

ปัญหาการรับส่งสินค้าแบบสถิติด้วยรถหลายคัน แบบหนึ่งต่อหนึ่ง และหลายจุดจอด

นายเปาโล เอียน คาซิพิต ลูเซโร



จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

บทคัดย่อและแฟ้มข้อมูลฉบับเต็มของวิทยานิพนธ์ตั้งแต่ปีการศึกษา 2554 ที่ให้บริการในคลังปัญญาจุฬาฯ (CUIR)

เป็นแฟ้มข้อมูลของนิสิตเจ้าของวิทยานิพนธ์ ที่ส่งผ่านทางบัณฑิตวิทยาลัย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

The abstract and full text of theses from the academic year 2011 in Chulalongkorn University Intellectual Repository (CUIR) are the thesis authors' files submitted through the University Graduate School.

สาขาวิชาวิศวกรรมโยธา ภาควิชาวิศวกรรมโยธา
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2559

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

STATIC ONE-TO-ONE MULTI-VEHICLE PICKUP AND DELIVERY PROBLEM WITH MULTIPLE
DEPOTS

Mr. Paolo Ian Casipit Lucero



A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering Program in Civil Engineering

Department of Civil Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2016

Copyright of Chulalongkorn University

Thesis Title	STATIC ONE-TO-ONE MULTI-VEHICLE PICKUP AND DELIVERY PROBLEM WITH MULTIPLE DEPOTS
By	Mr. Paolo Ian Casipit Lucero
Field of Study	Civil Engineering
Thesis Advisor	Assistant Professor Manoj Lohatepanont, Sc.D.

Accepted by the Faculty of Engineering, Chulalongkorn University in Partial Fulfillment of the Requirements for the Master's Degree

.....Dean of the Faculty of Engineering
(Associate Professor Supot Teachavorasinskun, D.Eng.)

THESIS COMMITTEE

.....Chairman
(Associate Professor Kasem Choocharukul, Ph.D.)

.....Thesis Advisor
(Assistant Professor Manoj Lohatepanont, Sc.D.)

.....Examiner
(Associate Professor Saksith Chalermpong, Ph.D.)

.....External Examiner
(Professor Alexis Morales Fillone, Ph.D.)

.....External Examiner
(Associate Professor Kenetsu Uchida, Ph.D.)

เปาโล เอียน คาซิพิต ลูเซโร : ปัญหาการรับส่งสินค้าแบบสถิตย์ด้วยรถหลายคันแบบหนึ่งต่อหนึ่ง และหลายจุดจอด (STATIC ONE-TO-ONE MULTI-VEHICLE PICKUP AND DELIVERY PROBLEM WITH MULTIPLE DEPOTS) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: ผศ. ดร. มาโนช โลหเตปานนท์, 85 หน้า.

งานวิจัยนี้นำเสนอปัญหาการรับและส่งสินค้าในรูปแบบสถิตย์จากหลายศูนย์กระจายสินค้าย่อย และเสนอวิธีการแก้ปัญหาโดยการจัดกลุ่มและแบ่งกลุ่มคำร้องขอส่งสินค้าในแต่ละเส้นทางการขนส่ง การหาผลเฉลยของปัญหา ทางผู้วิจัยได้เลือกใช้โปรแกรม GAMS และ CPLEX เป็นเครื่องมือในการหาผลเฉลย วิธีการคำนวณต้นทุนค่าขนส่งจากจุดยอดไปยังปลายทางพิจารณาจากระยะทางแบบยูคลิเดียน (Euclidean Distance) เปรียบเทียบระหว่างวิธี best-bound search และ depth-first search แบบบรันคแอนด์บาวด์ (branch-and-bound) ในการศึกษานี้ได้แสดงผลการวิเคราะห์ที่ได้จากโปรแกรม GAMS ซึ่งผลเฉลยที่ดีที่สุดพบว่าเส้นทางที่มีคำร้องขอแบบรวมกันสามารถลดระยะการเดินทางได้อย่างมีนัยสำคัญ เมื่อเปรียบเทียบกับวิธีการส่งสินค้าแบบหนึ่งเส้นทางหนึ่งคำร้องขอ และมีจำนวนคำร้องขอในหนึ่งเส้นทางเพิ่มมากขึ้น นอกจากนี้พบว่าอิทธิพลของจำนวนและการกระจายตัวสถานีจุดเริ่มต้นมีผลที่ลดลง ผลการศึกษาดังกล่าวแสดงถึงประโยชน์ในการประยุกต์ใช้งานโปรแกรม GAMS เพื่อแก้ปัญหาการรับส่งสินค้าแบบสถิตย์จากหลายศูนย์กระจายสินค้า ซึ่งถือว่าเป็นเครื่องมือช่วยในการตัดสินใจที่มีประโยชน์อย่างมากสำหรับบริษัทที่ทำธุรกิจด้านขนส่งและโลจิสติกส์

ภาควิชา วิศวกรรมโยธา

ลายมือชื่อนิสิต

สาขาวิชา วิศวกรรมโยธา

ลายมือชื่อ อ.ที่ปรึกษาหลัก

ปีการศึกษา 2559

5770522821 : MAJOR CIVIL ENGINEERING

KEYWORDS: THE MESSENGER PROBLEM / MULTI-DEPOT PICKUP AND DELIVERY PROBLEM

PAOLO IAN CASIPIT LUCERO: STATIC ONE-TO-ONE MULTI-VEHICLE PICKUP AND DELIVERY PROBLEM WITH MULTIPLE DEPOTS. ADVISOR: ASST. PROF. MANOJ LOHATEPANONT, Sc.D., 85 pp.

The research describes a static case of the pickup and delivery problem with multiple depots and proposes a solution approach that forces a combination of requests in a particular route. To solve the problem, the model is implemented using General Algebraic Modelling System (GAMS) with CPLEX as the solver. Euclidean distances are considered as the total cost between nodes. Instances based on existing studies are used for computational experiments. Best-bound search and depth-first search methods of the branch-and-bound algorithm are also compared. Results of the trials implemented in GAMS are presented in the study. The optimal solutions found with routes having combined requests show significant distance reduction compared to the typical individual service of one request per route. Results also show that as the number of allowed requests in one route is increased, the effect of the number and variation of depots also decreases. The main contribution of this research are applications in GAMS that can solve the static case of the messenger problem in a multi-depot variation, a setup with few available literature. The study can be beneficial for logistics companies especially those of which specializing in pickup and delivery services.

Department: Civil Engineering

Student's Signature

Field of Study: Civil Engineering

Advisor's Signature

Academic Year: 2016

ACKNOWLEDGEMENTS

Being able to take up my master's degree abroad is an experience that has surely taught me things beyond I could ever imagine. First of all, I would like to thank AUN/Seed-Net for granting me a scholarship and for giving me the opportunity to study in one of the most prestigious universities in Thailand, Chulalongkorn University. Second, I would like to express my sincerest gratitude to my research advisor, Asst. Prof. Manoj Lohatepanont, Sc. D., for his invaluable help, endless encouragement and utmost patience throughout the development of my research. I also would like to sincerely thank Assoc. Prof. Sorawit Narupiti, Ph. D. and Assoc. Prof. Kasem Choocharukul, Ph. D. for making the continuation of this research possible. Third, I would like to thank Assoc. Prof. Kenetsu Uchida, Ph. D., Assoc. Prof. Saksith Chalermpong, Ph. D. and Prof. Alexis Morales Fillone, Ph. D. for being part of my panel and giving me advice to improve my research. Many thanks, too, to Mr. Nutchapon Thanasathit and Mr. Ponn Chalermwongsawej for helping me understand the algebraic modelling language that became useful to the study.

My sincere gratitude is also extended to the friends that I met along the way who shared with me a roller coaster ride of adventures in Thailand. I would have never made it until the end without each and everyone of you. Thank you to my transportation engineering classmates, to the best seniors ever who have constantly guided me along the way, to my friends that I met in my one-semester stay in Hokkaido University, to my Filipino friends especially my room mates who have become my family, to the UST interns who have become a special part of my Bangkok life, and to every friend, every person I met who never got tired of telling me that I can do it, thank you for believing in me. Finally, I would like to thank my family, the biggest source of my strength which kept me going during the most difficult times of my life living abroad. To Papa, Mama, Gelo, Jugene, Rain and Nanay, this is for you. And the most important acknowledgement of all, finishing this research would not have been possible if not for You, Lord God. My master's degree is all for Your glory. Thank You.

CONTENTS

	Page
THAI ABSTRACT	iv
ENGLISH ABSTRACT	v
ACKNOWLEDGEMENTS	vi
CONTENTS	vii
LIST OF FIGURES	ix
LIST OF TABLES	x
1 INTRODUCTION	1
1.1 Background of the Study	1
1.2 Problem Description	4
1.3 Research Objectives	7
1.4 Motivation for the Multiple-Depot Setup	7
1.5 Scope of Study	9
1.6 Expected Benefits	9
2 LITERATURE REVIEW	10
2.1 PDP in General	10
2.2 Classifications of the PDP	14
2.3 One-to-One PDP	18
2.4 The Messenger Problem	20
2.5 Research Gap	22
3 METHODOLOGY	24
3.1. Overall Research Framework	24
3.2. Formulation of Model	24

	Page
3.3. Modified Instances	29
3.4. General Algebraic Modelling System (GAMS)	35
3.5. Assumptions for Computational Experiments.....	37
4 RESULTS AND DISCUSSION.....	38
4.1 Tested Instances.....	38
4.2 Model Performance	41
4.2.1 Instances with six requests	41
4.2.2 Instances with twelve requests	49
4.3 Improvement through Combining Requests.....	57
4.3.1 Instances with six requests	57
4.3.2 Instances with twelve requests	59
4.4 Multiple-Depot Variation.....	62
4.5 Advantages and Disadvantages of Proposed Model.....	65
5 SUMMARY AND CONCLUSION.....	68
5.1 Summary.....	68
5.2 Conclusion	69
5.3 Recommendations	71
REFERENCES	72
VITA.....	85

LIST OF FIGURES

Figure 1-1 An illustrated sample problem	5
Figure 1-2 An optimal solution to the illustrated sample problem	6
Figure 1-3 Motorcycle taxis in Bangkok (The Thailand Life, 2013)	8
Figure 2-1 Classifications of pickup and delivery problems (Parragh et al., 2008b)	15
Figure 2-2 Static pickup and delivery problem classification (Berbeglia et al., 2007) ..	17
Figure 3-1 Overall design flow of research.....	25
Figure 3-2 Plot of sample modified instance with six requests.....	32
Figure 3-3 Plot of sample modified instance with twelve requests	34
Figure 3-4 Variation of instances containing three depots.....	34
Figure 3-5 Summary of depot variation with the instances.....	35
Figure 3-6 Structure of GAMS model (Chattopadhyay, 1999)	36
Figure 4-1 Summary of CPU run time results for instances with six requests under best-bound search	47
Figure 4-2 Summary of CPU run time results for instances with six requests under depth-first search.....	48
Figure 4-3 Distance reduction results from computational experiments	62
Figure 4-4 Effect of multiple-depot variation on instances having six requests (depth-first search)	63
Figure 4-5 Effect of multiple-depot variation on instances having twelve requests (depth-first search)	64

LIST OF TABLES

Table 1.1 Several pickup and delivery services in Bangkok	3
Table 3.1 Sample instance from Ropke et al. (2007).....	30
Table 3.2 Sample modified instance with six requests	31
Table 3.3 Sample modified instance with twelve requests	33
Table 4.1 Characteristics of instances with six requests.....	39
Table 4.2 Characteristics of instances with twelve requests.....	40
Table 4.3 Summary of results for instances with six requests (A6).....	44
Table 4.4 Summary of results for instances with six requests (B6).....	45
Table 4.5 Summary of results for instances with six requests (C6)	46
Table 4.6 Summary of results for instances with six requests (A12)	52
Table 4.7 Summary of results for instances with six requests (B12)	53
Table 4.8 Summary of results for instances with six requests (C12)	54
Table 4.9 Summary of model performance results for instances with 12 requests....	55
Table 4.10 Average percentage of gaps from lower bounds of 12-request instances.....	56
Table 4.11 Average percentage of distance reduction of combined requests compared to ‘one request per route’ service (A6, B6, and C6).....	58
Table 4.12 Summary of distance reduction results of six-request instances	59
Table 4.13 Average percentage of distance reduction of combined requests compared to ‘one request per route’ service (A12, B12, and C12)	60
Table 4.14 Summary of distance reduction results of twelve-request instances.....	61

CHAPTER I

INTRODUCTION

1.1 Background of the Study

Recognized for its manufacturing capabilities and numerous multinational company investments, Thailand is continuously growing to be a key logistics hub in Mainland Southeast Asia (Yuen, 2015). Moreover, in light of the recently launched ASEAN Economic Community (AEC) in 2015, the country's trade and commerce industry is expected to grow further. This means more growth potential for small and large businesses alike, more movement of goods inside Thailand's major urban city center- Bangkok and thus, more demand for a sufficient and convenient transportation system of commodities. Particularly, in the field of logistics and commodity movement, this research focuses on a case of a pickup and delivery problem. Pickup and delivery services have become a common option in transportation of commodities which may also be attributed to the continuous rise of electronic commerce (e-commerce) (Ekvitthayavechnukul, 2016). More and more transactions are becoming easier and more convenient as businesses keep up with technological advancements and services become available online.





In terms of pickup and delivery services, the growing demand for such services can be observed from offerings of different logistics companies in recent years. In March 2015, *Grab*, a company specializing in transportation services such as *Grab Taxi* and *Grab Car*, offered a next day delivery service in the form of *Grab Express*. The service assures pickup within three hours limited to some areas in Bangkok and

deliveries are made the next day. More recently in August 2015, an instant parcel and delivery service, also by *Grab*, was launched in the form of *Grab Bike*. The service claims to complete pickup and delivery requests within one hour inside Bangkok. Another pickup and delivery service by *Kerry Express* started in June 2015 in the form of *Bangkok Sameday*. The service focuses on documents and parcel delivery that guarantees a two-hour pickup and same-day delivery within areas in Bangkok. An almost similar kind of service was also offered in 2014 by *Skootar Beyond Co. Ltd*, a start-up company that developed a messenger service accommodating documents, invoices and small parcel delivery in Bangkok and suburban areas. The price of the service depended on actual distances between service points. On the other hand, *Classic Express Services (CES)* have also offered the same type of service since January 2012 called *Thailand Door-to-Door Parcel Delivery: City Express* that deals with same-day delivery requests ranging from document to multiple parcels, also within areas in Bangkok. Comparing the service of CES to *Grab*, *Kerry Express*, and *Skootar*, their offered services are calculated by weight and not by shipment with regard to actual distance. It is observed that through the years, improvements for the said type of pickup and delivery service are continuous and thus prices are also becoming more competitive among logistics companies. Some of the costs are summarized in Table 1.1. The figures presented are based on the available information on the website of each respective company as of September 2016.

Typically, the express services serve each request individually corresponding to one vehicle serving one request, depending on the nearest vehicle available and the time window required for each request to be completed. If all individual requests are considered as a whole, the overall cost can be huge. This research in turn focuses on

formulating a routing model for the said type of pickup and delivery services that combines two or more requests into a single route. The study would consider a multiple-depot setup with vehicles in each depot instead of the usual nearest-vehicle approach done in pickup and delivery services. Computational experiments are performed and the resulting solutions are compared using hypothetical networks from previous researches related to the study.

Table 1.1 Several pickup and delivery services in Bangkok

Logistics Company	Name of Service	Location	Cost
	<i>Thailand Door-to-Door Parcel Delivery: City Express</i>	Bangkok (Zone A Downtown, Zone B Surrounding)	First 1.0kg: 180-220THB, Next 1.0kg: 20THB
	<i>Skootar Messenger Service</i>	Bangkok and suburban areas	Standard Price: 70 THB plus 10 THB per additional kilometre
	<i>Bangkok Sameday</i>	Bangkok Metropolitan (16 Districts)	30THB minimum per shipment
	<i>GrabExpress: Next Day Delivery Service</i>	Bangkok, Nonthaburi, Pratumtani, and Samut Prakan	30THB - 55THB based on dimensions 70THB - 120THB in other provinces
	<i>GrabBike: Instant Parcel and Delivery Service</i>	Bangkok, Nonthaburi, Pratumtani, and Samut Prakan	20THB starting fare, 9THB per additional kilometer

Source: ceslogistics.com; skootar.com; kerryexpress.com; grab.com

1.2 Problem Description

Before proceeding, the following terms are defined first as to how they are used in this research. These important terms are enumerated as follows:

- Static - all data such as pickup and delivery locations, distance and travel time between points, and other related data are known beforehand.
- One-to-one - considering the pickup and delivery requests, each node will only be allowed to be either a pickup point or a delivery point. One delivery node is a specific destination for a certain pickup origin. All requests will be in the form of pairs.
- Depot - no items are present in the depot. The word depot would only serve as a station where the vehicles would come from.
- Request - refers to the request made by a certain customer. One request is equivalent to a combination of one pickup request and one delivery request.
- Route - the sequence of serving the demands, a route would contain the requests and will start and end in the same depot.
- Multi-vehicle - more than one vehicle will be allowed for usage in one depot.

Given a set of pickup and delivery requests, generating one route to serve all demands may not be practical especially if requests must be served within a certain period of time. The set of requests may then be divided into subsets which can be served by different vehicles coming from different depots. The main goal is to design a set of least cost vehicle routes to satisfy all requests whose locations are already known in advanced. Routes are generated by combining the demands while

simultaneously assigning the most appropriate depot for each route. In Figure 1-1, a sample problem is illustrated.

