
C1	250557	251172	0.25
C2	215020	232799	8.27
C3	199346	210969	5.83
D1	322530	324155	0.50
D3	239479	246957	3.12
D4	205832	213681	3.81
E1	238880	250145	4.72
E2	212263	214644	1.12
E3	206659	228601	10.62

The total travelled is the distance of the best-known VRPB retrieved from the past journals (Refer to Chapter 4, Table 4.7: Summary of Literature Paper for Best-known Solution). However, the m-VRPB solution is the final outcome in this section. The Relative

Percentage Deviation (RD) value is calculated using Equation 4.2 (Refer to Chapter 4, Equation 4.2). From Table 5.3, the RD values of m-VRPB are lower than 11%. There are 3 out of 15 sets that have obtained a better result than the best-known solution, 9 datasets with an RD value of less than 5 %, and 2 datasets with RD values between 5% to 10%. The last, the largest RD value, is found in dataset E3, with an RD value of 10.62%. As observed from these results, the proposed method has shown its capability to solve the VRPB when the customer's type constraints are eliminated.

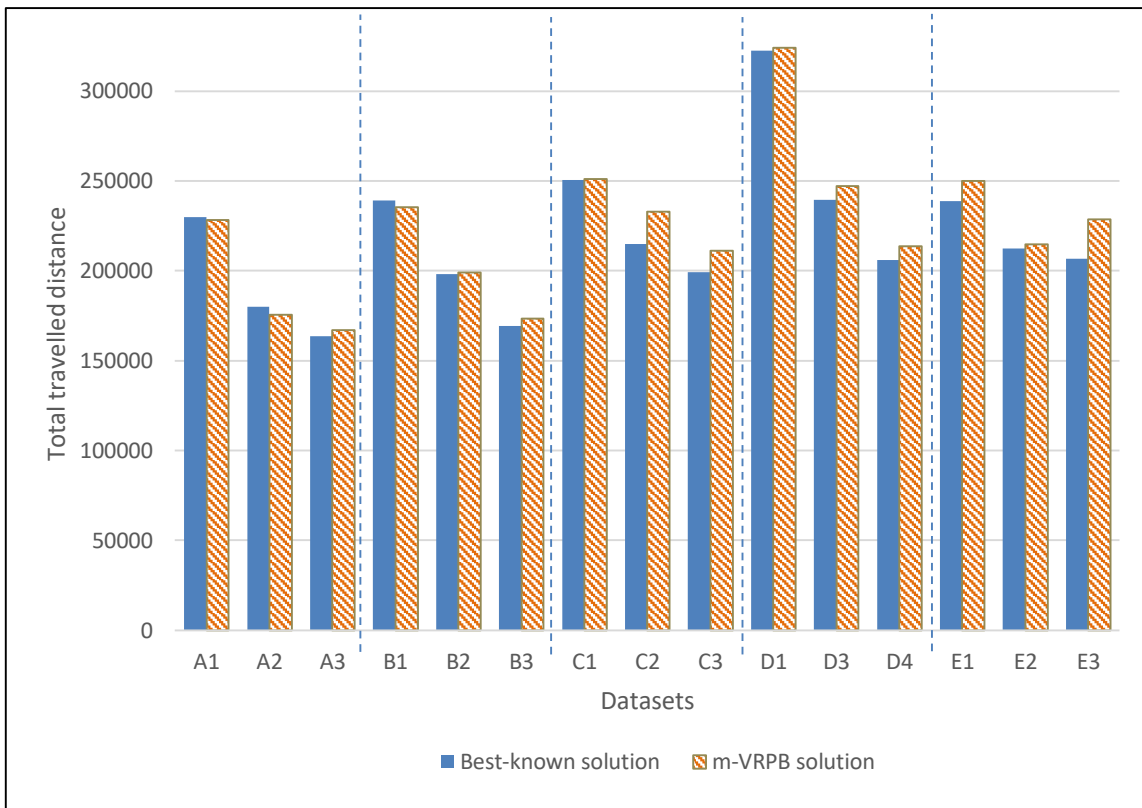


Figure 5.12: Total travelled distance of Best-Known and m-VRPB solution

Moreover, Figure 5.12 above illustrates the outcome of Table 5.3 in the bar chart format. Based on Figure 5.12, there is only a slight difference between the solutions obtained in m-VRPB and the best-known solution. The chart has shown that the total travel distance for datasets A1, B2, C1, D1, and E2 is very close to the best-known solution. The RD value of these datasets only shows a difference of 1%. The results showed that a VRPB model

could solve under a free-serving sequence environment. m-VRPB can achieve a result that is approximate to the best-known solution if the time consumed and the extra workforce used to rearrange the load after every serving is not considered.

Furthermore, the final solutions obtained by datasets A1, A2, and B1 are slightly shorter than the best-known solution. These results indicated that the serving sequence is an important factor affecting the achievement of the final solution. Thus, although the dataset and vehicle number are the same, the final solution might be different if the serving pattern changes.

5.3.2 Minimise the Number of Vehicles

Apart from that, the strength of the proposed m-VRPB methods is further tested by minimising the pre-assigned vehicle number used to complete the tasks. The objective is to minimise the number of vehicles while solving the benchmark instances in a dynamic and real-life situation. In this section, two factors below,

- i. Number of vehicles
- ii. Constraints
 - a. Routes are not allowed to involve only backhaul customer
 - b. Linehaul customers are served before any backhaul customer

are not taken into consideration when generating the solution. In this section, the number of vehicles is not restricted in forming the feasible solution in both the initial and VNS improvement stages. Therefore, the Part A: Route Forming process shown in Figure 5.4 is cancelled. The new route is created if a leftover customer exists that the available routes cannot serve. The whole process ends when all customers have been served. Figure 5.13

below shows the data flow diagram for generating the initial solution for m-VRPB without considering the predefined vehicle number.

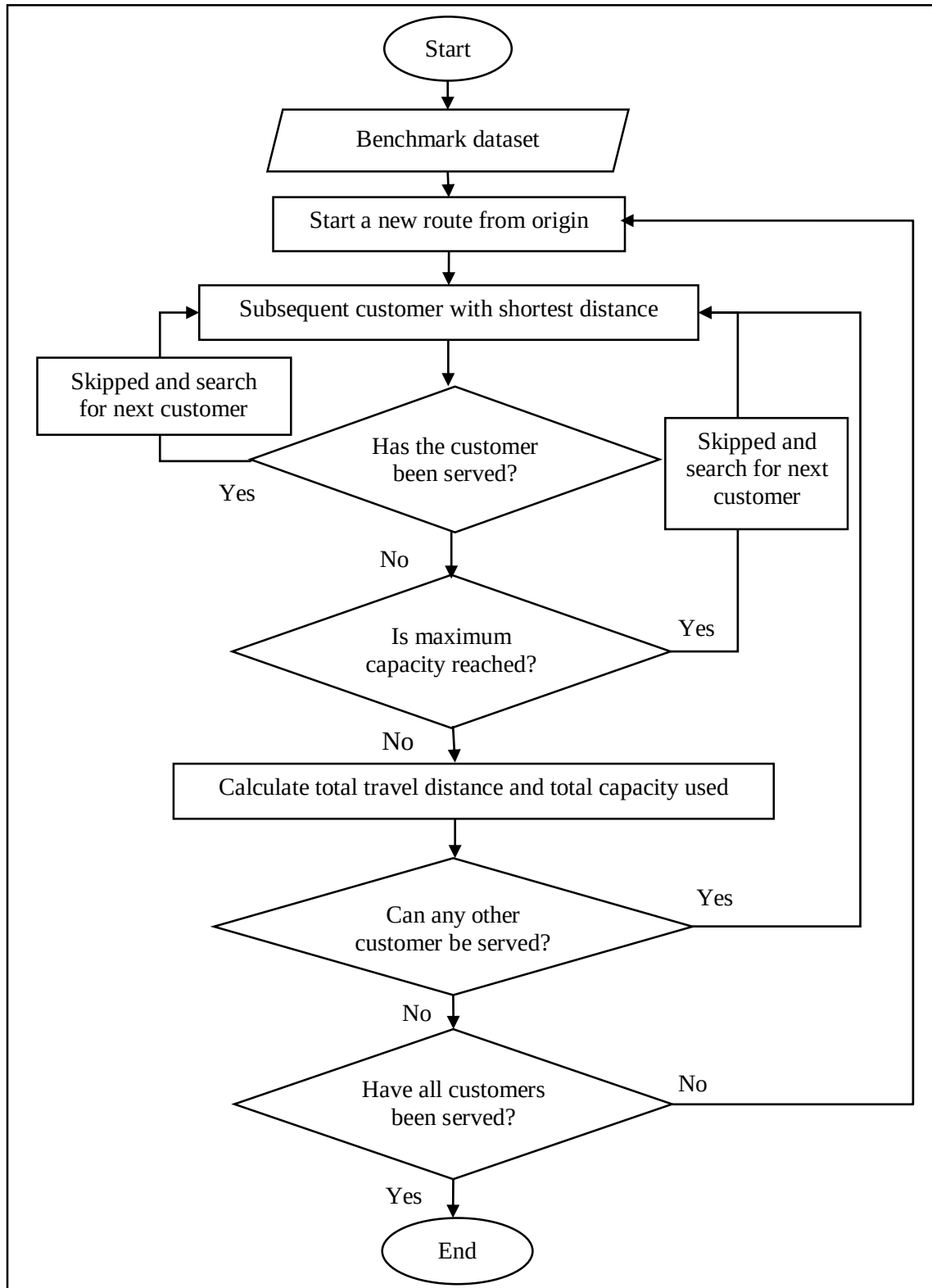


Figure 5.13: Data flow diagram of initial solution (unfixed vehicle number)

route_no	route_list	total_travelled_distance	total_delivery_amount	total_pickup_amount
1	0 25 15 17 21 0	31507.341	1196	743
2	0 11 24 5 7 20 0	47,061.9094	1446	1239
3	0 10 3 22 0	16130.0078	1431	0
4	0 13 2 16 0	29735.829	1531	0
5	0 18 14 1 8 0	66,287.4625	1269	0
6	0 19 4 12 0	24234.5552	1286	0
7	0 9 23 6 0	32615.107	1243	0

Figure 5.14: Initial solution of Dataset A1 in m-VRPB (unfixed vehicle number)

Figure 5.14 shows the sample outcome retrieved for Dataset A1. As observed from Figure 5.14, the number of routes formed is 7 instead of 8 (the number of vehicles stated in the benchmark dataset). This is because the new route is now only created when needed, and similar to the solution generated for m-VRPB, the backhaul customers can be served in any position. Table 5.4 and Figure 5.15 below compare the total travel distance and total vehicle use of the best-known and the m-VRBP without a pre-defined vehicle number. The system outputs for the m-VRPB (unfixed vehicle number) solutions are appended in Appendix C, Table A3.

Table 5.4: Lists of Best-Known and m-VRPB (unfixed vehicle number) solution

Dataset	Best-Known Solution		m-VRPB (Unfixed vehicle number)	
	Travelled Distance	Vehicle Number	Travelled Distance	Vehicle Number
A1	229886	8	247572	7*
A2	180119	5	191119	5
A3	163405	4	180137	3*
B1	239080	7	235160*	8
B2	198048	5	272542	4*
B3	169372	3	182269	3
C1	250557	7	271030	7
C2	215020	5	234360	5
C3	199346	5	193608*	4*
D1	322530	12	330263	10*
D3	239479	7	246988	6*
D4	205832	5	251192	4*
E1	238880	7	279780	7
E2	212263	4	241654	4
E3	206659	4	272944	3*

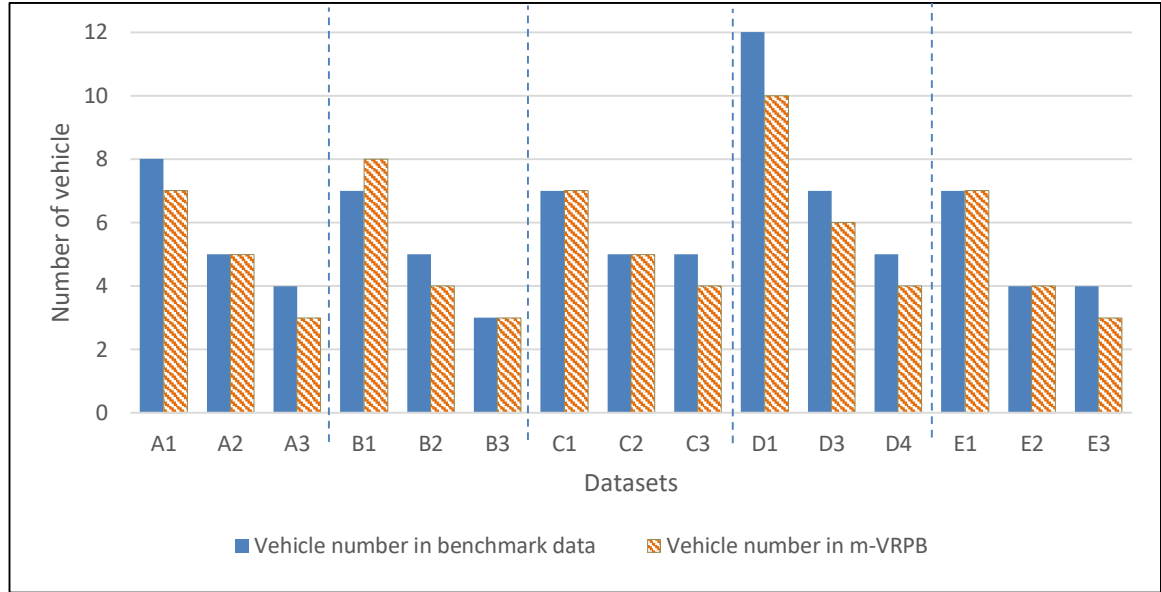


Figure 5.15: Number of vehicles of Best-Known and final research outcomes

From Figure 5.15, the number of vehicles for datasets A2, B3, C1, C2, E1 and E2 is the same as the fixed vehicle number pre-set in the benchmark. The results show that m-VRPB is able to generate the solution with a vehicle number that is the same as the pre-defined amount even though the number of vehicles that can be used was not restricted.

Moreover, dataset A1, A3, B2, C3, D1, D3, D4 and E3 shows that customers can serve completely with fewer vehicle numbers than the provided vehicle amount. Although the total travel distance obtained is increased, this solution may be suitable for an area with a shortage of human resources or when the human cost is higher than the fuel oil cost. In addition, dataset B1 is the only dataset that took the vehicle more than the pre-set vehicle number. However, the total travel distance obtained in B1 is shorter than the best-known solution. This result indicated that more vehicles or routes used do not always lead to a greater total distance.

Besides, dataset C3 has achieved a better result in total travel distance and total vehicle used. It shows that the proposed algorithm for m-VRPB in this section is able to minimise the total travel distance together with the number of vehicles used.

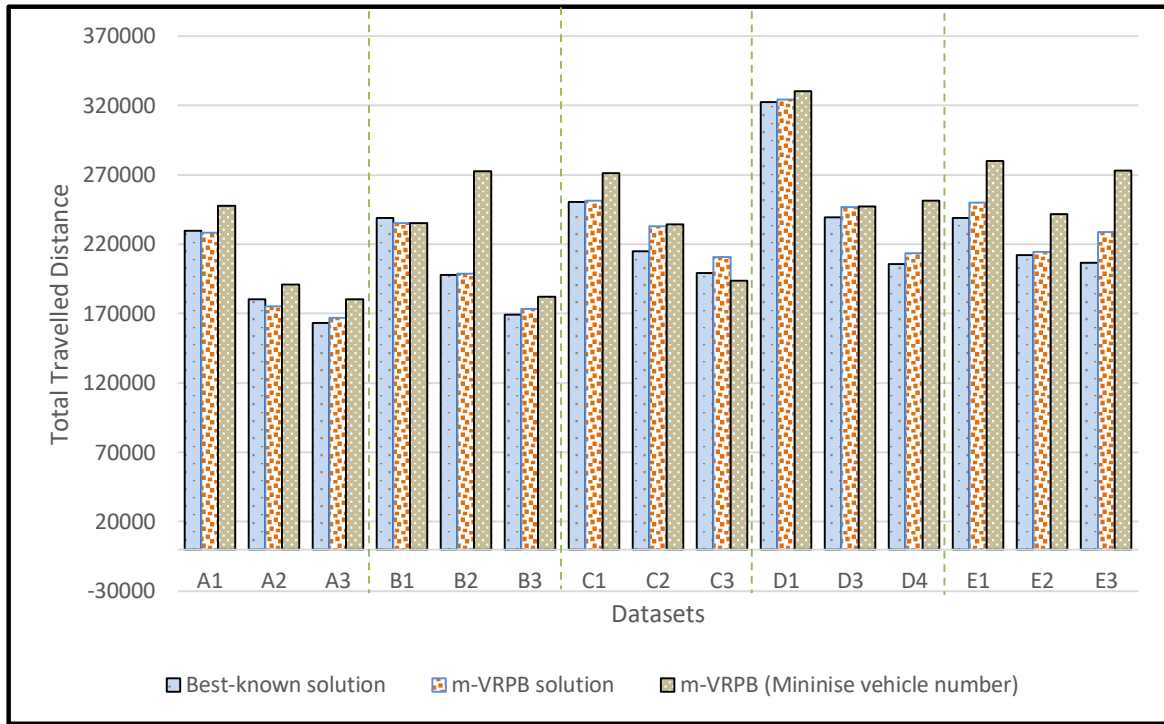


Figure 5.16: Total travelled distance of Best-Known, m-VRPB and m-VRPB (unfixed vehicle number)

Figure 5.16 shows the graph of the total distance travelled of the best-known solution, m-VRPB solution and m-VRPB solution with the minimised vehicle. In Figure 5.16, the bar chart for the benchmark datasets A2, A3, B1, B3, D1 and D3 shows that the total distance travelled is on par with each other among the best-known solutions and two m-VRPB solutions generated. It shows that the proposed m-VRPB has the ability to solve a VRPB. Besides, 2 out of 15 instances can achieve a better solution after relaxing the constraints and the prefixed number of vehicles of the VRPB.

5.4 Conclusion

In this chapter, the performance of the proposed algorithm has been further investigated by some transformation on the circumstance of classical VRPB. m-VRPB is tested using the same benchmark datasets as VRPB. However, only 15 sets of benchmark datasets have been chosen in this chapter. The datasets are removed based on the uniqueness

of the datasets and also the performance of the VNS algorithm proposed for VRPB. This chapter proposes a new simple priority rule heuristic, three local search techniques, and three random shaking methods.

Firstly, two constraints have been relaxed to simulate the practical environment. Thus, m-VRPB is generated based on the new constraints. The results obtained show that the proposed algorithm for m-VRPB can adapt to the new constraints. Moreover, it has achieved results that are close to the best-known solution.