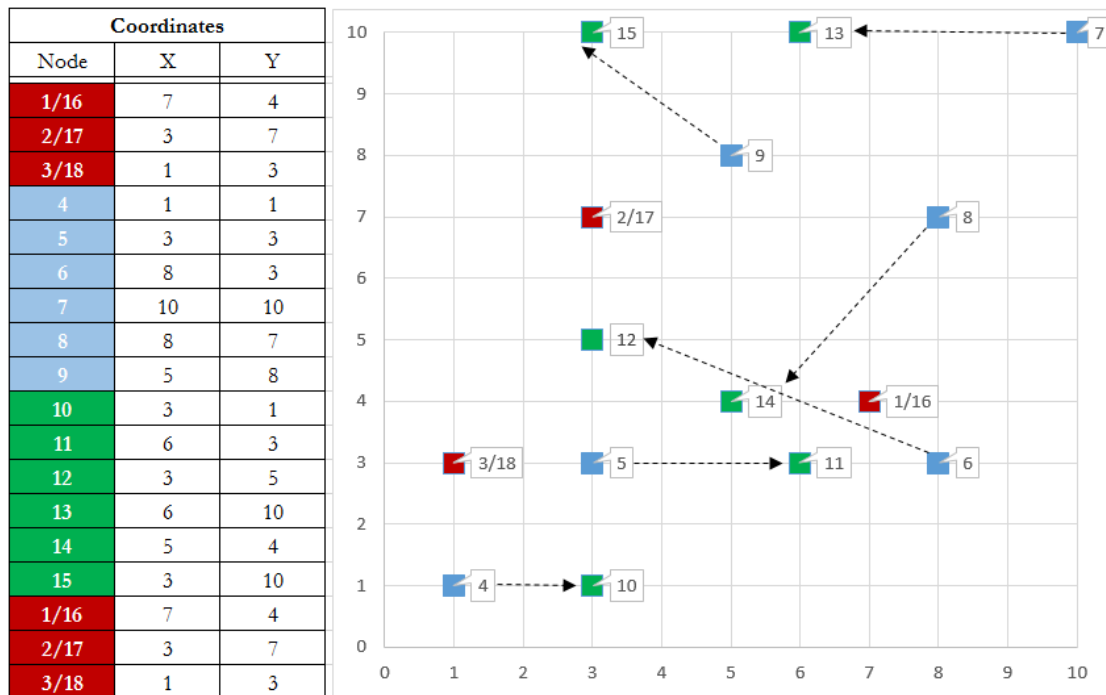


Figure 1-1 An illustrated sample problem

In the sample problem, there are three given depots in red and six given requests: pickup points in blue and delivery points in green. The dashed arrows denote where the pickup point should be delivered. A certain number of requests is allowed in one route which will also determine the number of total routes that will be used in the solution. For example, if six requests are given and two requests are allowed per route, the solution would result into a set of three least cost routes. Each route should start and end in the same depot. More than one vehicle can come from one depot. All given demands should be served and each node should be visited only once. One challenge in this problem is including the two nodes of one request in one route and making sure that the pickup node comes first. In serving one demand, the

routes. Two vehicles came from the first depot while one vehicle came from the second depot. No vehicle was chosen from the third depot.

1.3 Research Objectives

The main objective of this research is to formulate a routing model for the one-to-one static multi-vehicle pickup and delivery problem with multiple depots. The research will have the following specific objectives:

- To design a set of least cost vehicle routes through combining pickup and delivery demands and proper assignment to each of the multiple depots available.
- To analyze the efficiency of the formulated model using a hypothetical network and hypothetical demands from previous studies.

1.4 Motivation for the Multiple-Depot Setup

In Bangkok, one of the most common public transportation modes is done through the use of “motorcycle taxis” as seen from Figure 1-3. These vehicles are present in almost every area in Bangkok and they can usually be found in convenient and accessible, designated stations.

It is observed that these vehicles are mostly active in the peak hours in the morning where people go to school and work as well as in the peak hours in the evening where people head back home. During the middle of the day, less demand for ridership is observed with motorcycle taxis. This availability of the said vehicles may be put into use in the form of a pickup and delivery system especially for small

commodities such as documents and small parcels. An increasing demand for the said type of service may be seen from the recent offerings of various companies in Bangkok such as Classic Express Services, Grab, Skootar, and Kerry Express. Therefore, it is just right to provide more research on the subject matter which could then lead to more innovative logistics ways that do not just minimize costs but provide quality service as well. Since the motorcycle taxis are also present in multiple stations, the situation paves way as a good motivation for a research on a multiple-depot scenario of a pickup and delivery problem, a research area with few available related literature. Moreover, small vehicles such as motorcycles may be advantageous in the city center due to its maneuvering capacity between traffic and its ability to traverse narrow streets. Other possible benefits of the research may be its application to other urban delivery situations such as food deliveries, online shopping, and last-mile delivery problems.



Figure 1-3 Motorcycle taxis in Bangkok (The Thailand Life, 2013)

1.5 Scope of Study

The study mainly focused on the formulation of the static routing model for the one-to-one pickup and delivery problem having multiple depots. The formulated model was implemented in a General Algebraic Modelling System (GAMS) platform alone without the use of other programming languages. For computational experiments, instances were based on the modification of instances provided from previous researches. Since the motivation for the study are motorcycle taxis, only small commodities were assumed and thus, vehicle capacity was not considered. Furthermore, the problem in the study was also classified under the closed loop setup category of pickup and delivery services which means that the route formed should start and end in the same depot.

1.6 Expected Benefits

This study aims to formulate a routing model that can combine multiple requests in one singular route. Upon completion of the model, it is expected that the research can be of use to logistics companies specializing in pickups and deliveries to improve their services through combining requests and by having a multi-depot setup. The study may also be beneficial to real-life applications of new transportation ways in relation with the messenger problem. Other than courier services, real-life applications may also include food delivery services involving multiple restaurants as well as electronic shopping that involves online transactions. Another key benefit of this study is its main academic contribution which is the exploration of the pickup and delivery problem in a multi-depot setup.

CHAPTER II

LITERATURE REVIEW

This chapter discusses several related literature supporting this research. First, the pickup and delivery problem (PDP) and its general applications are reviewed. Second, different types of PDPs are discussed and the classification of the problem in this research is defined. The third part of this chapter reviews related literature about the one-to-one pickup and delivery problem as well as the solution approaches used for the said type of problem. Next to this part is a closely related problem to the study in the form of the 'messenger problem' based from the works of Fabry (2007, 2015) and Fábry and Kobzareva (2012). Lastly, the research gap is discussed to which this paper is based from.

2.1 PDP in General

The Pickup and Delivery Problem (PDP) is an extended case of the Vehicle Routing Problem (VRP) which has been studied for approximately 50 years now. The study dates back from an optimum routing of a fleet of gasoline trucks known as the truck dispatching problem (Dantzig & Ramser, 1959). The research was based on a linear programming formulation with minimum total mileage as the objective. It is considered to be the one that introduced the VRP (Irnich, Toth, & Vigo). In 1964, an iterative procedure for finding optimal routes was introduced by Clarke and Wright (1964). Both mentioned studies were described to be suitable for manual computations. Up to this day, research in the said field have already extended to complicated variations which are impossible to compute by hand. Different

mathematical models have been developed and various algorithms have also been formulated to solve different complications. This interest in solving VRPs can be attributed to the numerous applications that the problems have. Applications range from inter-modal transportation (Bräysy & Hasle), ship routing and scheduling (Christiansen & Fagerholt), disaster relief (Golden, Kovacs, & Wasil) and green vehicle routing (Eglese & Bektaş). Research has also improved alongside technological development especially with computing capabilities of computers, programming platforms, and various advanced software. In this study, the pickup and delivery problem will be in focus.

Pickup and delivery problems can mainly be for transportation of people (Battarra, Cordeau, & Iori) or transportation of goods (Doerner & Salazar-González). One main difference of dealing with transportation of people is that it includes additional factors for consideration such as delay reduction, convenience, and as well as noise and pollution. Dial-a-ride problems (DARP) such as those dealing with healthcare and transportation of handicapped and the elderly (Detti, Papalini, & Lara; Marković, Nair, Schonfeld, Miller-Hooks, & Mohebbi, 2015) are examples of PDPs for transportation of people. For transportation of goods, examples are those dealing with truck deliveries, distribution of beverages, collection of empty containers, as well as urban courier services and retail stores management (Doerner & Salazar-González). In general, the pickup and delivery problem is described as transportation requests that are satisfied through the construction of a set of routes to serve all given demands (M. W. Savelsbergh & Sol, 1995). According to M. W. Savelsbergh and Sol (1995), the General Pickup and Delivery Problem (GPDP) can be formulated as a mathematical program by introducing four variables as follows.

Variables:

$$\begin{aligned}
 z_i^k & \quad (i \in N, k \in M) \\
 x_{ij}^k & \quad ((i, j) \in (V \times V) \cup \{(k^+, j) \mid j \in V\} \cup \{(j, k^-) \mid j \in V\}, k \in M) \\
 D_i & \quad (i \in V \cup W) \\
 y_i & \quad (i \in V \cup W)
 \end{aligned}$$

The variable z_i^k is equal to 1 if transportation request i is assigned to vehicle k and 0 otherwise. Variable x_{ij}^k is equal to 1 if vehicle k travels from location i to location j and 0 otherwise. The departure time is denoted by D_i while y_i represents the load of the vehicle at vertex i . For all $k \in M$, $q_{k^+} = 0$.

Model parameters are summarized and defined as follows:

V	set of all vertices
N	set of transportation requests
N_i^-	set of origins
N_i^+	set of destinations
q_i	load size, positive for pickups and negative for deliveries
M	set of vehicles
M^-	set of end locations
M^+	set of start locations
W	set of all vehicle locations
k	one vehicle ($k \in M$), start location k^+ , end location k^-
d_{ij}	denotes travel distance
t_{ij}	denotes travel time
c_{ij}	denotes travel cost

The objective function $f(x)$ is then subjected to the following constraints.

$$\sum_{k \in M} z_i^k = 1 \quad \text{for all } i \in N \quad (2.1)$$

$$\sum_{j \in V \cup W} x_{lj}^k = \sum_{j \in V \cup W} x_{jl}^k = z_i^k \quad \text{for all } i \in N, l \in N_i^+ \cup N_i^-, k \in M \quad (2.2)$$

$$\sum_{j \in V \cup (k^-)} x_{k^+j}^k = 1 \quad \text{for all } k \in M \quad (2.3)$$

$$\sum_{i \in V \cup (k^+)} x_{ik^-}^k = 1 \quad \text{for all } k \in M \quad (2.4)$$

$$D_{k^+} = 0 \quad \text{for all } k \in M \quad (2.5)$$

$$D_p \leq D_q \quad \text{for all } i \in N, p \in N_i^+, q \in N_i^- \quad (2.6)$$

$$x_{ij}^k = 1 \rightarrow D_i + t_{ij} \leq D_j \quad \text{for all } i, j \in V \cup W, k \in M \quad (2.7)$$

$$y_{k^+} = 0 \quad \text{for all } k \in M \quad (2.8)$$

$$y_l \leq \sum_{k \in M} Q_k z_i^k \quad \text{for all } i \in N, l \in N_i^+ \cup N_i^- \quad (2.9)$$

$$x_{ij}^k = 1 \rightarrow y_i + q_i \leq y_j \quad \text{for all } i, j \in V \cup W, k \in M \quad (2.10)$$

$$x_{ij}^k \in (0, 1) \quad \text{for all } i, j \in V \cup W, k \in M \quad (2.11)$$

$$z_{ij}^k \in (0, 1) \quad \text{for all } i \in N, k \in M \quad (2.12)$$

$$D_i \geq 0 \quad \text{for all } i \in V \cup W \quad (2.13)$$

$$y_i \geq 0 \quad \text{for all } i \in V \cup W \quad (2.14)$$

M. W. Savelsbergh and Sol (1995) summarized all constraints as follows.

Constraint (2.1) assigns each transportation request to exactly one vehicle. In constraint (2.2), a vehicle is only allowed to enter or leave a location l if it is an origin or a destination of the request assigned to the vehicle. Constraints (2.3) and (2.4) ensure that each vehicle starts and ends at the right location. The precedence constraints are formed by constraints (2.5), (2.6), (2.7), and (2.13) while the capacity constraints are formed by constraints (2.8), (2.9), (2.10), and (2.14).

Pickup and delivery problems may further be complicated by different characteristics of transportation requests, various required time constraints, as well as the specified objective function (M. W. Savelsbergh & Sol, 1995). The objective function may minimize duration, completion time, travel time, route length, client convenience and total number of vehicles in completing all requests. On the other hand, profit may also be minimized in a pickup and delivery problem.

2.2 Classifications of the PDP

Different classifications of PDP are explored to identify existing solution approaches and be able to successfully formulate the model in this research. At present, various classifications have been made to identify different types of PDP. In a comprehensive survey conducted by Parragh, Doerner, and Hartl (2008a, 2008b) on PDP models, two main problem classes are identified namely the Vehicle Routing Problem with Backhauls (VRPB) and the Vehicle Routing Problem with Pickups and Deliveries (VRPPD).

This study falls on the VRPPD classification specifically dealing with paired requests and thus more emphasis is put in reviewing related literature in the said field. The classification scheme is shown in Figure 2-1.

Generally the VRPPD deals with problems wherein pickup points and delivery points are specified. The first VRPPD subclass accommodates homogeneous goods and thus each delivery demand may be fulfilled by any picked up good making the pickup and delivery locations unpaired. Some examples of unpaired pickup and delivery

problems are denoted as Pickup and Delivery Vehicle Routing Problem (PDVRP) and Pickup and Delivery Traveling Salesman Problem (PDTSP) which can further be classified depending on a multi-vehicle or single vehicle scenario, respectively.

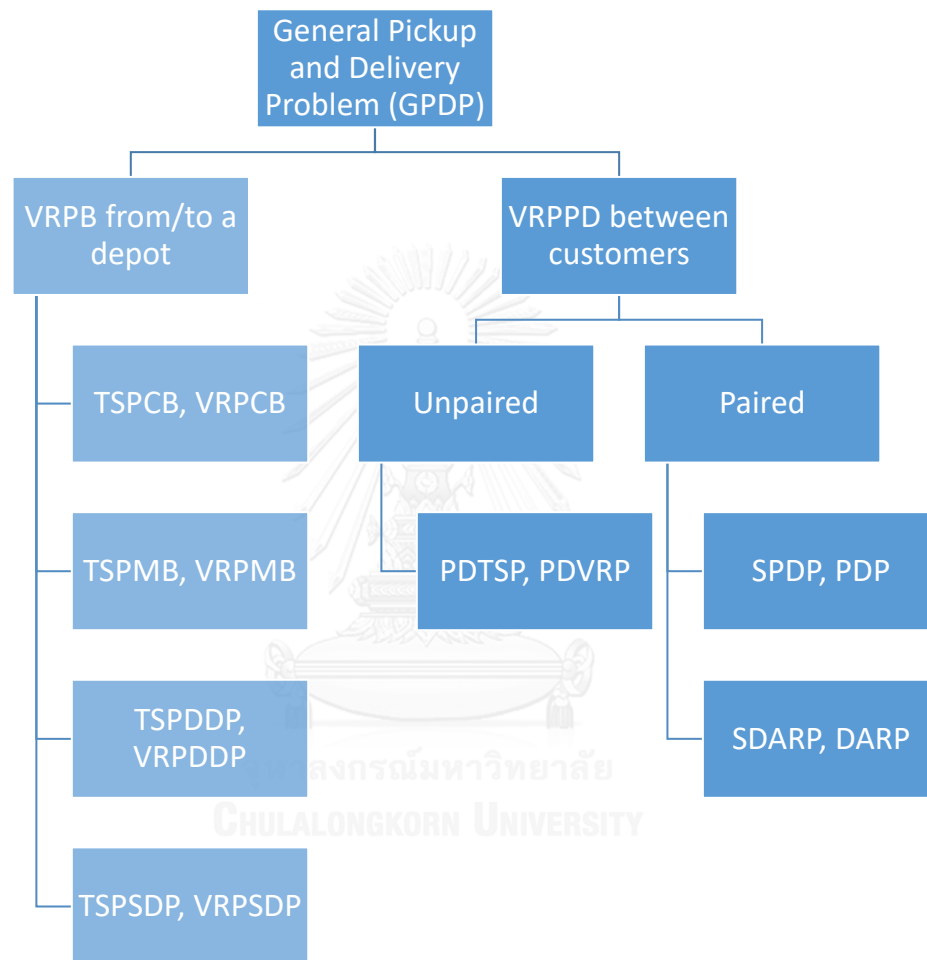


Figure 2-1 Classifications of pickup and delivery problems (Paragh et al., 2008b)

On the other hand, the other classification of the VRPPD tackles the classical Pickup and Delivery Problem (PDP) which deals with transportation of goods as well as the Dial-A-Ride Problem (DARP) which deals with transportation of people. In each transportation request, there is a designated pickup location and delivery location. The objects transported are different resulting in paired pickup and delivery points.