Furthermore, the new proposed algorithm for m-VRPB is further tested by relaxing the use of all predefined vehicle numbers. The results show that the proposed algorithm can obtain the outcome solution that shows the reductions in total transportation distance and vehicle use. In short, the overall result of m-VRPB has shown the strength of the proposed algorithm in solving VRPB.

CHAPTER 6

CONCLUSION

6.1 Introduction

This chapter concludes our research study in both Vehicle Routing Problem with Backhaul (VRPB) and mixed-VRPB (m-VRPB), from model formulation, proposed algorithms and discussion on the computation result. Besides, the contribution of this research work is presented. Future works are also discussed in this chapter.

6.2 Conclusion of research

In this research, the VRPB model has been investigated. VRPB is an extended problem from VRP. It consists of a group of linehaul and backhaul customers, a single depot and a list of homogeneous vehicles. The goal of VRPB is to optimise the feasible routing plans which involve linehaul and backhaul customers. The research objective of this study is to model a VRPB simulation with extra operational constraints consideration. Furthermore, design and develop an effective Variable Neighbourhood Search (VNS) algorithm for VRPB.

The simple priority heuristic rule is developed to generate the initial dataset solution. A VNS algorithm is then used to improve the initial solution. A set of local search approaches and random shaking methods are proposed. The proposed algorithms were used to relocate the customer node in a systematic arrangement order and produced a list of neighbourhood solutions. The solution with the minimal total travelled distance among the neighbourhood solution is selected as the final outcome of this project. The generated solution complies with VRPB constraints.

The computational result shows that the proposed algorithm is able to solve VRPB for up to a size of 113 customers. A total of 22 sets of benchmark data are selected to test the algorithm. The proposed algorithm has solved 63.64% of the selected benchmark to the optimum, 18.18% of them get a lower Relative Percentage Deviation (*RD*) value which is less than 5%, and 18.18% of them obtained an *RD* value between 5% to 15% compared to the best-known solution that found from literature. The results have proven the proposed algorithm can solve VRPB to or equal to the best-known solution with zero violation of the constraints.

Furthermore, some constraints are relaxed from VRPB to simulate a practical environment. m-VRPB is developed. The algorithm is released from linehaul before the backhaul rule; meanwhile, the fixed predefined number of vehicles is not considered. The result shows that m-VRPB can achieve a result close to the best-known solution, although some constraints have changed. There are 2 out of 15 sets of instances able to yield a shorter transportation distance compared to the best-known. Besides, 8 instances can complete serving all the customers with fewer vehicles. Moreover, m-VRPB has obtained the outcome solution showing reductions in total transportation distance and vehicle use. The results have illustrated that the proposed solution is effective and robust in solving VRPB.

6.3 Research Contribution

There are several contributions achieved in this project:

1. Proposed a computational bounded algorithm for VRPB

For the study on VRPB, a simple priority heuristic rule is proposed to generate a feasible initial solution. Besides, in the improvement stage, a VNS with three local search techniques and two random shaking methods is developed to solve VRPB. The initial algorithm is fast

response and able to provide a feasible solution. Other than that, the proposed VNS shows a significant improvement in the quality of the initial solution, as much as 77%. Furthermore, the proposed packaged heuristics method is a simple, feasible, and fast response.

ii. Develop a VRPB model that simulate Benchmark data

A comprehensive VRPB model that is able to fulfil all the given jobs and cover all the constraints is simulated. A VNS model, which is easy to cope with all considerations, is developed. The final outcomes for the 22 sets of benchmark data from this project show that the proposed method has solved the VRPB model with different data sizes and extra consideration and is able to cope with all the given constraints. The final result obtained is on par with the literature's best-known solution.

iii. Proposed a computational bounded algorithm for m-VRPB

A new m-VRPB is generated by relaxing certain constraints for simulating the practical environment. An easy and effective new VNS with three new local search techniques and three random shaking methods are developed to solve m-VRPB. The proposed packaged algorithm can cover and achieve multi-objective functions such as minimising the total travelled distance and reducing the number of vehicles used in m-VRPB.

iv. Develop an m-VRPB model that simulates Benchmark data

A new set of ordering rules and improvement techniques are applied to solve m-VRPB. Then, the algorithm's strength is further tested by relaxing the use of all predefined vehicle numbers. A total of 15 sets of benchmark datasets are tested. The results show that m-VRPB can adapt to all the new operational constraints, improve the initial solution

significantly, and obtain the outcome solution that shows the reductions in total transportation distance and vehicle use.

6.4 Future Works

There are some features that could be investigated in the future. Firstly, the proposed VRPB algorithm could be changed to improve the effectiveness of solving a big-scale dataset. The results yielded for small-scale instances are better than the big-scale instances. The solution for the big-scale data can be improved by further tuning the current local search and the shaking techniques. Moreover, more local search and shaking methods can improve the final output.

Besides, the current research has just tested up to 113 data size instances since the computing effort that needs to run the large-scale data is limited. The algorithm could be changed to cope with the large-scale dataset instance. Apart from that, the computing time could be reduced by implementing a powerful CPU processor or executing by parallel computing. Lastly, the stopping iteration proposed for the algorithm can be increased with the hardware upgrade. The current stopping criteria are only set to a maximum of 30 iterations due to the lengthy computing time and the limited hardware specifications. It can be more than 30 to increase the chances of generating the optimum solution.

However, for the m-VRPB, the performance of the proposed algorithm can be tested with other benchmark datasets. Then, we can model the m-VRPB by simulating simultaneous customer characteristics for pickup and delivery problems.

REFERENCES

- Anbuudayasankar, S. P., Ganesh, K., Koh, S. L., & Ducq, Y. (2012). Modified savings heuristics and genetic algorithm for bi-objective vehicle routing problem with forced backhauls. *Expert Systems with Applications*, 39(3), 2296-2305.
- Anbuudayasankar, S. P., Ganesh, K., & Mohapatra, S. (2014). *Models for practical routing problems in logistics*. Springer International Publication.
- Aravind, P., Babu, D., & Paul, B. (2014). A Review of Vehicle Routing Problem with Simultaneous Pickup and Delivery. *International Journal of Engineering Trends and Technology*, 15(4), 203-205.
- Aufa, M. (2022) Delivery, courier services gear up after successful 2021. *The Malaysian Reserve*. [online] Available at: <https://themalaysianreserve.com/2022/01/05/delivery-courier-services-gear-up-after-successful-2021>. [Assessed on 13 September 2022].
- Balseiro, S. R., Loiseau, I., & Ramonet, J. (2011). An ant colony algorithm hybridized with insertion heuristics for the time dependent vehicle routing problem with time windows. *Computers & Operations Research*, 38(6), 954-966.
- Barbarosoglu, G., & Ozgur, D. (1999). A tabu search algorithm for the vehicle routing problem. *Computers & Operations Research*, 26(3), 255-270.
- Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W., & Vance, P. H. (1998). Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46(3), 316-329.
- Battiti, R., & Tecchiolli, G. (1994). The reactive tabu search. *ORSA. Journal on Computing*, 6(2), 126-140.
- Bianchessi, N., & Righini, G. (2007). Heuristic algorithms for the vehicle routing problem

- with simultaneous pick-up and delivery. *Computers & Operations Research*, 34(2), 578-594.
- Braekers, K., Ramaekers, K., & Van Nieuwenhuysse, I. (2016). The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, 99, 300-313.
- Bräysy, O. (2003). A reactive variable neighborhood search for the vehicle-routing problem with time windows. *INFORMS Journal on Computing*, 15(4), 347-368.
- Carpaneto, G., Dell'Amico, M., Fischetti, M., & Toth, P. (1989). A branch and bound algorithm for the multiple depot vehicle scheduling problem. *Networks*, 19(5), 531-548.
- Casco, D. O., Golden, B. L., & Wasil, E. A. (1988). *Vehicle routing with backhauls: Models, algorithms, and case studies. vehicle routing: methods and studies. studies in management science and systems-Volume 16*. Publication of: Dalctraf.
- Chen, P., Golden, B., Wang, X., & Wasil, E. (2017). A novel approach to solve the split delivery vehicle routing problem. *International Transactions in Operational Research*, 24(1-2), 27-41.
- Cheung, R. K., & Hang, D. D. (2003). Multi-attribute label matching algorithms for vehicle routing problems with time windows and backhauls. *IIE Transactions*, 35(3), 191-205.
- Clarke, G., & Wright, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12(4), 568-581.
- Crainic, T. G., Gendreau, M., Hansen, P., & Mladenović, N. (2004). Cooperative parallel variable neighborhood search for the p-median. *Journal of Heuristics*, 10(3), 293-314.

- Crainic, T. G., & Laporte, G. (1997). Planning models for freight transportation. *European Journal of Operational Research*, 97(3), 409-438.
- Crispim, J., & Brandao, J. (2001). Reactive tabu search and variable neighborhood descent applied to the vehicle routing problem with backhauls. In *Proceedings of the 4th Metaheuristics International Conference, Porto, MIC* (pp. 631-636).
- Crispim, J., & Brandão, J. (2005). Metaheuristics applied to mixed and simultaneous extensions of vehicle routing problems with backhauls. *Journal of the Operational Research Society*, 56(11), 1296-1302.
- Cuervo, D.P., Goos, P., Sörensen, K., & Arráiz, E. (2014). An iterated local search algorithm for the vehicle routing problem with backhauls. *European Journal of Operational Research*, 237, 454-464.
- Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. *Management Science*, 6(1), 80-91.
- Dee-Ann, D. (2022) Demand for grocery delivery cools as food costs rise. *Los Angeles Times*. [online] Available at: <https://www.latimes.com/business/story/2022-08-07/demand-for-grocery-delivery-cools-as-food-costs-rise>. [Assessed on 13 August 2022].
- Deif, I., & Bodin, L. (1984). Extension of the Clarke and Wright algorithm for solving the vehicle routing problem with backhauling. In *Proceedings of the Babson Conference on Software Uses in Transportation and Logistics Management*, (pp. 75–96).
- Dell’Amico, M., Righini, G., & Salani, M. (2006). A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection. *Transportation Science*, 40(2), 235-247.
- Desaulniers, G., Desrosiers, J., Erdmann, A., Solomon, M. M., & Soumis, F. (2002). VRP

- with Pickup and Delivery. *The Vehicle Routing Problem*, 9, 225-242.
- Dong, G., Tang, J., Lai, K. K., & Kong, Y. (2011). An exact algorithm for vehicle routing and scheduling problem of free pickup and delivery service in flight ticket sales companies based on set-partitioning model. *Journal of Intelligent Manufacturing*, 22(5), 789-799.
- Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant colony optimization. *IEEE Computational Intelligence Magazine*, 1(4), 28-39.
- Dorigo, M., Di Caro, G., & Gambardella, L. M. (1999). Ant algorithms for discrete optimization. *Artificial Life*, 5(2), 137-172.
- Dorigo, M., Maniezzo, V., & Coloni, A. (1996). Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26(1), 29-41.
- Elshaer, R., & Awad, H. (2020). A taxonomic review of metaheuristic algorithms for solving the vehicle routing problem and its variants. *Computers & Industrial Engineering*, 140, 106242.
- Ehmke, J. F., Campbell, A. M., & Thomas, B. W. (2016). Vehicle routing to minimize time-dependent emissions in urban areas. *European Journal of Operational Research*, 251(2), 478-494.
- Errico, F., Desaulniers, G., Gendreau, M., Rei, W., & Rousseau, L. M. (2016). A priori optimization with recourse for the vehicle routing problem with hard time windows and stochastic service times. *European Journal of Operational Research*, 249(1), 55-66.
- Farahani, R. (2011). *Logistics operations and management: concepts and models*. Elsevier.
- Feng, M., Mangan, J., & Lalwani, C. (2012). Comparing port performance: Western

- European versus eastern Asian ports. *International Journal of Physical Distribution & Logistics Management*.
- Fischetti, M., & Toth, P. (1989). An additive bounding procedure for combinatorial optimization problems. *Operations Research*, 37(2), 319-328.
- Fisher, M. (1995). Vehicle routing. *Handbooks in Operations Research and Management Science*, 8, 1-33.
- Fleszar, K., Osman, I. H., & Hindi, K. S. (2009). A variable neighbourhood search algorithm for the open vehicle routing problem. *European Journal of Operational Research*, 195(3), 803-809.
- Gajpal, Y., & Abad, P. L. (2009). Multi-ant colony system (MACS) for a vehicle routing problem with backhauls. *European Journal of Operational Research*, 196(1), 102-117.
- Gao, S., Wang, Y., Cheng, J., Inazumi, Y., & Tang, Z. (2016). Ant colony optimization with clustering for solving the dynamic location routing problem. *Applied Mathematics and Computation*, 285, 149-173.
- García-Nájera, A., Bullinaria, J. A., & Gutiérrez-Andrade, M. A. (2015). An evolutionary approach for multi-objective vehicle routing problems with backhauls. *Computers & Industrial Engineering*, 81, 90-108.
- García-López, F., Melián-Batista, B., Moreno-Pérez, J. A., & Moreno-Vega, J. M. (2002). The parallel variable neighborhood search for the p-median problem. *Journal of Heuristics*, 8(3), 375-388.
- Garey, M. R., & Johnson, D. S. (1979). *Computers and Intractability*, San Francisco. CA: W. H. Freeman.
- Gayialis, S. P., & Tatsiopoulos, I. P. (2004). Design of an IT-driven decision support system

- for vehicle routing and scheduling. *European Journal of Operational Research*, 152(2), 382-398.
- Goetschalckx, M., (2022) *Linehaul- Backhaul*. [online] Available at: https://www2.isye.gatech.edu/~mgoetsch/lineback.html#Linehaul_Backhaul [Assessed on 08 August 2022].
- Goetschalckx, M., & Jacobs-Blecha, C. (1989). The vehicle routing problem with backhauls. *European Journal of Operational Research*, 42(1), 39-51.
- Hansen, P., Mladenović, N., & Moreno Perez, J. A. (2008). Variable neighbourhood search: methods and applications. *4OR*, 6(4), 319-360.
- Hansen, P., Mladenović, N., & Perez-Britos, D. (2001). Variable neighborhood decomposition search. *Journal of Heuristics*, 7(4), 335-350.
- Hwang, D. W., Hong, P. C., & Lee, D. Y. (2017). Critical factors that affect logistics performance: a comparison of China, Japan and Korea. *International Journal of Shipping and Transport Logistics*, 9(1), 107-129.
- Kalayci, C. B., & Kaya, C. (2016). An ant colony system empowered variable neighbourhood search algorithm for the vehicle routing problem with simultaneous pickup and delivery. *Expert Systems with Applications*, 66, 163-175.
- Koç, Ç., & Karaoglan, I. (2016). The green vehicle routing problem: A heuristic based exact solution approach. *Applied Soft Computing*, 39, 154-164.
- Koç, Ç., & Laporte, G. (2018). Vehicle routing with backhauls: Review and research perspectives. *Computers & Operations Research*, 91, 79-91.
- Kuo, R. J., Wibowo, B. S., & Zulvia, F. E. (2016). Application of a fuzzy ant colony system to solve the dynamic vehicle routing problem with uncertain service time. *Applied Mathematical Modelling*, 40(23-24), 9990-10001.

- Kytöjoki, J., Nuortio, T., Bräysy, O., & Gendreau, M. (2007). An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. *Computers & Operations Research*, 34(9), 2743-2757.
- Lalla-Ruiz, E., Expósito-Izquierdo, C., Taheripour, S., & Voß, S. (2016). An improved formulation for the multi-depot open vehicle routing problem. *OR Spectrum*, 38(1), 175-187.
- Laporte, G. (1992). The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(3), 345-358.
- Laporte, G., & Nobert, Y. (1987). Exact algorithms for the vehicle routing problem. In *North-Holland Mathematics Studies*, 132, 147-184.
- Li, L., Meng, J., & Chen, Y. (2019, June). Brief algorithm review on vehicle routing problems with different backhaul constraints. In *2019 Chinese Control and Decision Conference (CCDC)* (pp. 4152-4157).
- Lin, J. H., & Chou, T. C. (2012). A geo-aware and VRP-based public bicycle redistribution system. *International Journal of Vehicular Technology*, 2012.
- Mazzeo, S., & Loiseau, I. (2004). An ant colony algorithm for the capacitated vehicle routing. *Electronic Notes in Discrete Mathematics*, 18, 181-186.
- Mendoza, J. E., Guéret, C., Hoskins, M., Lobit, H., Pillac, V., Vidal, T., & Vigo, D. (2014). *VRP-REP: the vehicle routing community repository. Third meeting of the EURO Working Group on Vehicle Routing and Logistics Optimization (VeRoLog)*. [online] Available at: <http://www.vrp-rep.org/datasets/item/2016-0005.html> [Assessed on 08 August 2022].
- Mingozi, A., Giorgi, S., & Baldacci, R. (1999). An exact method for the vehicle routing problem with backhauls. *Transportation Science*, 33(3), 315-329.

- Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, 24(11), 1097-1100.
- Naddef, D., & Rinaldi, G. (2002). Branch-and-cut algorithms for the capacitated VRP. In *The Vehicle Routing Problem* (pp. 53-84). Society for Industrial and Applied Mathematics.
- Nagy, G., & Salhi, S. (2005). Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries. *European Journal of Operational Research*, 162(1), 126-141.
- Nagy, G., Wassen, N. A., & Salhi, S. (2013). The vehicle routing problem with restricted mixing of deliveries and pickups. *Journal of Scheduling*, 16(2), 199-213.
- Norouzi, N., Sadegh-Amalnick, M., & Tavakkoli-Moghaddam, R. (2017). Modified particle swarm optimization in a time-dependent vehicle routing problem: minimizing fuel consumption. *Optimization Letters*, 11(1), 121-134.
- Ochi, L. S., Silva, M. B., & Drummond, L. (2001). Metaheuristics based on GRASP and VNS for solving traveling purchaser problem. *MIC'2001*, 489-494.
- Osman, I. H., & Wassen, N. A. (2002). A reactive tabu search meta-heuristic for the vehicle routing problem with back-hauls. *Journal of Scheduling*, 5(4), 263-285.
- Padberg, M., & Rinaldi, G. (1991). A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Review*, 33(1), 60-100.
- Parragh, S. N., Doerner, K. F., & Hartl, R. F. (2008). A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, 58(1), 21-51.
- Pinto, T., Alves, C., & de Carvalho, J. V. (2017). Variable neighborhood search algorithms for pickup and delivery problems with loading constraints. *Electronic Notes in Discrete Mathematics*, 58, 111-118.