Problems dealing with a single vehicle situation for PDP are denoted as SPDP while problems dealing with a single vehicle situation for DARP are denoted as SDARP. The VRPPD problem specified in this research is classified under the classical PDP with paired requests and multiple vehicles. Furthermore, this classification is widened by a classification scheme suggested by Berbeglia, Cordeau, Gribkovskaia, and Laporte (2007) as shown in Figure 2-2. The proposed classification structure specifically deals with the static case of the classical pickup and delivery problem. It is important to note that PDPs may still be distinguished based on two main cases: the static case and the dynamic case. In static cases of the problem, all information are known beforehand while in dynamic cases, new sets of information are considered over time. The problem considered in this research is of the static case.

In the classification of Berbeglia et al. (2007), static pickup and delivery problems were categorized into three main types namely many-to-many, one-to-many-to-one, and one-to-one. Definitions of the said classes based on the work of Berbeglia et al. (2007) are enumerated as follows:

- Many-to-many problems – For each commodity, there can be several origins and destinations. Two common examples for the said class type include the swapping problem and the one-commodity pickup and delivery traveling salesman problem.
- One-to-many-to-one pickup and delivery problems – In this type of problem, commodities usually come from a depot and are sent to customer vertices while the customer vertices also has items to send back to the depot. The said type of problem is usually applied in reverse logistics where full containers are brought to customers and empty containers have

to be returned to the depot too. Variations of the one-to-many-to-one problem include single and combined demands as well as single and multi-vehicle variations.

- One-to-one problems – Every commodity in this case has only one pickup and one delivery vertex. There is always a one-to-one correspondence between the pickup and delivery vertices. Two of the most important problems under this class are vehicle routing problem with pickups and deliveries (VRPPD) and the Dial-a-Ride Problem (DARP).

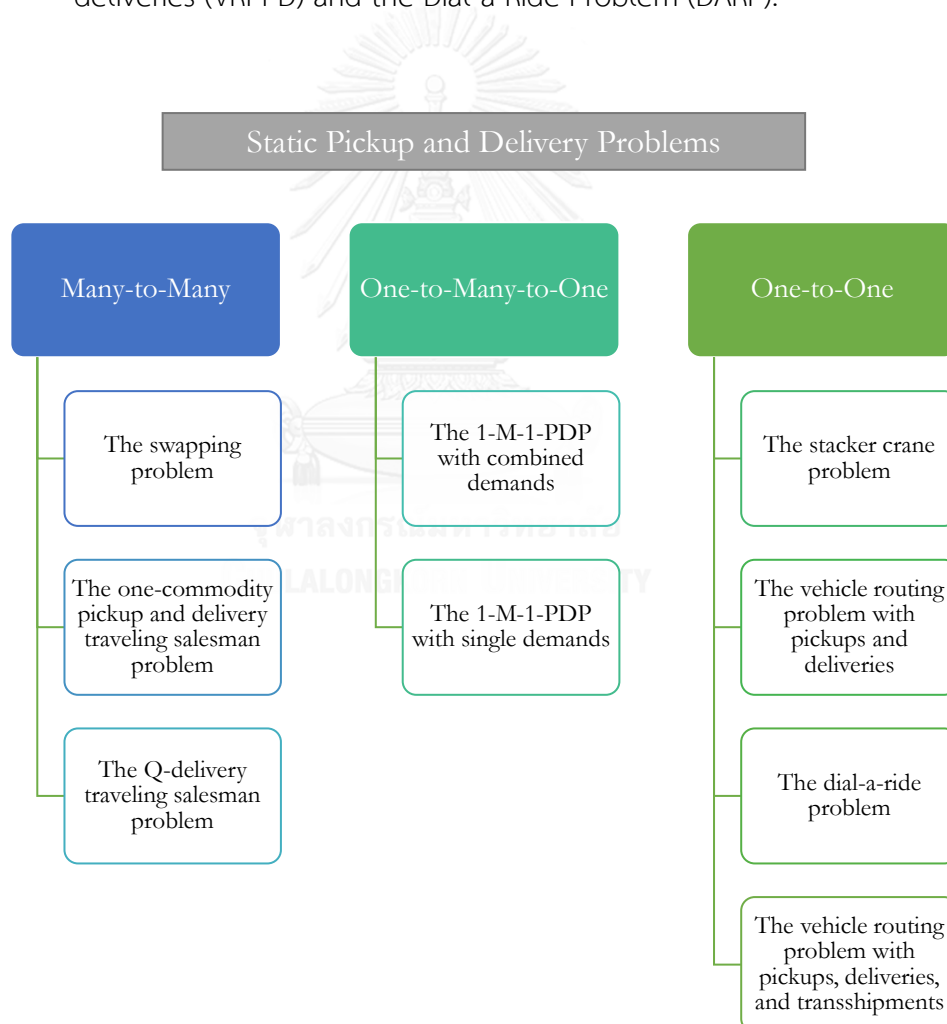


Figure 2-2 Static pickup and delivery problem classification (Berbeglia et al., 2007)

The research mainly falls to the static one-to-one pickup and delivery problem classification especially if the main application considered would be courier services and door-to-door type of services. With a wide variety of vehicle routing classification about pickup and delivery problems, it is important to be able to classify the specific group where this study belongs to, to be able to identify the previous studies that have already been done as well as the solution approaches that has been used to solve the problem.

2.3 One-to-One PDP

One of the most popular types of research falls under the one-to-one PDP classification. For a detailed survey of related literature on one-to-one static and pickup delivery problems, the work of Berbeglia et al. (2007) may be referred to. The authors divided the said type of problem into four categories namely the stacker crane problem, the vehicle routing problem with pickups and deliveries, the dial-a-ride problem, and the vehicle routing problem with pickups, deliveries and transshipments. Variations of the said sub-types of one-to-one PDP are done with the use of single or multi-vehicle, and the methods proposed in each classification. On the other hand, Jean-François Cordeau, Laporte, and Ropke (2008) focused on reviewing recent models and algorithms for the one-to-one PDP case. In single vehicle pickup and delivery problems (SVPDPs), exact algorithms were reviewed such as branch-and-cut algorithms for both SVPDP with and without last-in-first-out (LIFO) constraints. The authors also reviewed heuristics for the SVPDP. On the other hand, for the multi-vehicle pickup and delivery problems (MVPDPs), branch-and-cut algorithm and branch-and-cut-and-price algorithm for the dial-a-ride problem (DARP) and the pickup and

delivery problem with time windows (PDPTW) were reviewed. Heuristics for the MVPDPs include tabu search for the DARP, hybrid heuristic and adaptive large neighborhood search heuristic for the PDPTW, and the double-horizon heuristic for the dynamic PDPTW. Jean-François Cordeau et al. (2008) concluded that solutions based on branch-and-cut and branch-and-cut-and-price algorithms served as the best exact solution methodologies for the reviewed problem sets. Furthermore, the researchers also noted that the success of the proposed heuristics depends on its smart design. Recent studies involving one-to-one PDPs include the research of Hernández-Pérez and Salazar-González (2009) which considered the multi-commodity setup for the one-to-one PDP. The problem was also defined as the capacitated version of the classical traveling salesman problem (TSP) with precedence constraints. The researchers presented two mixed integer linear programming models, describing a decomposition technique for each to find the optimal solution. Meanwhile, Berbeglia, Cordeau, and Laporte (2010) extended the one-to-one PDP to its dynamic case. The authors created a general framework for the dynamic one-to-one pickup and delivery problem.

To summarize, the following solution approaches were tried to solve the one-to-one pickup and delivery problem varying between single vehicle setup and multi-vehicle setup.

Single Vehicle Variation

- Branch-and-Bound Algorithm
- Dynamic Programming
- Branch-and-Cut Algorithm

- Variable Neighborhood Search

Multi-Vehicle Variation

- Column Generation
- Branch-and-Cut Algorithm
- Branch-and-Cut-and-Price Algorithm
- Tabu Search Heuristic

It is observed that even though previous researches varied between single vehicle setup and multi-vehicle setup, most of these studies dealt with a single depot scenario. In this research, a multi-depot scenario is solved. Moreover, a one-to-one static case is discussed in this paper. A detailed review of recent models and algorithms for the one-to-one PDP may be referred to in the work of Jean-François Cordeau et al. (2008). PDPs may further vary depending on the objective on which number of vehicles, vehicle capacity, load size, origin and destination, and time constraints may all be considered.

2.4 The Messenger Problem

With more recent research, the problem discussed in this paper may closely be comparable to the “messenger problem” referred to in the works of Fabry (2007, 2015); Fábry and Kobzareva (2012).

The messenger problem, classified under the one-to-one PDP type, is defined as a special case of the traveling salesman problem (TSP) wherein both the pickup origin and the delivery destination comprise one complete request.

From the static case of the messenger problem, Fabry (2007) extended his research to the dynamic case of the messenger problem. In his study in 2007, he used optimization, insertion algorithm and re-optimization techniques to solve the dynamic messenger problem. Fabry (2007) concludes that re-optimization algorithm can be used for problems with a small size and that huge problems would need heuristics. Moreover, insertion algorithm proved to be a simple and effective method for the dynamic messenger problem. Fabry (2007) also notes that models can be made closer to real messenger problems by also considering time windows, vehicle capacity, and number of vehicles.

Fábry and Kobzareva (2012) solved a multiple messenger problem using modified nearest neighbor algorithm, modified insertion algorithm, and modified exchange algorithm. The study was able to generate multiple routes. Aside from determining efficiency of proposed methods, the research also focused on improving solutions through the use of modified exchange algorithm wherein shipments from other routes are exchanged with other routes to minimize total distance travelled by all messengers.

The most recent work regarding the multiple messenger problem dealt with multiple depots which was solved using similar concepts namely insertion method, nearest neighbor algorithm, and exchange algorithm (Fabry, 2015). Results of the

study's computational experiments showed higher efficiency of proposed insertion method over nearest neighbor algorithm and that significant improvements were made through the use of exchange algorithm.

The most recent research of Fabry (2015) is the closest one to the problem tackled in this study although one main difference that can be distinguished is that in Fabry (2015)'s multi-depot setup, all depots must be included in the solution and only one vehicle can come from one depot. In this research, not all depots are required to be used. The study also allows multiple vehicles to come from a single depot.

2.5 Research Gap

With other studies related to this research, most variations observed are made between single-vehicle problems and multiple-vehicle problems using a singular depot. There are solution approaches that can deal with multiple depots but it is observed that the impact of the number of depots on the PDP network has received far less attention. Research on the single-vehicle PDP focused on methodologies such as exact algorithms related to branch-and-cut algorithm (Jean-François Cordeau, Iori, Laporte, & Salazar González, 2010; Dumitrescu, Ropke, Cordeau, & Laporte, 2008; Hernández-Pérez & Salazar-González, 2003) and heuristic approaches using variable neighborhood search and insertion method techniques (Carrabs, Cordeau, & Laporte, 2007; Hernández-Pérez & Salazar-González, 2004). With the multiple-vehicle variation, static cases still depended on branch-and-cut algorithm (Ropke & Cordeau, 2009; Ropke, Cordeau, & Laporte, 2007) and the developed heuristics used tabu search and large neighborhood search techniques (Jean-François Cordeau & Laporte, 2003; Ropke &

Pisinger, 2006). Overall, it was concluded that solutions based on branch-and-cut and branch-and-cut-and price algorithms serve as the best exact solution methodologies for PDPs (Jean-François Cordeau et al., 2008). Heuristics may vary depending on the objective and complexity of constraints of the formulation.

In summary, formulating the pickup and delivery problem may be a difficult task and the solution approach may further complicate the said problem especially if more depots are considered. Therefore, this paper mainly aims to formulate a routing model to the static case of a one-to-one pickup and delivery problem which includes both generation of routes as well as the depot assignment simultaneously. Typical PDP setup separates generation of routes from depot assignment. Furthermore, the research will also be of great contribution to the very limited amount of pickup and delivery problem resource handling multiple depots. The model can be of much benefit to logistics companies especially those specializing in pickup and delivery services.

CHAPTER III

METHODOLOGY

3.1. Overall Research Framework

The overall design flow of this research is shown in Figure 3-1. In order to accomplish the objectives of the study, review of related literature was first done. Pickup and delivery problems and its classifications were studied. The problem in this research was then classified under the static case, one-to-one multi-vehicle PDP with multiple depots. Based on existing solution approaches, the routing model was formulated and solved by implementing the model in a General Algebraic Modelling System (GAMS) environment. Instances from previous studies were modified for computational experiments. Lastly, results were discussed and analyzed, arriving with final conclusions and recommendations.

3.2. Formulation of Model

The problem consists of n set of requests. Each request consists of one pickup origin and one delivery destination of a certain customer. Let K be the set of available depots and R be the set of routes. Considering $k \in K$ depots in the problem, the total number of locations in the distribution network is $2(k+n)$. The starting depot and ending depot are designated into two separate nodes and are assigned with similar coordinates. A certain route $r \in R$ should start and end in a similar depot. In one particular instance of the problem, the main objective of the model is to combine the given set of requests into $r \in R$ set of routes resulting to the least possible total distance.

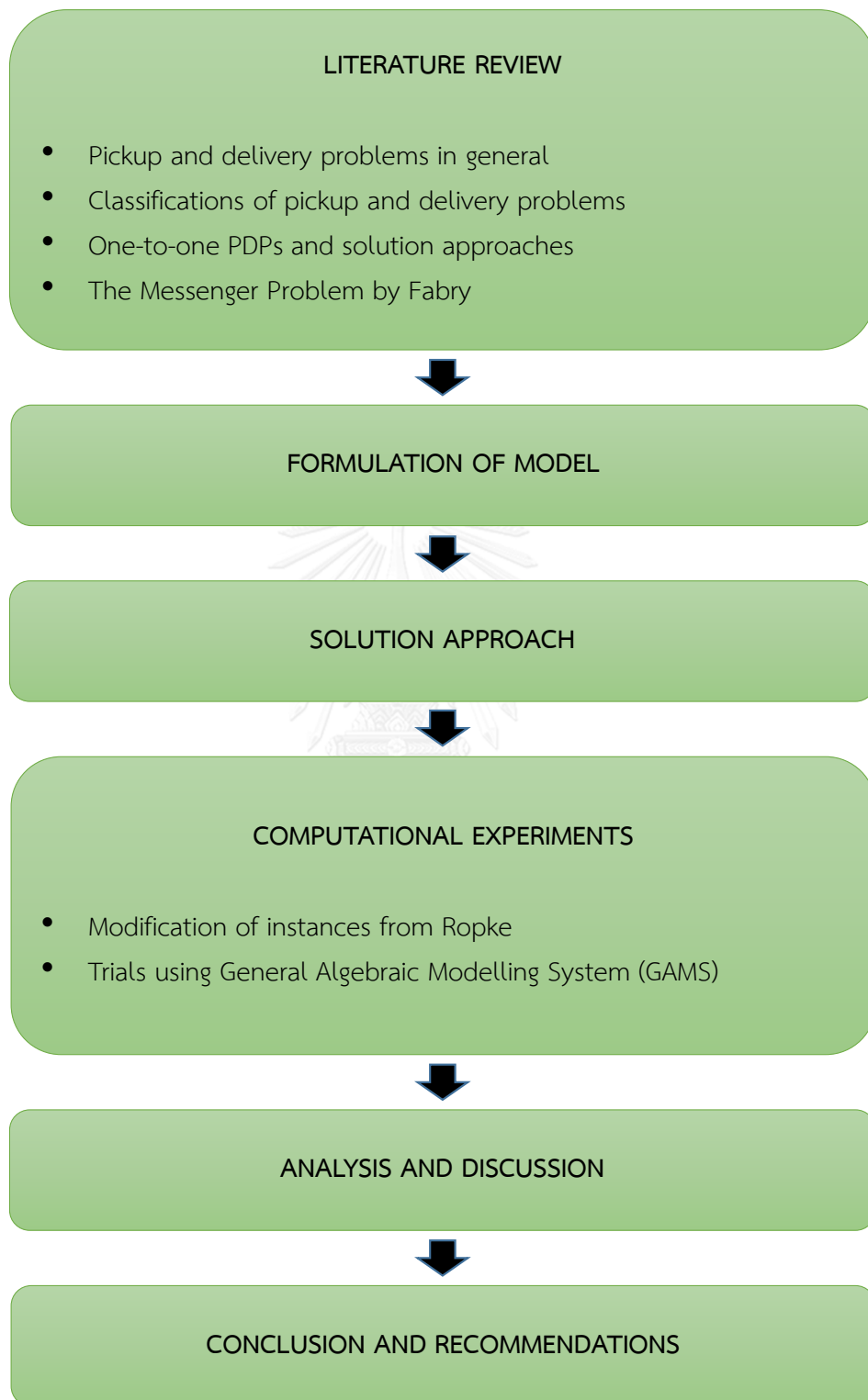


Figure 3-1 Overall design flow of research

The network considers Euclidean distances between each node and denotes c_{ij} as the shortest distance from location i to location j . The problem may be defined on a complete directed graph $G = (V, A)$, where $V = K_s \cup P \cup D \cup K_e$. Subset $K_s = \{1, \dots, k\}$, subset $P = \{1+k, \dots, k+n\}$, subset $D = \{1+k+n, \dots, k+2n\}$, and subset $K_e = \{1+k+2n, \dots, 2(k+n)\}$. The subsets K_s and K_e represent the starting and ending depots, respectively, which are paired according to their similar coordinates. Meanwhile, subset P represents the pickup nodes and subset D represents the delivery nodes. The subsets P and D may be combined to subset C to represent both pickup and delivery nodes.

Based on the messenger problem by Fabry (2007), the mathematical model of the static one-to-one pickup and delivery problem in this paper can be formulated as the following mixed-integer program.

$$\text{Minimize } z = \sum_{i \in K_s \cup C} \sum_{j \in C \cup K_e} \sum_{r \in R} c_{ij} x_{ij}^r \quad (3.1)$$

$$\text{Subject to } \sum_{r \in R} x_{ij}^r = \alpha \quad \forall i \in K_s \cup C, j \in C \cup K_e \quad (3.2)$$

$$\sum_{i \in C} x_{ij}^r \leq 1 \quad \forall j \in C \cup K_e, r \in R \quad (3.3)$$

$$\sum_{j \in C} x_{ij}^r \leq 1 \quad \forall i \in K_s \cup C, r \in R \quad (3.4)$$

$$\sum_{r \in R} x_{ij}^r = 1 \quad \forall i \in K_s, j \in P \quad (3.5)$$

$$\sum_{r \in R} x_{ij}^r = 1 \quad \forall i \in D, j \in K_e \quad (3.6)$$

$$\sum_{r \in R} x_{ij}^r - x_{pq}^r = 0 \quad \forall i \in K_s, j \in P, \quad p \in D, q \in K_e \quad (3.7)$$

$$\sum_{i \in C} \sum_{r \in R} x_{ij}^r - \sum_{q \in C} \sum_{r \in R} x_{pq}^r = 0 \quad \forall j \in C \cup K_e, p \in K_s \cup C \quad (3.8)$$

$$\sum_{r \in R} x_{ij}^r \sqrt{\beta_P} - x_{pj}^r \sqrt{\beta_D} = 0 \quad \forall i \in P, p \in D, j \in C \cup K_e \quad (3.9)$$

$$\sum_{i \in C} \sum_{j \in C \cup K_e} u_i - u_j + (2n + 1)x_{ij}^r \leq 2n \quad \forall r \in R \quad (3.10)$$

$$u_i - u_j \leq 0 \quad \forall i \in C, j \in D \quad (3.11)$$

Model parameters are defined and summarized as follows:

- n total number of given requests; one request consists of one pickup point and one delivery point
- α number of links in one route which is equivalent to $2n+1$
- q number of allowed requests in one route r
- R number of routes needed to serve all demands equivalent to n/q
- K total number of depots in the network; a depot is considered as the station of vehicles
- K_s reference for starting depots
- K_e reference for ending depots
- V total number of nodes in the network
- P subset of V; reference for pickup nodes
- D subset of V; reference for delivery nodes
- C subset of V; total number of pickup and delivery nodes equivalent to $P+D$
- β_P a special integer number assigned to each pickup node to pair with corresponding delivery node
- β_D a special integer number assigned to each delivery node to pair with corresponding pickup node

i, p	index of origin nodes
j, q	index of destination nodes
r	index of route used; $r = 1, 2, \dots, R$
k	index of depot/vehicle used; $k = 1, 2, \dots, K$

The decision variables are defined and enumerated as follows:

z	the total cost of all the chosen routes to be implemented
x_{ij}^r	if link i to j is used for route r , 0 otherwise
u	special integer variable used for sequencing purposes

The objective function (3.1) minimizes the total routing costs to serve all given requests. In constraint (3.2), the model forces a certain number of links into one route. The value of α may be computed depending on the given requests n and how it is divided to the desired number of requests q that one route is chosen to serve ($\alpha=2q+1$). It is important to distinguish that n denotes the total number of requests in the network while q represents the desired number of requests to be served in one complete route. Constraints (3.3) and (3.4) ensure that in each node, there is only one incoming link and one outgoing link. In constraints (3.5) and (3.6), a starting depot and an ending depot are chosen for a particular route. Constraint (3.7) makes sure that the starting and ending depots are corresponding with each other. For the pickup nodes and their respective delivery nodes, constraints (3.8) and (3.9) are implemented to ensure that the pickup-delivery pair of nodes are included in one route. The parameters β_p and β_d are special integer numbers that are assigned to pickup nodes and delivery nodes, respectively for pairing purposes. Based on the work of Fabry (2007) using Miller-Tucker-Zemlin's inequalities, constraint (3.10) is included to avoid

partial cycles in the solution. Lastly, constraint (3.11) ensures that the pickup node comes before the corresponding delivery node (Fabry, 2007).

3.3. Modified Instances

All the instances used for the computational experiments were based on the instances provided by Ropke et al. (2007) in their previous research regarding models and branch-and-cut algorithms for pickup and delivery problems. The instances were generated as suggested by M. Savelsbergh and Sol (1998). Coordinates of every pickup and delivery location were randomly chosen over a $[0, 50] \times [0, 50]$ square. The instances used the same format with the DARP instances generated in the work of Jean-François Cordeau (2006). One sample instance (AA30) is shown in Table 3.1. The sample instance contains 30 requests and has a given capacity of 15 for vehicles. The first column shows the number of nodes including the starting and ending node representing the depot. The second and third column show the coordinates of each pickup and delivery location. The fourth column shows the service time and the fifth column shows the demand. Finally, the last two columns show the time windows for each of the nodes in the instance.

Computational experiments started in handling six requests first and thus the instances were modified by using the first pickup points and their corresponding delivery locations. Since the model focuses on combining requests and each request only includes one item, capacity was not included in the research. Time windows were also not considered in solving the modified instances.

Table 3.1 Sample instance from Ropke et al. (2007)

Node No.	X	Y	Time	Demand	Start	End
0	25	25	0	0	0	1000
1	19.942	16.269	0	7	175	235
2	13.131	45.499	0	9	522	582
3	41.105	42.288	0	15	256	316
4	29.622	23.653	0	5	529	589
5	20.387	33.75	0	12	469	529
6	8.195	24.765	0	5	436	496
7	39.947	22.762	0	10	517	577
8	32.184	39.039	0	8	414	474
9	46.07	49.15	0	11	28	88
10	15.395	23.92	0	12	96	156
11	38.336	13.008	0	8	169	229
12	49.706	28.635	0	14	54	114
13	17.748	22.292	0	15	423	483
14	42.549	29.924	0	8	112	172
15	44.178	9.661	0	11	511	571
16	26.591	36.243	0	11	36	96
17	3.855	9.895	0	5	24	84
18	37.401	26.569	0	11	293	353
19	45.989	20.442	0	7	402	462
20	35.511	15.228	0	10	496	556
21	39.026	12.034	0	13	477	537
22	28.364	23.084	0	10	299	359
23	40.399	46.524	0	10	267	327
24	42.439	43.835	0	14	61	121
25	48.019	11.272	0	10	410	470
26	7.711	35.675	0	13	300	360
27	12.451	19.724	0	6	538	598
28	11.466	10.009	0	8	378	438
29	31.894	3.068	0	9	549	609
30	31.193	29.453	0	11	64	124
31	35.573	46.733	0	-7	209	269
32	36.158	11.897	0	-9	562	622
33	43.76	46.624	0	-15	261	321
34	13.387	36.048	0	-5	549	609
35	21.442	34.131	0	-12	470	530
36	4.515	24.464	0	-5	439	499
37	14.379	35.446	0	-10	545	605
38	17.381	24.472	0	-8	434	494
39	43.627	19.724	0	-11	57	117
40	0.062	49.145	0	-12	125	185
41	42.389	13.101	0	-8	173	233
42	6.799	22.468	0	-14	97	157
43	24.429	6.787	0	-15	439	499
44	44.972	41.699	0	-8	124	184
45	30.09	18.098	0	-11	527	587
46	17.571	39.599	0	-11	45	105
47	30.378	32.49	0	-5	58	118
48	20.594	9.692	0	-11	316	376
49	19.93	25.913	0	-7	428	488
50	5.808	45.172	0	-10	538	598
51	0.497	25.269	0	-13	517	577
52	43.952	38.259	0	-10	320	380
53	13.843	16.968	0	-10	306	366
54	19.524	12.881	0	-14	99	159
55	30.948	13.248	0	-10	427	487
56	15.145	19.745	0	-13	317	377
57	18.778	35.536	0	-6	555	615
58	47.436	7.99	0	-8	414	474
59	6.843	25.729	0	-9	582	642
60	17.743	42.465	0	-11	82	142
61	25	25	0	0	0	1000

Table 3.2 shows a sample modified instance containing $n = 6$ requests with each location's coordinates. Containing three depots, the first three nodes and last three nodes are included to represent the starting depots and ending depots, respectively. The requests are color-coded with rows in blue representing pickup points and rows in green representing delivery points. If a request is picked up from location i , it will be delivered to location $(i + n)$. In the fourth column, a special integer number is assigned per node to denote the pairing correspondence of each request. Figure 3-2 shows the plotted coordinates of the sample instance with six requests.

Table 3.2 Sample modified instance with six requests

Node	X	Y	Pair	No.
node1	20	30	0	1/16
node2	25	20	0	2/17
node3	30	30	0	3/18
node4	19.942	16.269	1	4
node5	8.195	24.765	2	5
node6	38.336	13.008	3	6
node7	26.591	36.243	4	7
node8	39.026	12.034	5	8
node9	7.711	35.675	6	9
node10	35.573	46.733	1	10
node11	4.515	24.464	2	11
node12	42.389	13.101	3	12
node13	17.571	39.599	4	13
node14	0.497	25.269	5	14
node15	15.145	19.745	6	15
node16	20	30	0	16
node17	25	20	0	17
node18	30	30	0	18

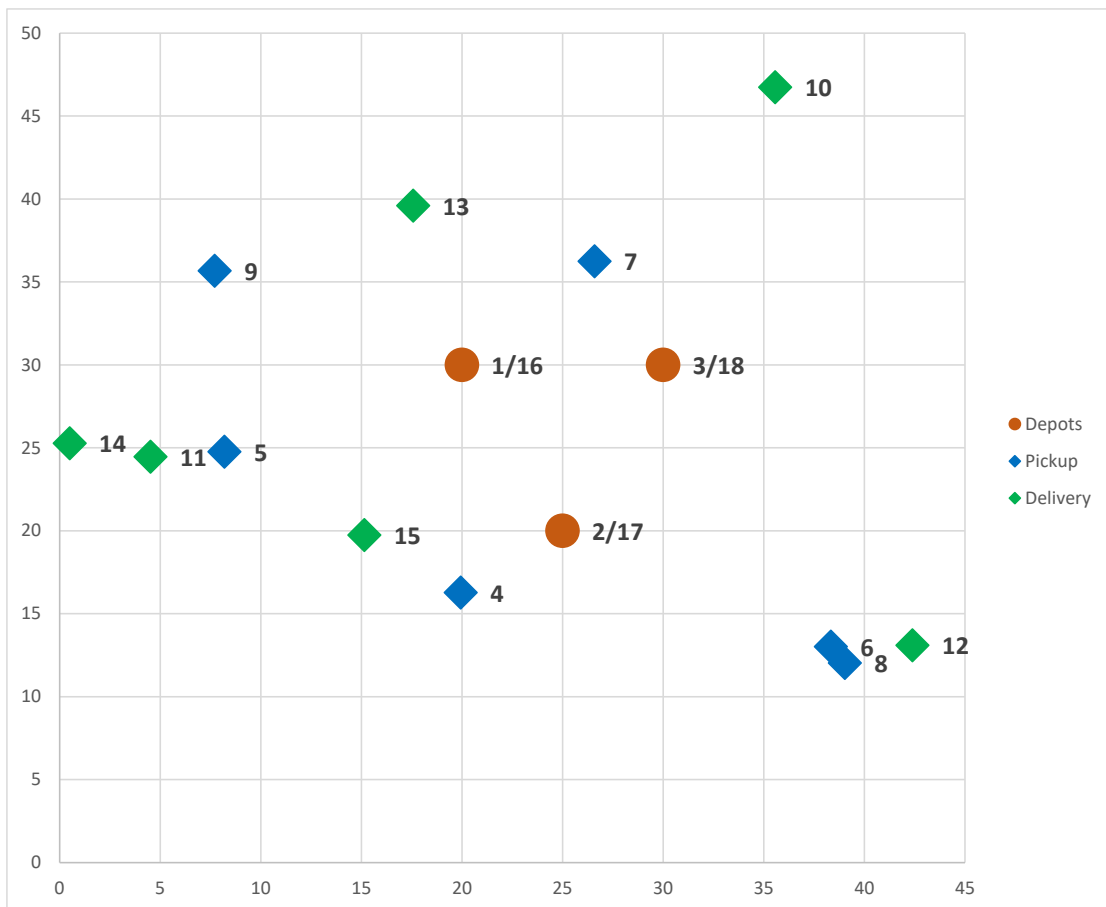


Figure 3-2 Plot of sample modified instance with six requests

On the other hand, a sample instance containing twelve requests can be seen in Table 3.3 while the plotted coordinates of the said instance can be seen in Figure 3-3. The sample instances shown, containing six requests and twelve requests, respectively, contain three depots having the same location: centered at the perimeter of the area. This location of depots is also modified as seen from Figure 3-4. Instances were varied between those having clustered depots and those of which having scattered depots.

Table 3.3 Sample modified instance with twelve requests

Node	X	Y	Pair	No.
node1	20	30	0	1/28
node2	25	20	0	2/29
node3	30	30	0	3/30
node4	19.942	16.269	1	4
node5	13.131	45.499	2	5
node6	29.622	23.653	3	6
node7	20.387	33.75	4	7
node8	39.947	22.762	5	8
node9	15.395	23.92	6	9
node10	17.748	22.292	7	10
node11	26.591	36.243	8	11
node12	45.989	20.442	9	12
node13	28.364	23.084	10	13
node14	48.019	11.272	11	14
node15	11.466	10.009	12	15
node16	35.573	46.733	1	16
node17	36.158	11.897	2	17
node18	13.387	36.048	3	18
node19	21.442	34.131	4	19
node20	14.379	35.446	5	20
node21	0.062	49.145	6	21
node22	24.429	6.787	7	22
node23	17.571	39.599	8	23
node24	19.93	25.913	9	24
node25	43.952	38.259	10	25
node26	30.948	13.248	11	26
node27	47.436	7.99	12	27
node28	20	30	0	28
node29	25	20	0	29
node30	30	30	0	30