- Pinto, T., Alves, C., & Valério de Carvalho, J. (2020). Variable neighborhood search algorithms for the vehicle routing problem with two-dimensional loading constraints and mixed linehauls and backhauls. *International Transactions in Operational Research*, 27(1), 549-572.
- Pishvaei, M. S., & Basiri, H. (2009). National logistics costs. In *Supply Chain and Logistics in National, International and Governmental Environment* (pp. 57-83). Physica-Verlag Heidelberg.
- Polat, O. (2017). A parallel variable neighborhood search for the vehicle routing problem with divisible deliveries and pickups. *Computers & Operations Research*, 85, 71-86.
- Póvoa, A. (2020). Vehicle Routing Problem with Backhauls. *Towards Solving A Robust and Sustainable Vehicle Routing Problem with Backhauls*, 11.
- Queiroga, E., Frota, Y., Sadykov, R., Subramanian, A., Uchoa, E., & Vidal, T. (2020). On the exact solution of vehicle routing problems with backhauls. *European Journal of Operational Research*, 287(1), 76-89.
- Raff, S. (1983). Routing and scheduling of vehicles and crews: The state of the art. *Computers & Operations Research*, 10(2), 63-211.
- Reshna, R. D. (2022) Expect prices to go up to 10%, FMM warns consumers. *Free Malaysia Today*. [online] Available at: <https://www.freemalaysiatoday.com/category/nation/2022/02/05/expect-prices-to-go-up-by-10-fmm-warns-consumers>. [Assessed on 14 October 2022].
- Rochat, Y., & Semet, F. (1994). A tabu search approach for delivering pet food and flour in Switzerland. *Journal of the Operational Research Society*, 45(11), 1233-1246.
- Rousseau, L. M., Gendreau, M., & Pesant, G. (2002). Using constraint-based operators to solve the vehicle routing problem with time windows. *Journal of Heuristics*, 8(1),

43-58.

- Sakhapov, R. L., Nikolaeva, R. V., Gatiyatullin, M. H., & Makhmutov, M. M. (2018). Modeling of regional transport and logistics systems. In *Journal of Physics: Conference Series 1141*(1). IOP Publishing.
- Salhi, S., & Nagy, G. (1999). A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. *Journal of the Operational Research Society*, 50(10), 1034-1042.
- Santos, M. J., Amorim, P., Marques, A., Carvalho, A., & Póvoa, A. (2020). The vehicle routing problem with backhauls towards a sustainability perspective: A review. *Top*, 28(2), 358-401.
- Santos, M. J., Martins, S., Amorim, P., & Almada-Lobo, B. (2021). A green lateral collaborative problem under different transportation strategies and profit allocation methods. *Journal of Cleaner Production*, 288, 125678.
- Soleimani, H., Chaharlang, Y., & Ghaderi, H. (2018). Collection and distribution of returned-remanufactured products in a vehicle routing problem with pickup and delivery considering sustainable and green criteria. *Journal of Cleaner Production*, 172, 960-970.
- Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2), 254-265
- Statista. (2022) *Platform-to-consumer delivery*. [online] Available at: <https://www.statista.com/outlook/dmo/eservices/online-food-delivery/platform-to-consumer-delivery>. [Assessed on 13 August 2022].
- Tian, X., Geng, Y., Zhong, S., Wilson, J., Gao, C., Chen, W., & Hao, H. (2018). A bibliometric analysis on trends and characters of carbon emissions from transport

- sector. *Transportation Research Part D: Transport and Environment*, 59, 1-10.
- Tok Pa: E-commerce, food delivery service gross merchandise value expanded nearly RM30b in 2021. (2022, January 13). *Malaymail*. [online] Available at: <https://www.malaymail.com/news/money/2022/01/13/tok-pa-e-commerce-food-delivery-service-gross-merchandise-value-expanded-ne/2034839>. [Assessed on 13 December 2022].
- Toth, P., & Vigo, D. (1996). A heuristic algorithm for the vehicle routing problem with backhauls. In *Advanced Methods in Transportation Analysis* (pp. 585-608). Springer, Berlin, Heidelberg.
- Toth, P., & Vigo, D. (1997). An exact algorithm for the vehicle routing problem with backhauls. *Transportation Science*, 31(4), 372-385.
- Toth, P., & Vigo, D. (1999). A heuristic algorithm for the symmetric and asymmetric vehicle routing problems with backhauls. *European Journal of Operational Research*, 113(3), 528-543.
- Toth, P., & Vigo, D. (2002). Branch-and-bound algorithms for the capacitated VRP. In *The Vehicle Routing Problem* (pp. 29-51). Society for Industrial and Applied Mathematics.
- Ubeda, S., Arcelus, F. J., & Faulin, J. (2011). Green logistics at Eroski: A case study. *International Journal of Production Economics*, 131(1), 44-51.
- Uchoa, E., Pecin, D., Pessoa, A., Poggi, M., Vidal, T., & Subramanian, A. (2017). New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, 257(3), 845-858.
- Wade, A., & Salhi, S. (2003). An ant system algorithm for the mixed vehicle routing problem with backhauls. In *Metaheuristics: Computer Decision-making* (pp. 699-719).

Springer, Boston, MA.

- Wang, J., Zhou, Y., Wang, Y., Zhang, J., Chen, C. P., & Zheng, Z. (2015). Multiobjective vehicle routing problems with simultaneous delivery and pickup and time windows: formulation, instances, and algorithms. *IEEE Transactions on Cybernetics*, 46(3), 582-594.
- Wang, Y., Peng, S., Zhou, X., Mahmoudi, M., & Zhen, L. (2020). Green logistics location-routing problem with eco-packages. *Transportation Research Part E: Logistics and Transportation Review*, 143, 102118.
- Wantanakomol, S. (2021). The effect of guidelines on reducing logistics costs. *Uncertain Supply Chain Management*, 9(3), 667-674.
- Wassan, N. (2007). Reactive tabu adaptive memory programming search for the vehicle routing problem with backhauls. *Journal of the Operational Research Society*, 58(12), 1630-1641.
- Wassan, N. A., Salhi, S., Nagy, G., Wassan, N., & Wade, A. C. (2013). Solving the mixed backhauling vehicle routing: problem with ants. *International Journal of Energy Optimization and Engineering (IJE OE)*, 2(2), 62-77.
- Wassan, N. A., Wassan, A. H., & Nagy, G. (2008). A reactive tabu search algorithm for the vehicle routing problem with simultaneous pickups and deliveries. *Journal of Combinatorial Optimization*, 15(4), 368-386.
- Wassan, N., Wassan, N., Nagy, G., & Salhi, S. (2017). The multiple trip vehicle routing problem with backhauls: Formulation and a two-level variable neighbourhood search. *Computers & Operations Research*, 78, 454-467.
- Wilson, N. H., Weissberg, R. W., & Hauser, J. (1976). *Advanced Dial-a-ride Algorithms Research Project* (No. R76-20 Final RPt.).

- Wu, W., Tian, Y., & Jin, T. (2016). A label based ant colony algorithm for heterogeneous vehicle routing with mixed backhaul. *Applied Soft Computing*, 47, 224-234.
- Xiao, Y., & Konak, A. (2017). A genetic algorithm with exact dynamic programming for the green vehicle routing & scheduling problem. *Journal of Cleaner Production*, 167, 1450-1463.
- Yang, Y. (2022) Surging delivery business reflects new consumption growth. *ChinaDaily.Com.Cn* [online] Available at: <http://global.chinadaily.com.cn/a/202203/21/WS62381d6ca310fd2b29e52409.html> [Assessed on 13 August 2022].
- Yano, C. A., Chan, T. J., Richter, L. K., Cutler, T., Murty, K. G., & McGettigan, D. (1987). Vehicle routing at quality stores. *Interfaces*, 17(2), 52-63.
- Zachariadis, E. E., Tarantilis, C. D., & Kiranoudis, C. T. (2009). A hybrid metaheuristic algorithm for the vehicle routing problem with simultaneous delivery and pick-up service. *Expert Systems with Applications*, 36(2), 1070-1081.
- Zhao, X., & Tang, Q. (2009). Analysis and strategy of the chinese logistics cost reduction. *International Journal of Business and Management*, 4(4), 188-191.
- Zhang, H., Ge, H., Yang, J., & Tong, Y. (2022). Review of vehicle routing problems: Models, classification and solving algorithms. *Archives of Computational Methods in Engineering*, 29(1), 195-221.

APPENDICES

Appendix A: Journal Publication

1. Sze, S.N., Sek, S.Y.D., Sze, J.F., Cheah, W.S. & Chiew, K.L. (2020) Vehicle Routing Problem with Simultaneous Pickup and Delivery. *International Journal on Advanced Science, Engineering and Information Technology*, 10(4). pp. 1360-1366. ISSN 2088-5334.