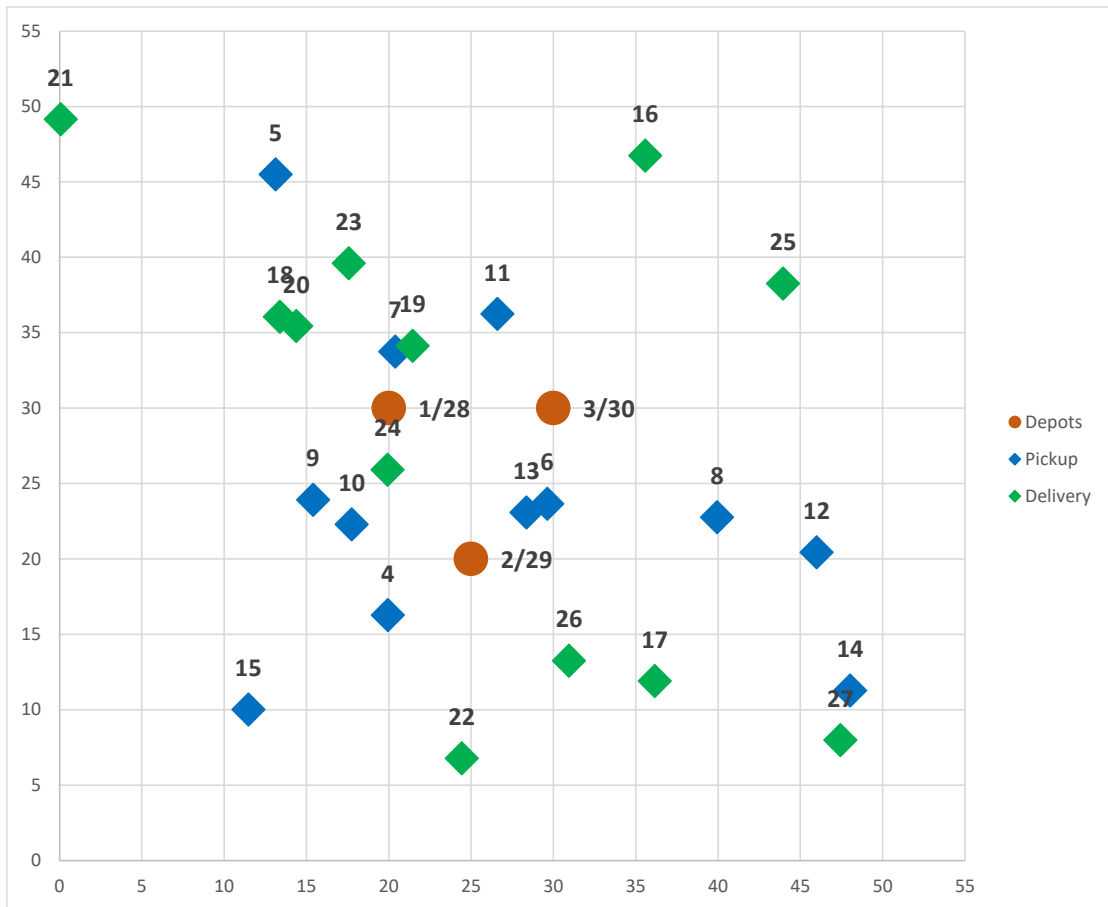


Figure 3-3 Plot of sample modified instance with twelve requests

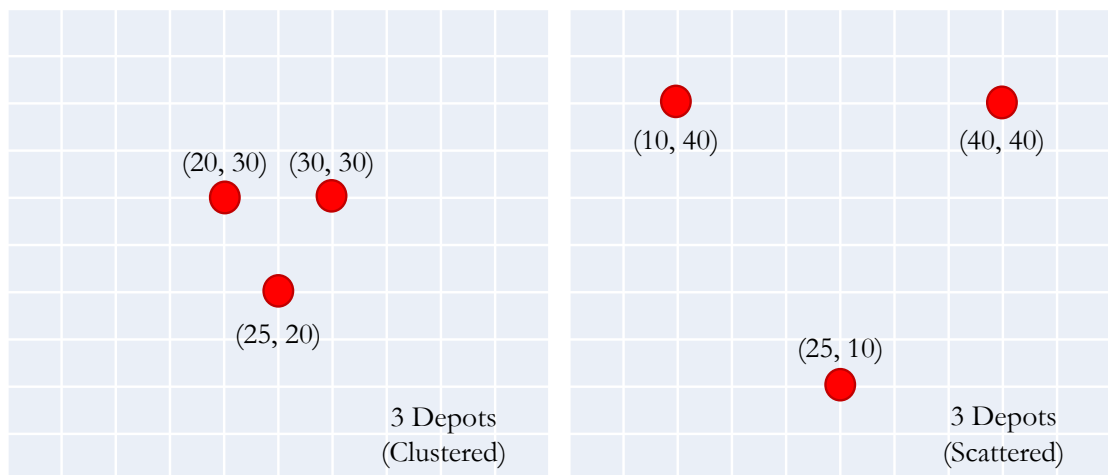


Figure 3-4 Variation of instances containing three depots

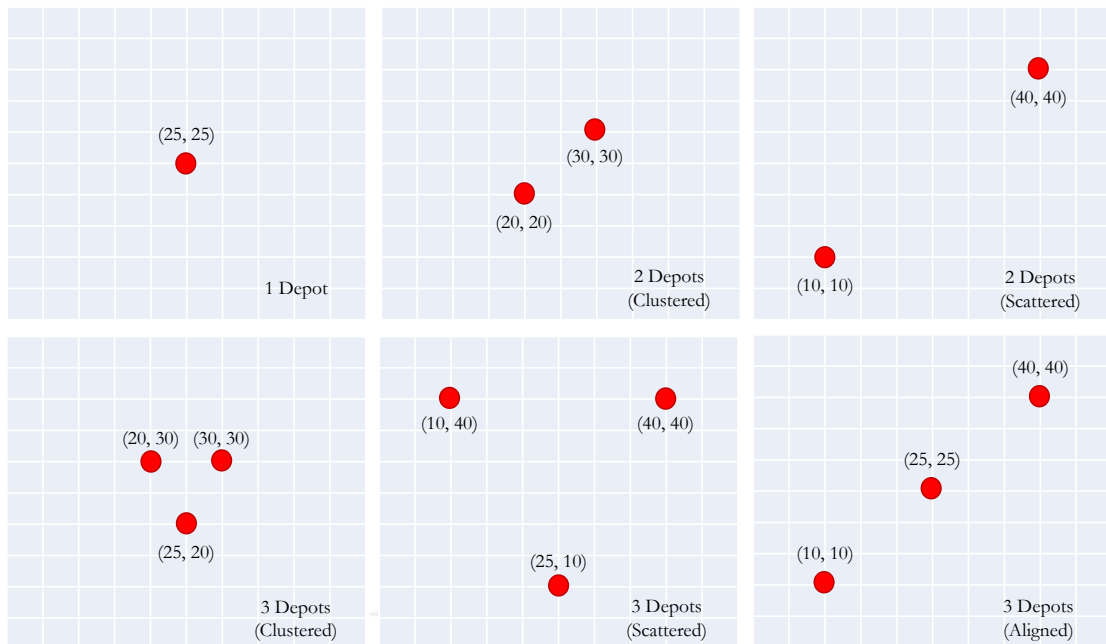


Figure 3-5 Summary of depot variation with the instances

Moreover, the number of depots in each instance was also varied from one to three depots. This depot variation is as shown in Figure 3-5. A special instance having three aligned depots is also added to represent the usual setup of vehicle terminals in corners of alleys along main roads.

3.4. General Algebraic Modelling System (GAMS)

To implement the formulated model in this research, General Algebraic Modelling System (GAMS) platform was used. Chattopadhyay (1999) since regarded GAMS as a high level mathematical model language for developing concise algebraic statements. The modeling language's two most desirable features include (1) the capability to separate the mathematical problem from the solution method making it possible to implement different algorithms for the same problem without changing

the model and (2) the capability to separate data and logic which makes changes in a model easier especially in terms of increasing or decreasing the size of the problem (Chattopadhyay, 1999). In Figure 3-6, the different components of the general GAMS structure may be seen. With the said structure of the modelling language, main advantages include convenience in developing models, separation of data, model and algorithm as well as the provision of choices of a number of powerful commercial solvers. GAMS is further noted to have the flexibility of implementation in a wide variety of optimization problems (Chattopadhyay, 1999).

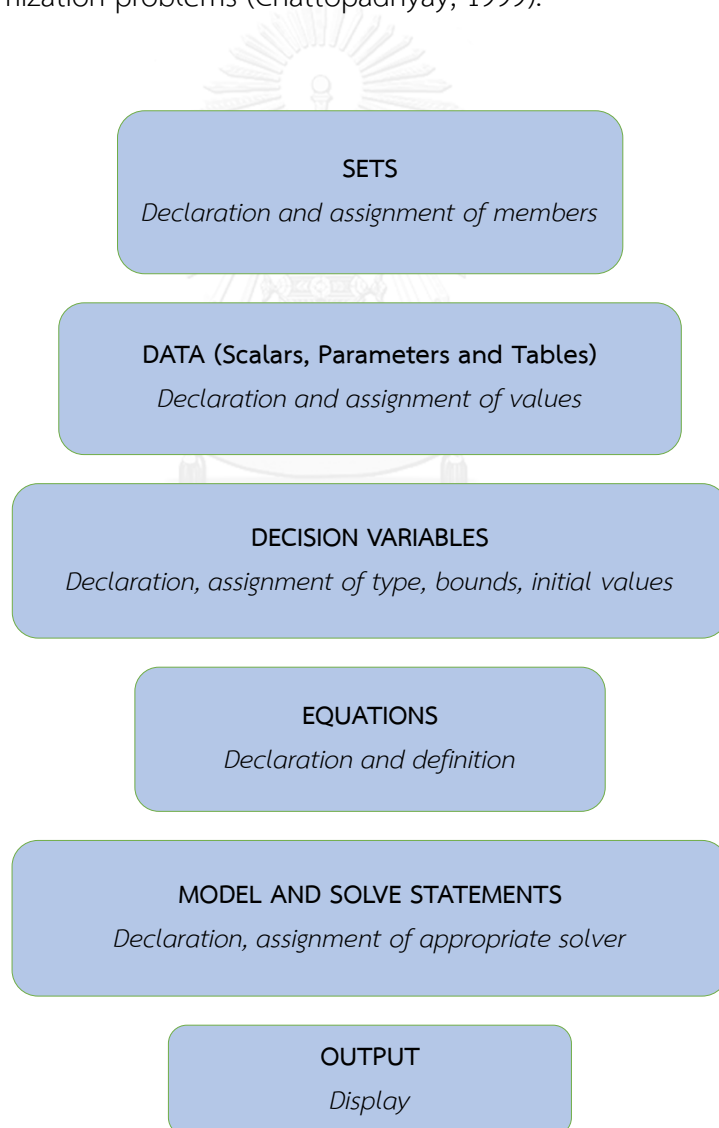


Figure 3-6 Structure of GAMS model (Chattopadhyay, 1999)

With GAMS' capabilities, the study was able to test modified instances using CPLEX as the solver and branch-and-bound as the algorithm. Moreover, under the branch-and-bound algorithm, instances were also varied between best-bound search method and depth-first search method.

3.5. Assumptions for Computational Experiments

Before proceeding to the discussion of results, it is important to remember the following assumptions that were made for the computational experiments conducted in this research.

- The main objective of the model is to minimize the overall distance in serving a given set of pickup and delivery requests. This minimized distance is the Euclidean distance between nodes in the considered network.
- Since the setup of the problem is classified under “one-to-one” pickup and delivery problem, each node is only limited to either one pickup job or one delivery job. The formulated does not support multiple commodities coming from one stop.
- In generating the routes, aside from combining requests in separate routes, the model can also be used to combine all requests in one singular route.
- The instances used only consider distances as the main parameter. Other parameters such as time windows and vehicle capacity are not considered.
- All computational experimental results are based from modified instances from previous studies having hypothetical networks and hypothetical demands.

CHAPTER IV

RESULTS AND DISCUSSION

The formulated model for the static one-to-one PDP with multiple depots was implemented in a General Algebraic Modelling System (GAMS) platform with CPLEX as the solver. The default solution approach of branch-and-bound algorithm was used. Moreover, branch-and-bound algorithm was varied between best-bound search and depth-first search for each of the instances tested. The model was run on a 2.6 GHz Intel(R) Core(TM) i7-4720HQ CPU with 16.0 GB RAM. Instances from Ropke et al. (2007) were modified and problem sets comprising of 6 requests up to 12 requests were tested.

This chapter is separated into six parts. First, a summary of the tested instances is provided. Second, the overall performance of the model implemented in GAMS is evaluated. The third part talks about the reduction of total distance travelled due to the combination of more than one request in one complete route. The fourth part discusses the effect of the depot variation that was implemented. Fifth, advantages and disadvantages of the proposed model are enumerated. Lastly, suggestions are made for using the proposed model in handling instances with a large amount of requests.

4.1 Tested Instances

In an instance with n requests, a number of desired q requests is specified beforehand to be combined in one r route. This in turn determines the total number

of R routes that are needed to serve the given demands ($R = n/q$). Through the desired number of q requests, the number of total links needed to complete one route is also determined. The links needed to complete one r route with q desired requests is equivalent to $\alpha = 2q+1$. These information are the primary input of each run of the tested instances aside from the node numbers, x-coordinates, y-coordinates and pairing designations.

Table 4.1 Characteristics of instances with six requests

Instance (A & B)	Number of Requests (n)	Number of Depots (K)	Depots	Allowed Requests per Route (q)	Total No. of Nodes in Network (V)	Resulting Routes (R)
1-P/R	6	1	Centered	1	14	6
1-6/2				2		3
1-6/3				3		2
1-6/6				6		1
2(1)-P/R		2	Clustered	1	16	6
2(1)-6/2				2		3
2(1)-6/3				3		2
2(1)-6/6				6		1
2(2)-P/R			Scattered	1		6
2(2)-6/2				2		3
2(2)-6/3				3		2
2(2)-6/6				6		1
3(1)-P/R		3	Clustered	1	18	6
3(1)-6/2				2		3
3(1)-6/3				3		2
3(1)-6/6				6		1
3(2)-P/R			Scattered	1		6
3(2)-6/2				2		3
3(2)-6/3				3		2
3(2)-6/6				6		1
3(3)-P/R			Aligned	1		6
3(3)-6/2				2		3
3(3)-6/3				3		2
3(3)-6/6				6		1

Table 4.2 Characteristics of instances with twelve requests

Instance (A, B, & C)	Number of Requests (n)	Number of Depots (K)	Location of Depots	Allowed Requests per Route (q)	Total No. of Nodes in Network (V)	Resulting Routes (R)
1-P/R	12	1	Centered	1	26	12
1-12/2				2		6
1-12/3				3		4
1-12/12				12		1
2(1)-P/R		2	Clustered	1	28	12
2(1)-12/2				2		6
2(1)-12/3				3		4
2(1)-12/12				12		1
2(2)-P/R			Scattered	1		12
2(2)-12/2				2		6
2(2)-12/3				3		4
2(2)-12/12				12		1
3(1)-P/R		3	Clustered	1	30	12
3(1)-12/2				2		6
3(1)-12/3				3		4
3(1)-12/12				12		1
3(2)-P/R			Scattered	1		12
3(2)-12/2				2		6
3(2)-12/3				3		4
3(2)-12/12				12		1
3(3)-P/R			Clustered	1		12
3(3)-12/2				2		6
3(3)-12/3				3		4
3(3)-12/12				12		1

Table 4.1 and Table 4.2 provide the characteristics of the tested instances. The first column provides the name of each case. The first number denotes the number of depots in the instance. The number of depots was varied from one to three depots. For instances with multiple depots, tests were performed between two sets: depots close to each other and depots which are far from each other. A number enclosed in

parentheses in multi-depot instances denotes the clustered case (1) and the scattered case (2) of depots. Additionally, aside from clustered and scattered cases of depots, a third setup, (3) aligned case, is added. This is done to also represent the typical setup of vehicle terminals in corners of alleys along main roads.

For the instances tested, two to three requests were combined in one route. The resulting routes from this combination is shown in the last column of Table 4.1 and Table 4.2. All instances used for testing are symmetrical ($c_{ij} = c_{ji}$ for every pair of locations i and j). The total number of nodes included in one instance of the network problem is also shown. Overall, there were three sets of instances tested (A, B & C). Each set contains two subsets, one with six requests and one with twelve requests. For each instance, both best-bound search method and depth-first search method are performed. In total, there were 288 instances tested in the research.

4.2 Model Performance

In this section, the performance of the proposed model in GAMS using branch-and-bound algorithm is evaluated.

4.2.1 Instances with six requests

The computational results for the three sets of instances with six requests are shown in Table 4.3 (A), Table 4.4 (B), and Table 4.5 (C). The results from two separate methods of branch-and-bound algorithm namely best-bound search and depth-first search are shown in separate parts. Generally,

the model was able to find optimal solutions with all the instances containing six requests. An asterisk preceding an objective function value indicates that the instance was solved to optimality.

In each of the sets A, B, and C, respectively, both the best-bound search method and depth-first search method showed almost similar results with the total distance objective value as shown in the 2nd and 6th columns of each table. The similarity of results for both methods may be attributed to the small size of the instances tested.

In set A6 of Table 4.3, 23 out of 24 instances showed similar objective value results from both best-bound search method and depth-first search method. One instance (A3(1)-6/3) showed a very minor difference of 0.07 between the two branch-and-bound algorithm methods with both methods having a 0.99% gap from the lower bound value and the best-bound result having a lower objective value. For this set, the gaps of the objective values from the lower bound of all set A6 instances are about 1% or less. The recorded CPU run time for almost all A6 instances is less than 60 seconds. Only one A6 instance (A3(1)-6/3), same instance previously mentioned from depth-first search recorded a CPU run time of about 71 seconds.

In set B6 of Table 4.4, there are also 23 out of 24 instances which showed similar objective value results from both best-bound search method and depth-first search method. One instance (B1-6/6) showed a minor difference of 0.70 but this time, the depth-first search method showed the

lower objective. However, it is important to note that the best-bound search in this particular instance was able to find a lower gap of 0.43% from the lower bound value than that of the 0.64% of the depth-first search. In set B6 of Table 4.4, the gaps of the objective values from the lower bound of all instances are also about 1% or less. The recorded CPU run time for all B6 instances are about 47 seconds or less except one single instance (*B3(3)-6/3*) which recorded a CPU run time of about 90 seconds. The highest CPU run time was found from a depth-first search run (*B3(3)-6/3*).

In set C6 of Table 4.5, there are 21 out of 24 instances which showed similar objective value results for both best-bound search method and depth-first search method. Three instances (*C3(1)-6/2*, *C3(1)-6/6* and *C3(3)-6/3*) showed minor differences between the objective values found but for these instances, a lower gap from the lower bound value was usually observed from best-bound search method. Overall, the gaps recorded are also about 1% or less. Larger CPU run times are observed in this set of instances. The highest CPU run time recorded was about 201 seconds from best-bound search method (*C3(1)-6/3*). Generally, the higher CPU run times are found from instances that are constructing a higher number of routes.

Overall, the performance of the model for the instances with six requests is excellent. Minor differences were found between the two branch-and-bound algorithm methods: best-bound search and depth-first search. The model was also able to find a low gap of 1% or less from the lower bound value for all instances tested with six requests.

Table 4.3 Summary of results for instances with six requests (A6)

Instance	Best-Bound Search				Depth-First Search			
	Objective Function	Gap	CPU Time (s)	Distance Reduction	Objective Function	Gap	CPU Time (s)	Distance Reduction
A1-P/R	*323.32	0.00%	0.02		*323.32	0.00%	0.00	
A1-6/2	*229.19	0.16%	15.61	-29.11%	*229.19	0.99%	15.39	-29.11%
A1-6/3	*194.76	1.00%	32.73	-39.76%	*194.76	0.99%	21.05	-39.76%
A1-6/6	*147.39	0.72%	1.72	-54.41%	*147.39	0.72%	1.47	-54.41%
A2(1)-P/R	*301.74	0.00%	0.02		*301.74	0.00%	0.03	
A2(1)-6/2	*216.37	0.99%	31.81	-28.29%	*216.37	0.86%	26.02	-28.29%
A2(1)-6/3	*179.41	0.97%	27.84	-40.54%	*179.41	0.99%	31.80	-40.54%
A2(1)-6/6	*134.72	0.00%	0.20	-55.35%	*134.72	0.00%	0.13	-55.35%
A2(2)-P/R	*353.69	0.00%	0.03		*353.69	0.00%	0.03	
A2(2)-6/2	*239.47	0.99%	25.17	-32.29%	*239.47	0.96%	25.64	-32.29%
A2(2)-6/3	*195.08	0.99%	26.69	-44.84%	*195.08	1.00%	43.56	-44.84%
A2(2)-6/6	*148.90	0.75%	1.17	-57.90%	*148.90	0.76%	2.36	-57.90%
A3(1)-P/R	*292.77	0.00%	0.50		*292.77	0.00%	0.16	
A3(1)-6/2	*217.43	0.97%	14.88	-25.73%	*217.43	0.54%	21.89	-25.73%
A3(1)-6/3	*187.25	0.99%	47.41	-36.02%	*187.32	0.99%	70.61	-36.04%
A3(1)-6/6	*142.27	0.86%	1.78	-51.41%	*142.27	0.95%	1.64	-51.41%
A3(2)-P/R	*305.64	0.00%	0.28		*305.64	0.00%	0.38	
A3(2)-6/2	*227.93	1.00%	22.38	-25.43%	*227.93	1.00%	37.02	-25.43%
A3(2)-6/3	*181.39	1.00%	31.59	-40.65%	*181.39	0.99%	30.88	-40.65%
A3(2)-6/6	*148.04	0.86%	3.06	-51.56%	*148.04	0.99%	3.11	-51.56%
A3(3)-P/R	*316.37	0.34%	0.36		*316.37	0.34%	0.34	
A3(3)-6/2	*229.19	1.00%	26.80	-27.56%	*229.19	0.99%	25.13	-27.56%
A3(3)-6/3	*191.04	1.00%	46.30	-39.62%	*191.04	0.97%	27.88	-39.62%
A3(3)-6/6	*147.39	0.99%	1.17	-53.41%	*147.39	0.82%	1.11	-53.41%

Table 4.4 Summary of results for instances with six requests (B6)

Instance	Best-Bound Search				Depth-First Search			
	Objective Function	Gap	CPU Time (s)	Distance Reduction	Objective Function	Gap	CPU Time (s)	Distance Reduction
B1-P/R	*342.93	0.00%	0.02		*342.93	0.00%	0.06	
B1-6/2	*239.50	0.95%	6.66	-30.16%	*239.50	0.88%	6.39	-30.16%
B1-6/3	*213.25	0.98%	11.86	-37.82%	*213.25	1.00%	21.20	-37.82%
B1-6/6	*160.58	0.43%	0.28	-53.17%	*159.88	0.64%	0.28	-53.38%
B2(1)-P/R	*312.58	0.00%	0.02		*312.58	0.00%	0.02	
B2(1)-6/2	*235.01	0.99%	26.47	-24.82%	*235.01	0.92%	16.89	-24.82%
B2(1)-6/3	*210.36	0.99%	35.19	-32.70%	*210.36	1.00%	34.52	-32.70%
B2(1)-6/6	*155.84	0.00%	0.44	-50.14%	*155.84	0.99%	0.59	-50.14%
B2(2)-P/R	*321.77	0.00%	0.02		*321.77	0.00%	0.02	
B2(2)-6/2	*243.21	0.98%	28.14	-24.41%	*243.21	0.98%	39.25	-24.41%
B2(2)-6/3	*212.57	1.00%	42.56	-33.94%	*212.57	1.00%	46.78	-33.94%
B2(2)-6/6	*168.22	0.93%	5.20	-47.72%	*168.22	0.97%	4.72	-47.72%
B3(1)-P/R	*324.31	0.99%	0.22		*324.31	0.99%	0.19	
B3(1)-6/2	*234.65	1.00%	27.52	-27.65%	*234.65	1.00%	19.05	-27.65%
B3(1)-6/3	*210.36	1.00%	32.67	-35.14%	*210.36	0.99%	29.69	-35.14%
B3(1)-6/6	*155.84	0.87%	0.48	-51.95%	*155.84	0.41%	0.48	-51.95%
B3(2)-P/R	*354.18	0.00%	0.19		*354.18	0.00%	0.19	
B3(2)-6/2	*244.80	0.99%	19.36	-30.88%	*244.80	0.98%	33.06	-30.88%
B3(2)-6/3	*211.38	0.23%	32.97	-40.32%	*211.38	0.99%	47.25	-40.32%
B3(2)-6/6	*168.22	0.89%	2.94	-52.50%	*168.22	0.99%	3.52	-52.50%
B3(3)-P/R	*284.57	0.00%	0.31		*284.57	0.00%	0.28	
B3(3)-6/2	*232.82	0.99%	35.88	-18.19%	*232.82	0.99%	44.06	-18.19%
B3(3)-6/3	*212.34	1.00%	92.64	-25.38%	*212.34	1.00%	94.66	-25.38%
B3(3)-6/6	*159.88	0.76%	1.75	-43.82%	*159.88	0.99%	2.84	-43.82%

Table 4.5 Summary of results for instances with six requests (C6)

Instance	Best-Bound Search				Depth-First Search			
	Objective Function	Gap	CPU Time (s)	Distance Reduction	Objective Function	Gap	CPU Time (s)	Distance Reduction
C1-P/R	*408.99	0.00%	0.06		*408.99	0.00%	0.00	
C1-6/2	*263.59	1.00%	28.89	-35.55%	*263.59	0.96%	22.42	-35.55%
C1-6/3	*216.10	1.00%	62.75	-47.16%	*216.10	0.95%	75.38	-47.16%
C1-6/6	*160.10	0.99%	8.61	-60.85%	*160.10	1.00%	6.00	-60.85%
C2(1)-P/R	*300.78	0.00%	0.02		*300.78	0.00%	0.08	
C2(1)-6/2	*223.51	0.99%	43.77	-25.69%	*223.51	0.99%	47.09	-25.69%
C2(1)-6/3	*194.16	1.00%	155.00	-35.45%	*194.16	0.99%	145.00	-35.45%
C2(1)-6/6	*155.97	1.00%	23.30	-48.14%	*155.97	0.99%	29.22	-48.14%
C2(2)-P/R	*374.65	0.00%	0.08		*374.65	0.00%	0.02	
C2(2)-6/2	*249.83	0.96%	24.56	-33.32%	*249.83	0.98%	60.66	-33.32%
C2(2)-6/3	*207.28	1.00%	165.28	-44.67%	*207.28	0.99%	105.94	-44.67%
C2(2)-6/6	*151.04	0.96%	5.16	-59.69%	*151.04	0.97%	6.45	-59.69%
C3(1)-P/R	*279.50	0.00%	0.48		*279.50	0.00%	0.50	
C3(1)-6/2	*221.70	0.77%	123.70	-20.68%	*221.05	0.99%	49.53	-20.91%
C3(1)-6/3	*189.98	1.00%	200.97	-32.03%	*189.98	0.98%	168.19	-32.03%
C3(1)-6/6	*147.53	0.84%	11.11	-47.22%	*147.72	0.99%	9.70	-47.15%
C3(2)-P/R	*299.96	0.00%	0.31		*299.96	0.00%	0.36	
C3(2)-6/2	*223.32	1.00%	66.63	-25.55%	*223.32	1.00%	69.08	-25.55%
C3(2)-6/3	*188.01	1.00%	190.64	-37.32%	*188.01	0.99%	159.61	-37.32%
C3(2)-6/6	*145.55	1.00%	11.41	-51.48%	*145.55	1.00%	10.63	-51.48%
C3(3)-P/R	*302.25	0.00%	0.03		*302.25	0.00%	0.08	
C3(3)-6/2	*225.45	0.99%	43.94	-25.41%	*225.45	0.99%	44.67	-25.41%
C3(3)-6/3	*196.24	1.00%	110.66	-35.07%	*196.60	0.99%	151.81	-34.95%
C3(3)-6/6	*151.04	0.99%	8.75	-50.03%	*151.04	0.99%	13.80	-50.03%

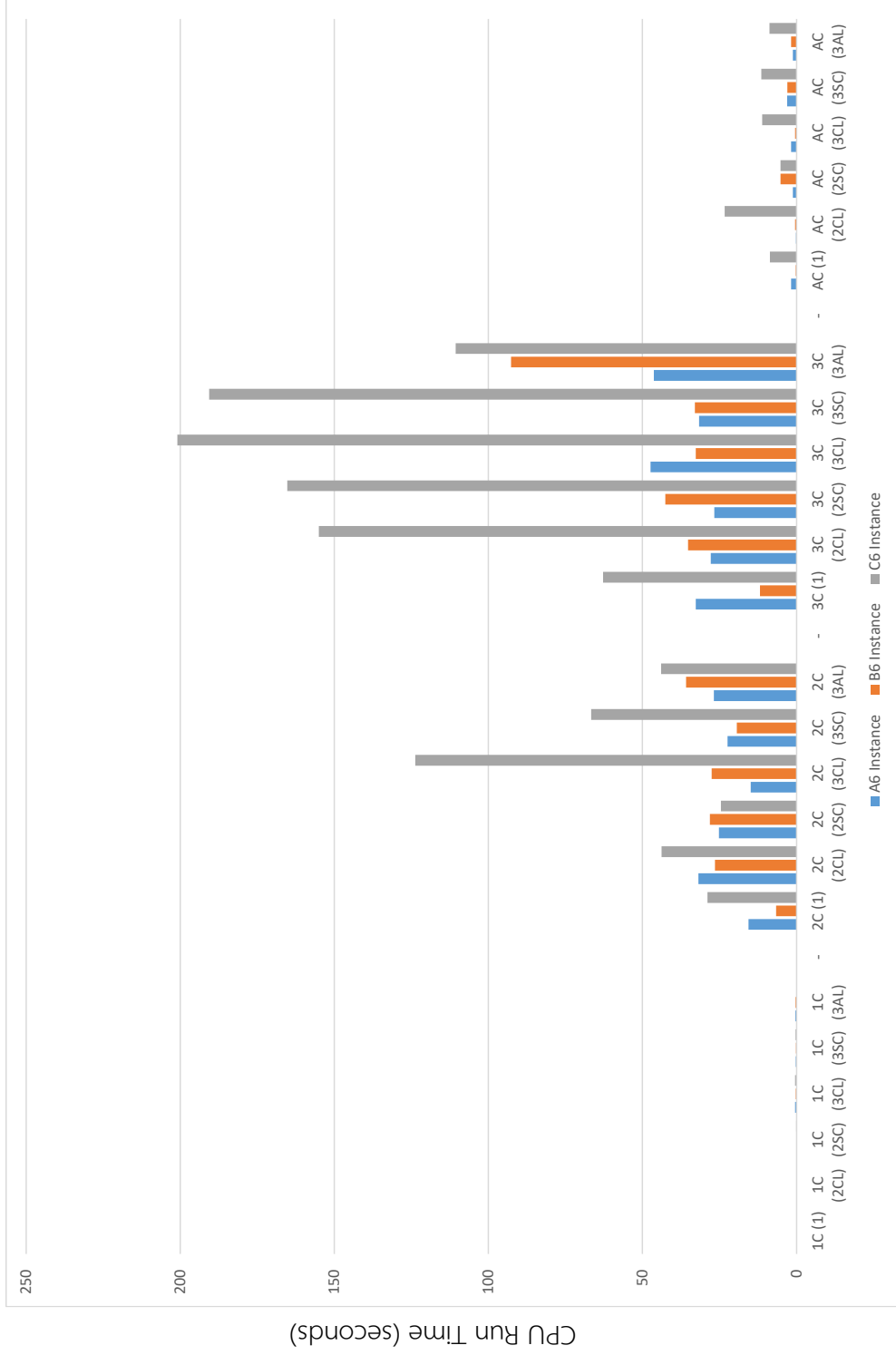


Figure 4-1 Summary of CPU run time results for instances with six requests under best-bound search

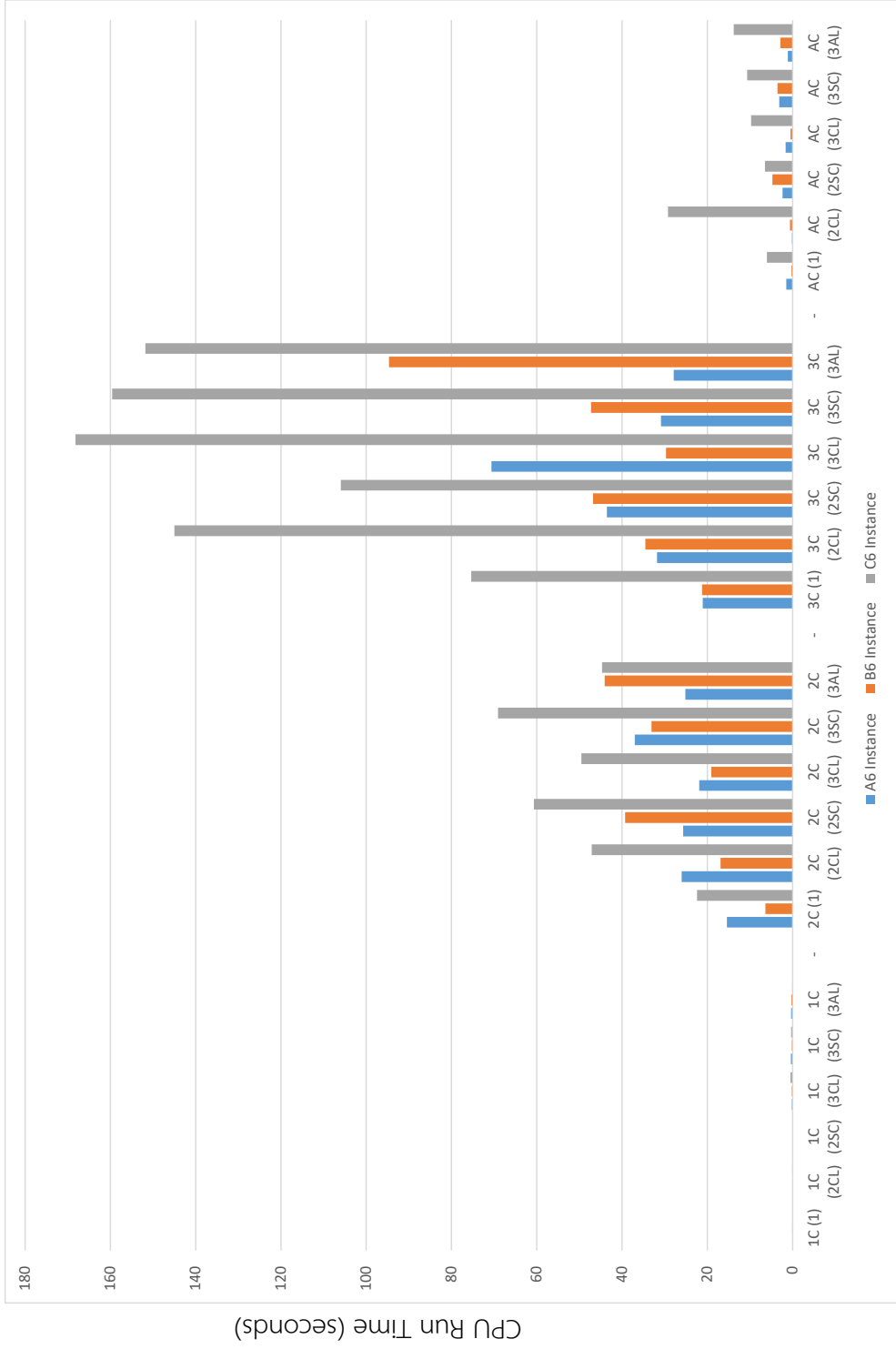


Figure 4-2 Summary of CPU run time results for instances with six requests under depth-first search

In Figure 4-1 and Figure 4-2, a summary of CPU run time results for both best-bound search and depth-first search methods are shown for the three instances A, B and C having six requests. The bar graphs are grouped according to the number of requests allowed per one route: (1C) one request per route, (2C) two requests per route, (3C) three requests per route, and (AC) all requests in one route. Moreover, the number of depots and its corresponding setup is also specified in the figures: (CL) clustered, (SC) scattered, and (AL) aligned. The number before these designations represent the number of depots in that particular instance. In both figures, it could be observed that the highest CPU run times were recorded from routes having three requests combined in each. Instances with one request per route were all solved in almost no time. With instances wherein all requests were combined in one route, computational time was not much. This may be due to the single route that the model considers and thus, less combinations are made. In summary, for sets A, B, and C, the formulated model was able to find optimal solutions for all instances in all tested combinations. Gaps recorded are about 1% or less for all instances with six requests. Best bound-search method and depth-first search showed an almost similar performance. Overall, the CPU run times recorded with six requests are generally desirable.

4.2.2 Instances with twelve requests

A more varied set of results may be observed with instances containing 12 requests. For Table 4.6, Table 4.7, and Table 4.8, the optimal solutions

found are shown with an asterisk in their objective function values. The objective values without an asterisk are the best lower bounds found for each instance that was not solved to optimality. Lower bounds are found from solving the LP relaxation of the problem. Given a larger amount of requests, each instance containing 12 requests was given a runtime limit of 21600 CPU seconds with a tolerance of 10% for the gap from the lower bound value.