Appendix B: Program code

```
$distance_grid = array();
for ($i=1;$i<=$count;$i++)
{
    $x1 = $arrX[$i];
    $y1 = $arrY[$i];

    $sql = $conn->query("SELECT data_x, data_y FROM origin WHERE id = 1");
    while($row = $sql->fetch_assoc())
    {
        $x2 = $row['data_x'];
        $y2 = $row['data_y'];
    }

    $d = sqrt((pow(($x1-$x2),2))+(pow(($y1-$y2),2)));
}
```

Figure A1: Program code for distance calculation

```
function parallel_sort_first() { // assign first node
    for($i = 0; $i < $vehicle_num; $i++)
    {
        $id = $row_array[$i]['data_id'];
        $n_type = $row_array[$i]['type'];
        $n_distance = $row_array[$i]['distance'];
        $n_demand = $row_array[$i]['demand'];

        if($n_type == 2)
        {
            $load = $o_load + $n_demand;
            if($load > $o_max_load)
            {
                $max_load = $load;
            }
        }
        if($n_type == 1)
        {
            $max_load = $o_max_load + $n_demand;
            $initial = $o_initial + $n_demand;
        }

        $distance = $o_distance + $n_distance;
        $total = $o_total + $n_demand;
        if (($total <= $max_capacity) && ($initial <= $max_capacity) && ($max_load <= $max_capacity))
        {
            if($n_type == 2)
            {
                $datacheck['id'][$i+1] = $id;
                $datacheck['load'][$i+1] = $load;
                $datacheck['initial_total'][$i+1] = $initial;
                $datacheck['distance'][$i+1] = $distance;
                $datacheck['total'][$i+1] = $total;
                $datacheck['max_load'][$i+1] = $max_load;
                $datacheck['pickup_total'][$i+1] = $max_load;
                $datacheck['delivery_total'][$i+1] = 0;
            }
            else
            {
                $datacheck['id'][$i+1] = $id;
                $datacheck['load'][$i+1] = $o_load;
                $datacheck['initial_total'][$i+1] = $initial;
                $datacheck['distance'][$i+1] = $distance;
                $datacheck['total'][$i+1] = $total;
                $datacheck['max_load'][$i+1] = $max_load;
                $datacheck['delivery_total'][$i+1] = $max_load;
                $datacheck['pickup_total'][$i+1] = 0;
            }
        }
    }
    return $datacheck;
}
```

Figure A2: Program code for first node assignment

```

//===== DELIVERY =====//
for($i = 0; $i < $delivery_num; $i++){
{
    $n_id = $delivery_array[$i]['data_id'];
    $n_type = $delivery_array[$i]['type'];
    $n_demand = $delivery_array[$i]['demand'];
    for($a = 1; $a <= $vehicle_num; $a++){
    {
        $o_id = $datacheck['id'][$a];
        $load = $o_load = $datacheck['load'][$a];
        $initial = $o_initial = $datacheck['initial_total'][$a];
        $distance = $o_distance = $datacheck['distance'][$a];
        $total = $o_total = $datacheck['total'][$a];
        $max_load = $o_max_load = $datacheck['max_load'][$a];
        $n_distance = $distance_grid['distance'][$o_id][$n_id];

        if($n_type == 2)
        {
            $load = $o_load + $n_demand;
            if($load > $o_max_load)
            {
                $max_load = $load;
            }
        }
        if($n_type == 1)
        {
            $max_load = $o_max_load + $n_demand;
            $initial = $o_initial + $n_demand;
        }

        $distance = $o_distance + $n_distance;
        $total = $o_total + $n_demand;
        if (($total <= $max_capacity) && ($initial <= $max_capacity) && ($max_load <= $max_capacity)
            && (($max_load >= $compare_load) || ($compare_load == 0)))
        {
            $selected_route = $a;
            $selected_id = $n_id;
            $selected_distance = $distance;
            $selected_load = $load;
            $selected_initial = $initial;
            $selected_max_load = $max_load;
            $selected_total = $total;
        }
    }
    $datacheck['id'][$selected_route] = $selected_id;
    $datacheck['load'][$selected_route] = $selected_load;
    $datacheck['initial_total'][$selected_route] = $selected_initial;
    $datacheck['distance'][$selected_route] = $selected_distance;
    $datacheck['total'][$selected_route] = $selected_total;
    $datacheck['max_load'][$selected_route] = $selected_max_load;
}
}

```

Figure A3: Program code for linehaul customers sorting process

```

//===== PICKUP =====//
for($i = 0; $i < $pickup_num; $i++){
    $n_type = $pickup_array[$i]['type'];
    $n_demand = $pickup_array[$i]['demand'];
    for($a = 1; $a <= $vehicle_num; $a++){
    {
        $o_id = $datacheck['id'][$a];
        $load = $o_load = $datacheck['load'][$a];
        $initial = $o_initial = $datacheck['initial_total'][$a];
        $distance = $o_distance = $datacheck['distance'][$a];
        $total = $o_total = $datacheck['total'][$a];
        $max_load = $o_max_load = $datacheck['max_load'][$a];
        $n_distance = $distance_grid['distance'][$o_id][$n_id];

        if($n_type == 2){
            $load = $o_load + $n_demand;
            if($load > $o_max_load)
            {
                $max_load = $load;
            }
        }
        if($n_type == 1){
            $max_load = $o_max_load + $n_demand;
            $initial = $o_initial + $n_demand;
        }

        $distance = $o_distance + $n_distance;
        $total = $o_total + $n_demand;

        if (($total <= $max_capacity) && ($initial <= $max_capacity) && ($max_load <= $max_capacity)
            && (($max_load >= $compare_load) || ($compare_load == 0))){
            $pre_distance = $distance;
            $compare_load = $max_load;
            $selected_route = $a;
            $selected_id = $n_id;
            $selected_distance = $distance;
            $selected_load = $load;
            $selected_initial = $initial;
            $selected_max_load = $max_load;
            $selected_total = $total;
        }
    }

    $datacheck['id'][$selected_route] = $selected_id;
    $datacheck['load'][$selected_route] = $selected_load;
    $datacheck['initial_total'][$selected_route] = $selected_initial;
    $datacheck['distance'][$selected_route] = $selected_distance;
    $datacheck['total'][$selected_route] = $selected_total;
    $datacheck['max_load'][$selected_route] = $selected_max_load;
}
}

```

Figure A4: Program code for backhaul customers sorting process

```

function check_distance($new_data,$distance_grid,$old_best_distance,$conn){
    $max_capacity = $distance_grid['max_capacity'];
    $num_data = $distance_grid['max_vehicle'];
    $total_distance = 0;
    $checkFlag = 0;
    for($i=1; $i<=$num_data; $i++){
        for($j=0; $j<(sizeof($new_data[$i])-1); $j++){
            $j2 = $j + 1;
            $dataA = $new_data[$i][$j];
            $dataB = $new_data[$i][$j2];

            if($dataB != 0){
                $d = $distance_grid['distance'][$dataA][$dataB];
                $distance = $distance + $d;
                $demand = $distance_grid['demand'][$dataB];
                $type = $distance_grid['type'][$dataB];
                if($type == 2)
                    $pickup_total = $pickup_total + $demand;
                if($type == 1){
                    $initial = $initial + $demand;
                    $delivery_total = $delivery_total + $demand;
                }
            }
            else{
                $d = $distance_grid['distance'][$dataA][0];
                $distance = $distance + $d;
            }

            if(($delivery_total > $max_capacity)||($pickup_total > $max_capacity)||($initial > $max_capacity)){
                $checkFlag = 1;
            }
        }
        $delivery_array[$i] = $delivery_total;
        $pickup_array[$i] = $pickup_total;
        $total_distance = $total_distance + $distance;
    }
    return array($checkFlag, $total_distance, $new_data, $delivery_array, $pickup_array);
}

```

Figure A5: Program code for constraints checking

```

function oneToZeroExchange($datacheck,$distance_grid,$conn){
    for($x = 1; $x <= $num_data; $x++){
        $newArray = $old_data;
        $rowA = $old_data[$x];
        $rowA_length = sizeof($rowA) - 1;

        for($a = ($rowA_length-1); $a >= 1; $a--){
            $array1 = $old_data[$x];
            $removed = $array1[$a];
            $removed_type = $distance_grid['type'][$removed];

            if((sizeof($array1) > 3)){
                for($y = 1; $y <= $num_data; $y++){
                    $newArray2 = $newArray;
                    $rowB = $newArray2[$y];
                    $rowB_length = sizeof($rowB) - 1;

                    for($b = $rowB_length; $b > 0; $b--){
                        $colB = $newArray2[$y][$b];
                        $valueB = $colB;
                        $valueB0 = $newArray2[$y][($b-1)];
                        $typeB = $distance_grid['type'][$valueB];
                        $typeB0 = $distance_grid['type'][$valueB0];

                        if((((($removed_type == 1)&&($typeB == 1) && ($typeB0 == 0)) || (($removed_type == 1)&&($typeB == 1) && ($typeB0 == 1)) ||
                            (($removed_type == 1)&&($typeB == 2) && ($typeB0 == 1)) || (($removed_type == 1)&&($typeB == 0) && ($typeB0 == 1)) ||
                            (($removed_type == 2)&&($typeB == 2) && ($typeB0 == 1)) || (($removed_type == 2)&&($typeB == 2) && ($typeB0 == 2)) ||
                            (($removed_type == 2)&&($typeB == 0) && ($typeB0 == 2)) || (($removed_type == 2)&&($typeB == 0) && ($typeB0 == 1))
                        )){
                            $new_data = $newArray2;
                            $return_array = check_distance($new_data,$distance_grid,$old_best_distance,$conn);
                            $checkFlag = $return_array[0];
                            $total_distance = $return_array[1];
                            $new_data = $return_array[2];
                            $delivery_array = $return_array[3];
                            $pickup_array = $return_array[4];
                            $datacheck['count'] = $datacheck['count'] + 1;

                            if(($checkFlag == 0)&&($total_distance < $old_best_distance)){
                                $return_best = oneToOneExchange($datacheck,$distance_grid,$conn);
                                if($total_distance < $check_distance){
                                    $best_list = $return_best['route'];
                                    $best_delivery_total = $return_best['delivery_total'];
                                    $best_pickup_total = $return_best['pickup_total'];
                                    $check_distance = $return_best['total_distance'];
                                }
                            }
                        }
                    }
                }
            }
        }
    }
}