In set A12 of Table 4.6, there were 9 instances out of 24 instances from the best-bound search run that showed no optimal integer solutions. On the other hand, there were 2 instances out of 24 instances from the depth-first search run which was not solved to optimality. However, the set resource limit of 21600 CPU runtime was reached for most instances except the instances that served one route per request (P/R) as well as those instances that served all requests in one route (12/12). In set A12 of Table 4.6, most instances having no optimal integer solutions are found with instances having three depots, 6 of which are from best-bound search run and 2 instances from the depth-first search run. Comparing the gaps of both branch-and-bound methods having optimal solutions, best-bound search run was able to find better values showing lower gaps from the best lower bounds found. Putting the 'per request' (P/R) instances and the 'all-in-one route' (12/12) instances aside, gaps under the best-bound search range from 31.80% to 36.60% while the gaps under the depth-first search run range from 34.68% to 45.82%. Higher gaps are generally found with instances having more depots. However, even though the runs of best-bound search having optimal solutions showed lower gaps than those of depth-first search runs, it is still notable that the latter was able to find more

optimal solutions compared to its counterpart. Based on optimal solutions found, depth-first search is preferred in this set of instances over best-bound search.

In set B12 of Table 4.7, there were 10 instances out of 24 instances from the best-bound search run that showed no optimal integer solutions. On the other hand, there were 4 instances out of 24 instances from the depth-first search run which were not solved to optimality. In this set of instances, the set resource limit of 21600 CPU runtime was also reach for instances having combined requests of two to three requests per route. The gaps found for ‘per request’ (P/R) instances and ‘all-in-one route’ instances (12/12) are about 10% or less. For instances with two to three request combination, there was only one instance wherein best-bound search and depth-first search both showed optimal solutions and best-bound search provided a lower gap from the lower bound found. Interestingly, there was one instance (*B2(2)-12/2*) wherein best-bound search was able to find an optimal solution while depth-first search failed in finding one. Again, putting the ‘per request’ (P/R) instances and the ‘all-in-one route’ (12/12) instances aside, gaps under the best-bound search range from 36.46% to 40.05% while the gaps under the depth-first search run range from 39.15% to 53.41%. Best-bound runs generally provided lower gaps from the lower bound but still, depth-first search was able to find more optimal solutions.

Table 4.6 Summary of results for instances with six requests (A12)

Instance	Best-Bound Search				Depth-First Search			
	Objective Function	Gap	CPU Time (s)	Distance Reduction	Objective Function	Gap	CPU Time (s)	Distance Reduction
A1-P/R	*668.42	0.00%	0.08		*668.42	0.00%	0.06	
A1-12/2	*460.60	31.80%	21601.03	-31.09%	*450.07	34.92%	21600.30	-32.67%
A1-12/3	266.94	-	21600.22	-	*385.47	34.68%	21600.11	-42.33%
A1-12/12	*252.24	10.00%	413.30	-62.26%	*248.26	10.00%	305.77	-62.86%
A2(1)-P/R	*652.09	3.69%	0.17		*652.09	3.69%	0.09	
A2(1)-12/2	*452.79	36.60%	21600.48	-30.56%	*442.44	38.43%	21600.22	-32.15%
A2(1)-12/3	*386.72	35.47%	21600.33	-40.70%	*419.71	44.36%	21600.20	-35.64%
A2(1)-12/12	*247.11	9.44%	300.28	-62.10%	*244.70	10.00%	204.09	-62.47%
A2(2)-P/R	*812.40	6.67%	0.13		*812.40	6.67%	0.02	
A2(2)-12/2	338.46	-	21601.11	-	*511.33	37.50%	21600.31	-37.06%
A2(2)-12/3	271.24	-	21600.69	-	*447.81	42.43%	21600.17	-44.88%
A2(2)-12/12	*238.30	8.68%	243.14	-70.67%	*239.24	10.00%	150.91	-70.55%
A3(1)-P/R	*642.60	9.89%	0.75		*622.86	7.03%	0.70	
A3(1)-12/2	266.55	-	21600.30	-	*433.08	41.91%	21600.08	-30.47%
A3(1)-12/3	238.61	-	21600.91	-	*412.24	45.69%	21601.23	-33.81%
A3(1)-12/12	*246.26	10.00%	321.00	-61.68%	*246.02	9.87%	296.14	-60.50%
A3(2)-P/R	*666.61	8.55%	0.70		*666.61	9.37%	1.63	
A3(2)-12/2	260.41	-	21600.11	-	*463.11	45.82%	21600.17	-30.53%
A3(2)-12/3	235.48	-	21600.27	-	215.14	-	21599.95	-
A3(2)-12/12	*244.69	10.00%	936.39	-63.29%	*240.86	10.00%	319.28	-63.87%
A3(3)-P/R	*677.37	9.59%	0.41		*677.37	9.59%	0.77	
A3(3)-12/2	277.77	-	21601.22	-	*449.90	42.05%	21600.63	-33.58%
A3(3)-12/3	234.71	-	21601.06	-	225.92	-	21599.98	-
A3(3)-12/12	*238.14	9.09%	260.03	-64.84%	*240.40	8.67%	268.30	-64.51%

Table 4.7 Summary of results for instances with six requests (B12)

Instance	Best-Bound Search				Depth-First Search			
	Objective Function	Gap	CPU Time (s)	Distance Reduction	Objective Function	Gap	CPU Time (s)	Distance Reduction
B1-P/R	*720.33	0.00%	0.06		*720.33	0.00%	0.02	
B1-12/2	*502.19	40.05%	21600.19	-30.28%	*493.89	42.65%	21600.16	-31.44%
B1-12/3	260.22	-	21600.53	-	*454.02	45.42%	21600.19	-36.97%
B1-12/12	*259.01	10.00%	132.25	-64.04%	*255.56	10.00%	175.39	-64.52%
B2(1)-P/R	*678.19	0.69%	0.08		*678.19	0.69%	0.09	
B2(1)-12/2	273.27	-	21600.14	-	263.92	-	21599.88	-
B2(1)-12/3	249.88	-	21600.69	-	*444.75	47.81%	21600.11	-34.42%
B2(1)-12/12	*255.24	9.60%	239.30	-62.36%	*254.98	9.85%	182.84	-62.40%
B2(2)-P/R	*792.16	0.64%	0.13		*792.16	0.64%	0.14	
B2(2)-12/2	*520.75	36.46%	21600.55	-34.26%	318.22	-	21599.91	-
B2(2)-12/3	276.50	-	21600.30	-	*430.87	39.15%	21600.20	-45.61%
B2(2)-12/12	*259.21	9.55%	221.44	-67.28%	*252.06	9.90%	128.11	-68.18%
B3(1)-P/R	*697.49	8.92%	0.19		*697.49	8.92%	0.19	
B3(1)-12/2	253.67	-	21600.42	-	*479.59	47.69%	21600.22	-31.24%
B3(1)-12/3	238.66	-	21600.33	-	223.99	-	21599.95	-
B3(1)-12/12	*256.37	10.00%	255.42	-63.24%	*249.65	9.99%	139.67	-64.21%
B3(2)-P/R	*727.15	9.77%	1.39		*727.15	9.77%	1.33	
B3(2)-12/2	268.31	-	21600.33	-	*494.89	49.45%	21600.25	-31.94%
B3(2)-12/3	226.45	-	21600.45	-	*463.06	53.41%	21600.14	-36.32%
B3(2)-12/12	*242.40	8.90%	74.17	-66.66%	*247.98	10.00%	270.31	-65.90%
B3(3)-P/R	*681.27	2.30%	0.11		*681.27	2.30%	0.16	
B3(3)-12/2	264.65	-	21600.64	-	*464.57	45.66%	21600.09	-31.81%
B3(3)-12/3	234.83	-	21600.30	-	226.96	-	21599.98	-
B3(3)-12/12	*261.29	10.00%	394.81	-61.65%	*252.13	10.00%	184.69	-62.99%

Table 4.8 Summary of results for instances with six requests (C12)

Instance	Best-Bound Search				Depth-First Search			
	Objective Function	Gap	CPU Time (s)	Distance Reduction	Objective Function	Gap	CPU Time (s)	Distance Reduction
C1-P/R	*961.39	0.00%	0.08		*961.39	0.00%	0.11	
C1-12/2	*606.39	31.61%	21601.06	-36.93%	*579.52	30.61%	21600.09	-39.72%
C1-12/3	322.51	-	21600.30	-	*488.39	37.51%	21600.22	-49.20%
C1-12/12	*274.64	13.36%	21600.28	-71.43%	*274.64	17.37%	21609.42	-71.43%
C2(1)-P/R	*709.94	3.44%	0.13		*709.94	3.44%	0.41	
C2(1)-12/2	291.77	-	21601.03	-	*503.08	44.44%	21600.05	-29.14%
C2(1)-12/3	254.99	-	21600.25	-	*445.49	44.44%	21600.17	-37.25%
C2(1)-12/12	*255.87	7.76%	3396.80	-63.96%	*257.73	10.00%	1043.22	-63.70%
C2(2)-P/R	*854.92	7.70%	0.13		*854.92	7.70%	0.19	
C2(2)-12/2	305.89	-	21600.47	-	*535.74	47.34%	21600.36	-37.33%
C2(2)-12/3	239.47	-	21600.22	-	*468.17	51.11%	21600.22	-45.24%
C2(2)-12/12	*260.02	14.56%	21600.23	-69.59%	*260.02	14.87%	21610.56	-69.59%
C3(1)-P/R	*687.27	9.55%	0.61		*677.97	8.31%	0.55	
C3(1)-12/2	284.64	-	21600.33	-	269.03	-	21599.92	-
C3(1)-12/3	249.58	-	21600.58	-	241.24	-	21599.88	-
C3(1)-12/12	*260.24	10.00%	1703.55	-62.13%	*256.16	10.00%	770.92	-62.22%
C3(2)-P/R	*738.34	9.45%	0.70		*738.34	9.45%	1.73	
C3(2)-12/2	254.88	-	21601.11	-	*534.54	54.09%	21600.09	-27.60%
C3(2)-12/3	230.71	-	21600.36	-	221.44	-	21599.98	-
C3(2)-12/12	*254.65	10.00%	1329.38	-65.51%	*248.47	8.69%	1562.55	-66.35%
C3(3)-P/R	*690.72	9.47%	0.77		*690.72	9.47%	0.63	
C3(3)-12/2	256.96	-	21600.28	-	*491.09	49.51%	21600.09	-28.90%
C3(3)-12/3	221.16	-	21600.45	-	210.32	-	21600.06	-
C3(3)-12/12	*248.47	10.84%	21600.66	-64.03%	*260.02	16.79%	21600.23	-62.36%

In set C12 of Table 4.8, there were 11 instances out of 24 instances under the best-bound search run which showed no optimal integer solutions. Depth-first search run on the other hand showed 4 instances out of 24 instances with no optimal integer solutions. In set A12 and set B12, only those instances with two to three request combination reached the set resource time limit but in set C12, most of the instances reached the given 21 600 CPU run time limit including the instances with ‘all-in-one route’ variation. In the runs with best-bound search method, only one instance among the two to three combined request instances, showed an optimal solution which had a slightly higher gap from the lower bound than that of the depth-first search. Putting the ‘per request’ (P/R) instances and the ‘all-in-one route’ (12/12) instances aside, gaps under the depth-first search range from 30.61% to 54.09%.

Table 4.9 Summary of model performance results for instances with 12 requests

	No. of Instances Per Set	Best-Bound Search	%	Depth-First Search	%	% Difference	
A12	24	15	62.50%	22	91.67%	29.17%	
B12	24	14	58.33%	20	83.33%	25.00%	
C12	24	13	54.17%	20	83.33%	29.17%	
						27.78%	AVERAGE

In summary, in terms of optimal solutions found, depth-first search performed better than best-bound search in solving instances with twelve requests. A summary of model performance results may be seen from Table 4.9. Depth-first search performed better than best-bound in about 28% of the time. However, in cases wherein both branch-and-bound algorithm methods

were able to provide optimal solutions, lower gaps from the lower bound found by best-bound search are better than that of depth-first search.

Table 4.10 Average percentage of gaps from lower bounds of 12-request instances

	No. of Requests per Route			
	1	2	3	All
Best Bound Search	5.57%	35.30%	35.47%	10.10%
Depth First Search	5.39%	43.47%	44.18%	10.89%

On the other hand, from the recorded gaps from lower bounds of instances having 12 requests, an average percentage may be seen from Table 4.10. From an initial set tolerance of 10% for the instances, only the requests with one request per route and all requests in one route were able to find satisfying optimal solutions. With instances having a combined requests of two and three, gaps were about 35% for best-bound search and 44% for depth-first search. These gaps from the lower bound are way higher than the set tolerance although it is important to note that this does not necessarily mean that the optimal solutions found were not good. There is a possibility that the model was not only able to find a good lower bound for the particular instance but the optimal solution may be the best one. Moreover, even though the gaps found from the lower bound are higher than the set tolerance, comparing the objective values found of combined requests from the typical “one-request-per-route” service still suggest significant distance reduction. This is further discussed in the next section.

4.3 Improvement through Combining Requests

This section discusses the reduction of overall distances that resulted from the combination of two to three requests in one route as compared to the instances that served one request per one route. The reduction of distances are discussed separately between instances with six requests and instances with twelve requests.

4.3.1 Instances with six requests

From Table 4.3 to Table 4.5, the percentage of distance reduction for instances with six requests can be seen in the 5th column for best-bound search method and 9th column for the depth-first search method. The distance reduction is obtained by comparing the instances with combined requests to the instances that serve only one request per one route.

In Table 4.11, an average percentage between the three sets of instances A, B, and C is shown with each instance. Naturally, the reduction for each instance increased as the number of requests included in one route is also increased. For instances with two requests allowed per route, average distance reduction ranges from 23.72% to 31.61%. For instances with three requests allowed per route, average distance reduction ranges from 33.32% to 41.15%. Lastly, for instances having all requests in one route, average distance reduction ranges from 49.09% to 56.22%.

Table 4.11 Average percentage of distance reduction of combined requests compared to 'one request per route' service (A6, B6, and C6)

Instance	Best-Bound Search	Depth-First Search
1-P/R		
1-6/2	-31.61%	-31.61%
1-6/3	-41.58%	-41.58%
1-6/6	-56.15%	-56.22%
2(1)-P/R		
2(1)-6/2	-26.27%	-26.27%
2(1)-6/3	-36.23%	-36.23%
2(1)-6/6	-51.21%	-51.21%
2(2)-P/R		
2(2)-6/2	-30.01%	-30.01%
2(2)-6/3	-41.15%	-41.15%
2(2)-6/6	-55.10%	-55.10%
3(1)-P/R		
3(1)-6/2	-24.69%	-24.76%
3(1)-6/3	-34.39%	-34.40%
3(1)-6/6	-50.19%	-50.17%
3(2)-P/R		
3(2)-6/2	-27.29%	-27.29%
3(2)-6/3	-39.43%	-39.43%
3(2)-6/6	-51.85%	-51.85%
3(3)-P/R		
3(3)-6/2	-23.72%	-23.72%
3(3)-6/3	-33.36%	-33.32%
3(3)-6/6	-49.09%	-49.09%

It is apparent that combining more than one request in one particular route would lead into a reduction of overall distance travelled. However, through the instances that were tested, the huge percentage reduction is given emphasis. Furthermore, it is also important to note that with the combination of requests, the number of needed vehicles to satisfy all demands would also reduce as the number of combined requests is increased per route. Since the formulation also chooses the best depots simultaneously, it is possible that not all depots will be used in the network. In Table 4.12, a summary of distance reduction results for instances having six requests is shown for both best-bound search method and depth-first search method.

Table 4.12 Summary of distance reduction results of six-request instances

	No. of Requests		
	2	3	All
Best Bound Search	-27.26%	-37.69%	-52.26%
Depth First Search	-27.28%	-37.69%	-52.27%

4.3.2 Instances with twelve requests

From Table 4.6 to Table 4.8, the percentage of distance reduction for instances with twelve requests can be seen in the 5th column for best-bound search method and 9th column for the depth-first search method. In Table 4.13, an average percentage between the three sets of instances A, B, and C is shown with each instance.

Table 4.13 Average percentage of distance reduction of combined requests compared to 'one request per route' service (A12, B12, and C12)

Instance	Best-Bound Search	Depth-First Search
1-P/R		
1-12/2	-32.77%	-34.61%
1-12/3	-	-42.83%
1-12/12	-65.91%	-66.27%
2(1)-P/R		
2(1)-12/2	-30.56%	-30.64%
2(1)-12/3	-40.70%	-35.77%
2(1)-12/12	-62.81%	-62.86%
2(2)-P/R		
2(2)-12/2	-34.26%	-37.20%
2(2)-12/3	-	-45.24%
2(2)-12/12	-69.18%	-69.44%
3(1)-P/R		
3(1)-12/2	-	-30.85%
3(1)-12/3	-	-33.81%
3(1)-12/12	-62.35%	-62.31%
3(2)-P/R		
3(2)-12/2	-	-30.02%
3(2)-12/3	-	-36.32%
3(2)-12/12	-65.16%	-65.37%
3(3)-P/R		
3(3)-12/2	-	-31.43%
3(3)-12/3	-	-
3(3)-12/12	-63.51%	-63.29%

The missing percentages found under the best-bound search are those wherein the model was not able to find optimal solution. For instances with two requests allowed per route, average distance reduction ranges from 30.02% to 37.20%. For instances with three requests allowed per route, average distance reduction ranges from 33.81% to 45.24%. Lastly, for instances having all requests in one route, average distance reduction ranges from 62.35% to 69.44%. In Table 4.14, a summary of distance reduction results for instances having twelve requests is shown for both best-bound search method and depth-first search method.