```

Figure A6: Program code for Local 1-to-0 Swap


```

public function actionRandom1(){
    $max_loop = $update2['max_loop'];
    if ($max_loop <= 30){
        $random_check = 0;
        do{
            $oldDistance = $do2['distance'];
            $totalRow = $check2['totalRow'];
            $randomRow1 = rand(1,$totalRow);
            do{
                $randomRow2 = rand(1,$totalRow);
            }while($randomRow1 == $randomRow2);
            $randomCol1 = rand(1,(sizeof($old_list[$randomRow1]) - 2));
            $randomCol2 = rand(1,(sizeof($old_list[$randomRow2]) - 2));
            $valueA = $old_list[$randomRow1][$randomCol1];
            for ($i=1;$i<=$totalRow;$i++){
                $no = 0;
                for($j=0; $j<(sizeof($old_list[$i])); $j++){
                    $value = $old_list[$i][$j];
                    if($i == $randomRow1){
                        if($j != $randomCol1){
                            $dataList[$i][$no] = $value;
                            $no = $no + 1;
                        }
                    }
                    if($i == $randomRow2){
                        if($j == $randomCol2){
                            $dataList[$i][$no] = $valueA;
                            $no = $no + 1;
                        }
                        $dataList[$i][$no] = $value;
                        $no = $no + 1;
                    }
                    if($j != $randomCol2){
                        $dataList[$i][$no] = $value;
                        $no = $no + 1;
                    }
                }
            }
            if(($i != $randomRow1)&&($i != $randomRow2))
                $dataList[$i][$j] = $value;
        }
    }
}

```

Figure A9: Program code for Random multiple 1-to-0 Swap

```

public function actionRandom22(){
    $random_check = 0;
    do{
        $oldDistance = $do2['distance'];
        $totalRow = $check2['totalRow'];
        $data_count = 0;
        for ($i = 1; $i<=$totalRow; $i++){
            $data = $data2['up_list'];
            $dataList[$i] = $data;
            if ((sizeof($dataList[$i])- 3) > 0){
                $dataset[$data_count] = $i;
                $data_count ++;
            }
        }
        $randomRow1 = $dataset[array_rand($dataset, 1)];
        do{
            $randomRow2 = $dataset[array_rand($dataset, 1)];
        }while($randomRow1 == $randomRow2);
        if($dataset > 3){
            do{
                $randomRow3 = $dataset[array_rand($dataset, 1)];
            }while(($randomRow3 == $randomRow1)||($randomRow3 == $randomRow2));
        }
        if($dataset <= 3)
            $randomRow3 = $randomRow2;

        $randomColA1 = rand(1,(sizeof($dataList[$randomRow1]) - 2));

        do{
            $randomColA2 = rand(1,(sizeof($dataList[$randomRow1]) - 2));
        }while($randomColA1 == $randomColA2);

        $randomCol2 = rand(1,(sizeof($dataList[$randomRow2]) - 2));
        $randomCol3 = rand(1,(sizeof($dataList[$randomRow3]) - 2));

        $value1a = $dataList[$randomRow1][$randomColA1];
        $value1b = $dataList[$randomRow1][$randomColA2];
        $value2 = $dataList[$randomRow2][$randomCol2];
        $value3 = $dataList[$randomRow3][$randomCol3];

        $dataList[$randomRow1][$randomColA1] = $value2;
        $dataList[$randomRow1][$randomColA2] = $value3;
        $dataList[$randomRow2][$randomCol2] = $value1a;
        $dataList[$randomRow3][$randomCol3] = $value1b;
    }
}

```

Figure A10: Program code for Random 2-to-2 Swap

```

function randomSolution($datacheck,$distance_grid,$conn)
{
    $set_num = $total_num/sizeof($datacheck['route']);
    $random_num = rand($set_num, 2);

    for($i = 0; $i < sizeof($datacheck['route']); $i++)
    {
        $update_record[$i] = $row['data_id'];
        $return_list = $row['data_list'] ;
        $return_distance = $row['total_distance'] ;
        $return_delivery_total = $row['delivery_total'] ;
        $return_pickup_total = $row['pickup_total'] ;

        $datacheck['route'][$i+1] = explode ( ' | ', $return_list );
        $datacheck['delivery_total'][$i+1] = $return_delivery_total;
        $datacheck['pickup_total'][$i+1] = $return_pickup_total;
        $datacheck['total_distance'] = $return_distance;
        $datacheck['loop_checking'] = 0;
    }
}

```

Figure A11: Program code for Random Pick process

Appendix C: Data output

Table A1: Data output for VRPB

(Data outputs for the final obtained result as listed in Table 4.6 under Section 4.6.2)

Dataset	Data output
A1	0 6 0 0 8 20 7 5 23 0 0 16 2 13 0 0 10 3 22 0 0 19 17 21 0 0 1 14 18 0 0 4 15 12 25 0 0 11 9 24 0
A2	0 13 2 1 14 18 0 0 6 20 8 12 25 0 0 4 19 15 17 21 0 0 10 3 22 0
A3	0 13 18 14 1 2 11 16 9 24 0 0 12 25 0 0 4 19 15 17 21 0 0 6 10 3 8 20 7 5 23 22 0
A4	0 12 4 19 15 17 21 0 0 13 18 14 1 2 11 16 9 24 0 0 6 10 3 8 20 7 5 23 22 25 0
B1	0 4 5 7 2 0 0 6 1 20 30 10 15 0 0 18 14 11 24 23 0 0 12 8 13 27 22 0 0 3 25 9 0 0 17 16 19 29 26 0 0 21 28 0
B2	0 6 15 20 1 10 30 28 0 0 4 9 25 0 0 2 7 5 12 13 8 22 27 0 0 3 18 14 11 24 23 0 0 16 17 19 29 26 21 0
B3	0 11 3 18 14 25 0 0 16 17 19 20 15 6 1 10 30 24 23 28 0 0 2 7 5 4 9 12 8 13 27 22 29 26 21 0
C1	0 6 9 37 25 31 0 0 12 4 16 39 23 28 24 0 0 8 18 7 17 21 0 0 2 20 11 26 29 40 0 0 1 15 3 22 27 35 0 0 19 13 36 34 0 0 14 5 10 33 38 32 30 0
C2	0 8 18 19 13 21 36 34 31 0 0 2 15 14 5 10 33 38 32 27 30 0 0 1 12 4 16 39 23 28 24 35 0 0 7 17 11 20 3 22 40 29 26 0 0 6 9 37 25 0

C3	0 2 7 17 11 20 3 15 22 40 29 26 0 0 9 4 16 39 23 28 24 0 0 6 0 0 1 12 14 5 10 33 38 32 27 30 35 0 0 8 18 19 13 21 31 34 36 25 37 0
C4	0 9 4 16 39 23 28 24 0 0 1 12 14 5 10 33 38 32 27 30 35 0 0 6 8 18 19 13 21 31 34 36 25 37 0 0 2 7 17 11 20 3 15 22 40 29 26 0
D1	0 7 0 0 26 10 37 0 0 22 3 1 0 0 23 15 35 34 0 0 21 28 14 0 0 13 6 29 0 0 4 19 9 31 0 0 20 8 32 0 0 2 17 25 0 0 11 27 30 0 0 5 16 38 0 0 12 18 24 36 33 0
D2	0 25 10 26 37 0 0 11 27 30 0 0 13 19 9 31 0 0 14 28 21 0 0 7 0 0 4 12 18 24 36 33 0 0 23 15 35 34 0 0 3 1 8 32 0 0 29 6 22 0 0 5 16 38 0 0 17 2 20 0
D3	0 7 29 13 6 22 0 0 23 15 5 16 38 35 34 0 0 25 17 30 27 11 0 0 2 20 8 1 3 32 0 0 21 26 10 37 0 0 4 19 9 31 36 33 0 0 12 18 24 14 28 0
D4	0 14 28 23 15 5 16 38 35 34 0 0 21 26 10 37 0 0 25 17 11 27 30 2 20 0 0 7 4 19 9 12 18 24 36 33 31 0 0 29 13 6 22 3 1 8 32 0
E1	0 19 18 8 6 22 45 0 0 13 16 30 20 26 10 33 37 39 0 0 29 7 25 17 0 0 4 21 1 12 0 0 9 24 15 36 42 34 31 44 0 0 28 2 23 3 40 41 35 0 0 27 11 5 14 38 32 43 0

E2	0 13 19 18 8 6 22 9 24 15 36 42 34 31 44 0
	0 29 28 2 23 3 40 41 35 45 0
	0 7 17 25 27 11 5 14 38 32 43 0
	0 12 1 21 4 16 30 20 26 10 33 37 39 0
E3	0 13 16 30 20 26 10 33 37 39 0
	0 4 21 1 12 27 11 5 14 38 32 43 0
	0 19 18 8 6 22 9 24 15 36 42 34 31 44 0
	0 29 28 17 25 7 2 23 3 40 41 35 45 0
K1	0 15 23 18 59 28 33 42 20 83 100 95 96 91 113 103 94 0
	0 4 3 49 48 52 51 54 109 98 89 0
	0 35 16 30 1 57 41 68 110 77 0
	0 12 65 40 9 75 45 14 88 101 0
	0 39 37 6 47 8 32 22 5 87 76 92 90 108 0
	0 44 67 50 72 56 10 43 26 105 0
	0 70 53 71 13 34 74 2 73 84 111 0
	0 55 7 38 31 25 17 81 78 80 104 107 0
	0 24 66 27 11 36 61 46 21 85 112 86 106 82 0
	0 58 69 63 60 29 19 64 62 93 102 99 97 79 0
K2	0 3 49 16 30 72 56 10 43 26 50 105 0
	0 70 71 45 9 40 33 42 20 62 64 93 83 88 101 0
	0 55 66 46 21 38 31 7 25 17 81 82 106 86 85 112 78 107 0
	0 4 35 67 44 24 27 11 36 5 61 104 80 87 76 0
	0 12 53 75 65 54 28 59 18 51 52 113 91 109 98 96 95 100 89 0
	0 39 58 60 14 19 29 13 34 74 84 99 102 97 79 111 0
	0 15 23 48 68 1 57 41 103 94 110 77 0
	0 37 47 8 32 22 6 69 63 73 2 92 90 108 0
K3	
	0 24 66 25 17 81 86 106 82 0
	0 39 37 6 47 8 32 22 5 36 11 107 80 104 87 76 0
	0 70 53 45 71 14 93 83 88 101 0
	0 23 15 54 52 51 18 59 28 91 109 98 96 95 100 89 0
	0 55 7 31 38 43 10 50 44 0
	0 4 3 35 67 16 48 49 0
	0 12 65 75 9 40 33 42 20 62 64 13 84 92 90 108 0
	0 58 60 69 63 2 73 74 34 29 19 102 99 97 79 111 0
	0 68 41 57 1 30 72 56 26 27 61 46 21 85 112 78 105 77 110 94 103 113 0
K4	
	0 70 58 69 108 0
	0 4 3 35 67 16 41 18 59 51 52 23 0
	0 12 53 75 45 64 19 29 73 2 63 60 15 54 28 96 95 100 83 93 88 101 0
	0 65 40 9 33 42 20 62 14 71 13 34 74 84 111 0
	0 44 26 10 43 38 31 7 25 66 17 81 78 80 104 87 76 90 92 99 102 97 79 0
	0 39 37 6 47 8 32 22 5 61 46 21 56 72 30 77 110 94 103 113 91 0
	0 49 48 68 57 1 50 55 24 27 11 36 107 112 85 86 106 82 105 109 98 89 0

Table A2: Data output for m-VRPB

(Data outputs for m-VRPB solution as listed in Table 5.3 under Section 5.3.1)

Dataset	Data output
A1	0 17 15 25 0 0 12 4 19 21 0 0 6 0 0 16 13 2 0 0 10 3 22 0 0 9 24 11 0 0 1 14 18 0 0 8 20 7 23 5 0
A2	0 4 15 17 19 21 0 0 25 12 8 20 6 0 0 13 2 18 14 1 0 0 10 3 22 0 0 16 11 24 9 5 23 7 0
A3	0 3 8 20 7 5 23 9 24 16 0 0 11 13 2 1 14 18 0 0 15 17 19 4 21 0 0 25 12 22 10 6 0
B1	0 3 18 23 0 0 7 4 5 2 21 0 0 6 15 10 1 30 20 29 0 0 16 19 17 26 0 0 28 0 0 12 8 13 22 27 0 0 9 25 14 11 24 0
B2	0 3 18 14 11 24 23 0 0 7 12 27 16 13 8 22 29 21 0 0 17 6 15 10 30 1 20 19 26 0 0 25 9 4 5 2 0 0 28 0
B3	0 21 23 24 11 28 6 15 20 1 30 10 0 0 2 4 3 18 14 25 9 5 0 0 26 16 17 19 29 22 8 13 27 12 7 0
C1	0 17 7 20 11 26 29 0 0 14 5 10 33 38 0 0 4 12 28 23 16 39 24 0 0 6 13 36 34 0 0 2 40 3 22 15 32 27 0 0 9 37 25 31 0 0 8 18 19 21 1 30 35 0
C2	0 6 8 7 17 11 20 26 29 40 35 0 0 32 15 38 14 5 33 10 0 0 2 3 22 27 30 12 28 24 0 0 1 4 23 16 39 25 37 9 0 0 34 18 19 21 13 36 31 0

C3	0 8 18 19 21 13 31 34 36 25 37 9 0 0 4 24 1 12 28 23 16 39 14 5 38 10 33 0 0 7 17 2 3 40 29 26 11 20 22 15 32 0 0 6 0 0 30 27 35 0
D1	0 2 30 27 11 0 0 21 28 0 0 4 12 31 9 19 0 0 17 25 0 0 15 23 34 0 0 20 32 8 1 0 0 26 10 37 0 0 7 0 0 5 16 38 35 0 0 29 6 13 0 0 14 24 33 18 36 0 0 22 3 0
D3	0 4 29 13 7 0 0 26 35 5 16 38 24 33 12 0 0 9 19 31 18 36 14 28 0 0 27 11 30 8 1 0 0 25 17 0 0 21 15 23 34 10 37 0 0 2 20 32 3 22 6 0
D4	0 13 6 3 1 8 22 29 0 0 7 28 38 16 5 35 26 10 0 0 25 32 30 27 11 17 2 20 0 0 15 23 34 21 37 0 0 12 33 14 18 24 36 31 9 19 4 0
E1	0 13 39 30 20 10 33 26 37 16 0 0 18 19 8 6 22 45 34 31 44 0 0 27 11 5 38 14 32 1 0 0 4 21 12 43 0 0 42 36 15 24 9 0 0 28 2 41 3 23 40 35 0 0 29 7 25 17 0
E2	0 28 2 3 23 40 41 35 8 6 31 44 0 0 18 19 22 45 9 24 34 42 15 20 33 10 37 26 16 39 30 36 0 0 7 27 11 5 14 32 38 1 21 0 0 13 4 43 12 25 17 29 0
E3	0 19 18 8 6 22 45 9 24 15 20 33 10 26 37 14 41 0 0 2 23 0 0 28 7 25 17 27 11 5 1 3 0 0 29 13 44 31 34 42 36 30 39 16 4 21 43 12 38 32 40 35 0

Table A3: Data output for m-VRPB with vehicle relaxation

(Data outputs for m-VRPB (Unfixed vehicle number) as listed in Table 5.4 under Section 5.3.2)

Dataset	Data output
A1	0 25 15 17 21 0 0 11 24 5 7 20 0 0 10 3 22 0 0 13 2 16 0 0 18 14 1 8 0 0 19 4 12 0 0 9 23 6 0
A2	0 4 15 17 19 21 0 0 25 12 8 20 6 0 0 13 2 18 14 1 0 0 10 3 22 0 0 16 11 24 9 5 23 7 0
A3	0 10 3 8 7 23 20 9 5 6 0 0 25 12 22 16 11 24 2 13 0 0 17 19 4 15 21 1 14 18 0
B1	0 4 5 7 2 0 0 6 1 20 30 10 15 0 0 18 14 11 24 23 0 0 12 8 13 27 22 0 0 3 25 9 0 0 17 16 19 29 26 0 0 21 0 0 28 0
B2	0 6 15 20 1 10 30 28 0 0 4 9 25 0 0 2 7 5 12 13 8 22 27 0 0 3 18 14 11 24 23 0 0 16 17 19 29 26 21 0 0 2 7 5 12 13 8 22 27 0 0 4 9 25 0 0 16 17 19 29 26 21 0 0 6 15 20 1 10 30 28 0 0 3 18 14 11 24 23 0
B3	0 23 18 14 25 9 5 12 29 26 19 17 0 0 3 11 24 10 30 20 15 1 6 28 0 0 16 22 8 13 27 4 7 2 21 0
C1	0 1 5 14 38 12 30 35 0 0 18 19 31 39 16 23 28 0 0 2 3 22 40 29 0 0 7 17 11 20 26 21 8 0 0 6 13 36 34 0 0 4 24 9 37 25 0 0 27 15 10 33 32 0

C2	0 6 8 18 19 21 11 26 29 40 20 0 0 9 37 4 12 5 14 30 35 0 0 2 3 22 15 32 38 33 10 27 0 0 24 39 16 23 28 1 0 0 7 17 13 36 25 31 34 0
C3	0 6 8 18 19 21 13 36 34 31 25 9 0 0 27 32 38 14 5 33 10 15 22 3 0 0 2 29 40 26 11 20 7 17 0 0 37 16 39 23 28 24 4 12 1 30 35 0
D1	0 21 26 16 38 0 0 23 15 34 35 37 0 0 19 9 31 14 0 0 13 29 7 0 0 8 1 32 20 0 0 4 12 18 24 36 33 0 0 28 5 0 0 10 30 25 17 0 0 11 27 2 0 0 3 22 6 0
D3	0 1 8 3 17 25 0 0 28 14 5 16 34 35 38 0 0 7 13 29 6 22 0 0 21 15 23 26 37 0 0 4 9 19 31 12 18 36 24 33 0 0 10 11 27 30 32 20 2 0
D4	0 13 6 29 8 32 18 31 9 19 4 0 0 23 16 5 28 33 12 36 24 38 14 0 0 25 30 27 11 10 26 15 21 35 34 37 0 0 20 2 17 1 22 3 7 0
E1	0 13 36 30 16 33 20 10 26 37 39 0 0 19 18 8 6 22 45 35 41 31 44 0 0 14 11 5 38 32 1 4 0 0 21 43 12 29 0 0 34 42 15 24 9 0 0 2 23 3 40 27 0 0 28 17 25 7 0
E2	0 18 8 6 45 22 34 15 24 9 31 44 13 0 0 19 28 7 11 14 5 27 38 32 0 0 2 41 3 23 35 40 25 17 29 0 0 12 43 21 1 4 42 36 10 37 26 20 33 3...
E3	0 16 26 37 10 4 21 1 33 18 6 8 45 22 36 20 30 39 43 0 0 19 7 11 38 32 5 14 27 12 17 25 0 0 13 29 28 40 35 2 41 3 23 24 9 15 42 34 44 31 0