Table 4.14 Summary of distance reduction results of twelve-request instances

	No. of Requests		
	2	3	All
Best Bound Search	-32.53%	-40.70%	-64.82%
Depth First Search	-32.46%	-38.80%	-64.92%

With the instances having twelve requests, large gaps from the lower bounds can be observed from the optimal solutions found in Table 4.6, Table 4.7, and Table 4.8. The gaps for the instances with combined requests range from about 30% to 54%. This gap from the lower bound is still very large. However, even if this is the case, the improvement of minimized distances from combining the requests is still very noticeable as shown in Table 4.13 and Table 4.14.



Figure 4-3 Distance reduction results from computational experiments

In Figure 4-3, the distance reduction results from instances having six requests and twelve requests, respectively, are shown. It could be remembered that for instances with combined requests (2 to 3 requests per route), the gaps from lower bounds for six requests are very desirable (1% or less) while considerably large gaps were recorded from instances having twelve requests (around 35% for best-bound and around 44% for depth-first). However, even if this is the case, both sets of instances were still able to show a significant reduction in the overall distances computed from serving all demands.

4.4 Multiple-Depot Variation

The effect of the number of allowed requests per route on the variation of multiple depots on the distance objective function in instances having six requests and twelve requests are shown in Figure 4-4 and Figure 4-5, respectively. In the figures, results from depth-first search methods are used since the said method was able to find more optimal solutions than its counterpart method.

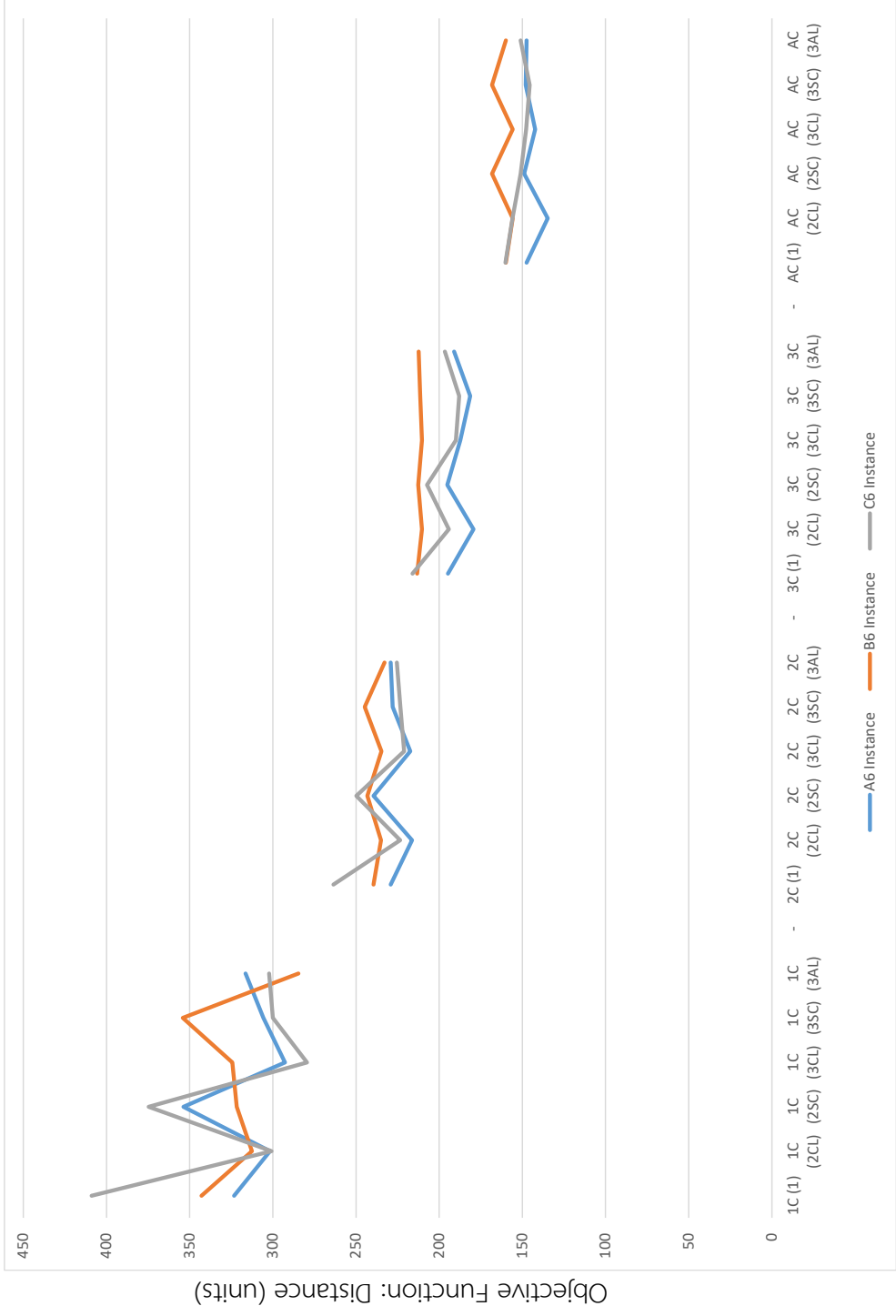


Figure 4-4 Effect of multiple-depot variation on instances having six requests (depth-first search)

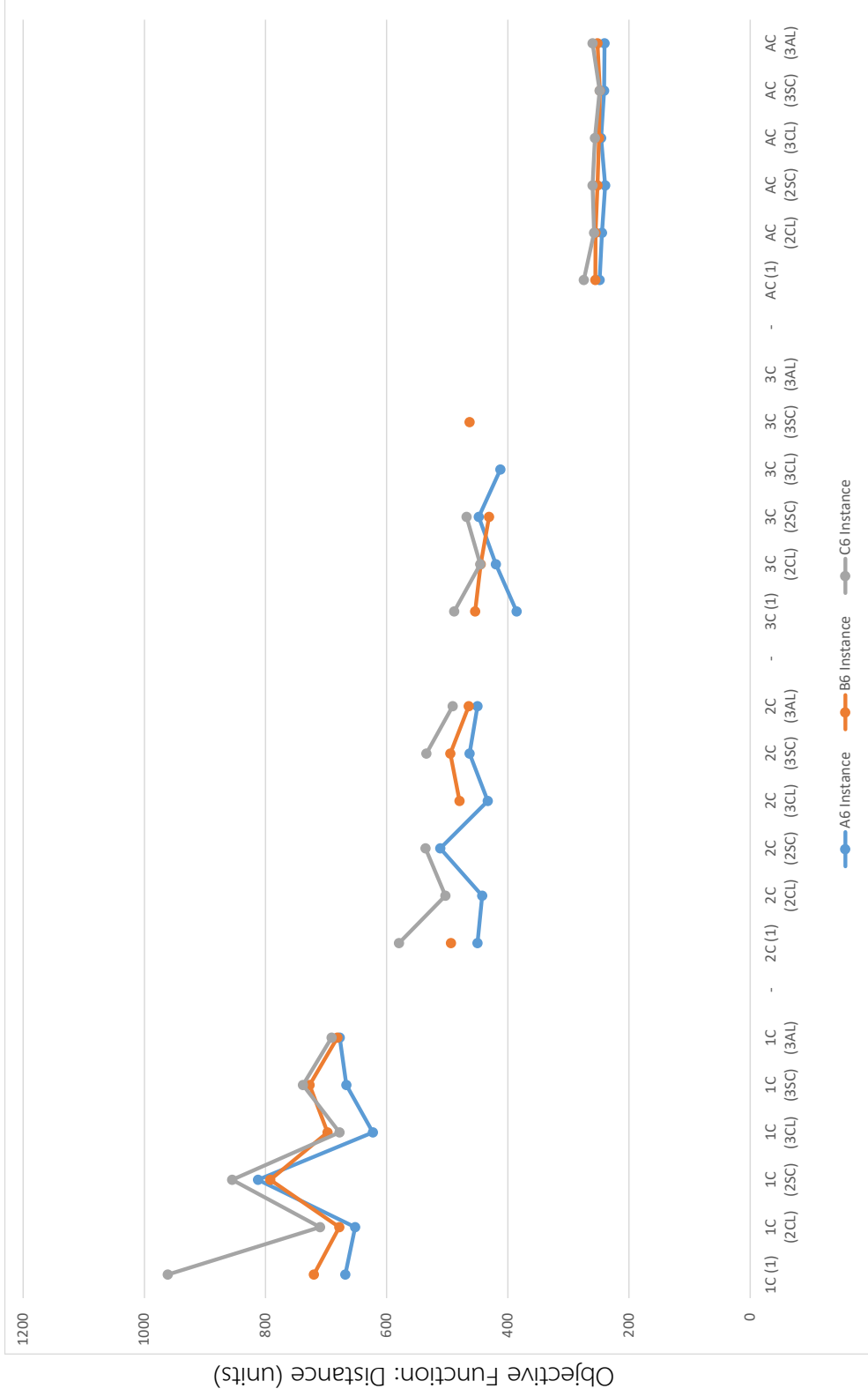


Figure 4-5 Effect of multiple-depot variation on instances having twelve requests (depth-first search)

Starting from the graph on the left of both figures, data are grouped according to the number of requests allowed in one route: (1C) one request per route, (2C) two requests per route, (3C) three requests per route, and (AC) all requests in one route. Moreover, the number of depots and its corresponding setup is also specified in the figures: (CL) clustered, (SC) scattered, and (AL) aligned. The number before these designations represent the number of depots in that particular instance. Each instance is graphed against its corresponding distance objective value found from computational experiments.

In both figures, the distance reduction is noticeable as the number of requests is increased in each set of instances. This is discussed in the previous section. But aside from significant distance reduction, the effect of the number of requests allowed per route on the varying number of multiple depots (1, 2, and 3) as well as its distribution of locations (clustered, scattered, and aligned) is also very noticeable in Figure 4-4 and Figure 4-5. As the number of allowed requests per route is increased, the differences in the distance objective values become minimal. The graphs on the left of both figures show a very varied series of values as their multiple depots and locations were modified. As the number of requests is increased, the graphs that follow on the right become less varied and the illustrations become more levelled than the previous case.

4.5 Advantages and Disadvantages of Proposed Model

The formulated model in this paper highlights a choice of the number of requests that are allowed to be served in one route. In this way, the overall distance travelled in each instance may be compared and the most desirable result may be

chosen for application. Moreover, the model also simultaneously assigns the best depot that should serve one particular route. In this way, aside from minimizing the overall total distance through the combination of requests, characteristics of parameters such as depot location, number of serving vehicles, and cost per route may also be determined. Another advantage of the model is its flexible sequencing of pickup and delivery nodes. Once a commodity is picked up from its node, its corresponding delivery node does not have to come right after. Multiple pickups may be done first and multiple deliveries may follow depending on what results into the least overall distance objective value.

On the other hand, due the arc-based formulation of the model, it generally takes a considerable amount of time as the instances become larger. The formulated model may only be directly applicable to smaller instances. Moreover, only a small number of requests is tested to be combined in this study and each route included in the overall solution contains similar amount of requests. The formulated model is not designed to construct routes with different number of combined requests. The model also does not allow multiple commodities to be picked up from a single node.

Between depth-first search and best-bound search, the former is preferred due to its speed in finding an optimal solution for a certain instance although it is also important to note that although best-bound search takes time, once it finds a solution, it is observed that the found solution improves quicker than that of depth-first search. Another major advantage of depth-first search is that it only takes up a small memory of the computer and can continuously run for hours. Best-bound search takes a considerable amount of memory and stops running the program once the computer

memory limit is reached. This hinders the method in finding a solution when ran for longer hours.

In solving larger instances in future research, the formulated model may still be used by clustering the set of given requests first to break them down into smaller instances. In this way, the formulated model may work on the smaller instances separately and be able to find optimal solutions. Finally, it is suggested to use the model in real-life applications to analyze its feasibility and other important factors of the problem especially with logistics companies specializing in pickup and delivery services.



CHAPTER V

SUMMARY AND CONCLUSION

5.1 Summary

Overall, there were three sets of instances tested in this research (A, B & C). Each set contained two subsets, one with six requests and one with twelve requests. For each instance, both best-bound search method and depth-first search method were performed. In total, there were 288 instances tested in the research.

The performance of the model for the instances with six requests was excellent. Minor differences were found between the two branch-and-bound algorithm methods: best-bound search and depth-first search. The model was also able to find a low gap of 1% or less from the lower bound value for all instances tested with six requests.

With instances having twelve requests, depth-first search performed better than best-bound search in terms of finding optimal solutions. Depth-first search performed better than best-bound in about 28% of the time. However, in cases wherein both branch-and-bound algorithm methods were able to provide optimal solutions, lower gaps from the lower bound found by best-bound search were found to be better than that of depth-first search.

Comparing instances with routes having combined requests from instances with routes serving only one request per route, distance reduction was found to be very significant for both instances having six requests and twelve requests. Even though

instances with twelve requests showed huge gaps from lower bounds, overall distance reduction was still considerably huge.

In terms of depot variation, as the number of allowed requests per route is increased, the differences in the distance objective values become minimal. Depots were varied according to number (1, 2, and 3) and according to location distribution (clustered, scattered, and aligned).

The model showed advantages in terms of having choices on the number of desired requests per route as well as the simultaneous generation of routes and depot assignment. Disadvantages included limitations of model such as handling larger instances and handling multiple requests from similar nodes.

5.2 Conclusion

The model formulated in this research performs well on one-to-one static pickup and delivery problems with instances of up to 12 requests. It can directly be applied to problems of such size wherein distance is the main objective function. The formulation also highlights a choice for the allowed number of requests per route and the number of depots to be used in one particular solution set. In this way, aside from optimizing costs, users of the formulation may also analyze additional information such as depot location, number of serving vehicles and cost per route. This provides more flexibility on the management side in choosing what routes to implement to satisfy all demands. However, it is important to note that the proposed model only considers

distances to construct routes. It is suggested to include more parameters such as time windows, vehicular capacity and other important costs in future research.

Between best-bound search method and depth-first search method of the branch-and-bound algorithm, the latter is preferred if speed is a main priority in finding optimal solutions. Depth-first search method is also preferred because it takes up less memory from the computer compared to best-bound search method. Best-bound search may be more applicable for long term planning of routes wherein more time can be spared for computing time.

On the other hand, in solving a particular instance, different setups of multiple depots prove to be of much effect if requests are served individually in one route (one request per route). The effect of the multiple depot setup on the distance objective value becomes minimal as the number of combined requests in one route is increased. Therefore, if multiple depots are considered in one particular problem, it is strongly suggested to combine requests that will result into a set of routes in order to save costs.

Additionally, with the formulation in GAMS, parameters and constraints are easily understood due to the structure of the model. This, in turn, gives more room for improvement especially for the development of more advanced heuristics.

5.3 Recommendations

The research considered Euclidean distances as the main parameter in the instances solved in the computational experiments. For future work, it is suggested to include other parameters such as time windows, vehicle capacity as well as provider and customer preferences. Data from field may also be gathered to have a better representation of the transportation network in the instances.

In solving larger instances in future research, the formulated model may still be used by clustering the set of given requests first to break them down into smaller instances. In this way, the formulated model may work on smaller instances separately and be able to find optimal solutions.

Future work for the formulated model may also concentrate on integrating it with other solution approaches such as nearest neighborhood algorithm, insertion method, as well as applying column generation. Moreover, since this paper focused on a static case of PDP, the research may be extended to solve its dynamic case.

Finally, it is suggested to use the model in real-life applications to analyze its feasibility and other important factors of the problem especially with logistics companies specializing in pickup and delivery services. Aside from the initial motivation of the research in using motorcycle taxis to deliver documents and small parcels, other applications may also include food deliveries wherein more establishments will be able to provide pickup and delivery services to a wider range of customers.

REFERENCES

- Battarra, M., Cordeau, J.-F., & Iori, M. Chapter 6: Pickup-and-Delivery Problems for Goods Transportation *Vehicle Routing* (pp. 161-191).
- Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I., & Laporte, G. (2007). Static pickup and delivery problems: a classification scheme and survey. *Top*, 15(1), 1-31.
- Berbeglia, G., Cordeau, J.-F., & Laporte, G. (2010). Dynamic pickup and delivery problems. *European Journal of Operational Research*, 202(1), 8-15.
- Bräysy, O., & Hasle, G. Chapter 12: Software Tools and Emerging Technologies for Vehicle Routing and Intermodal Transportation *Vehicle Routing* (pp. 351-380).
- Carrabs, F., Cordeau, J.-F., & Laporte, G. (2007). Variable Neighborhood Search for the Pickup and Delivery Traveling Salesman Problem with LIFO Loading. *INFORMS Journal on Computing*, 19(4), 618-632. doi:doi:10.1287/ijoc.1060.0202
- Chattopadhyay, D. (1999). Application of general algebraic modeling system to power system optimization. *IEEE transactions on power systems*, 14(1), 15-22.
- Christiansen, M., & Fagerholt, K. Chapter 13: Ship Routing and Scheduling in Industrial and Tramp Shipping *Vehicle Routing* (pp. 381-408).
- 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. doi:doi:10.1287/opre.12.4.568
- Cordeau, J.-F. (2006). A Branch-and-Cut Algorithm for the Dial-a-Ride Problem. *Operations Research*, 54(3), 573-586. doi:doi:10.1287/opre.1060.0283
- Cordeau, J.-F., Iori, M., Laporte, G., & Salazar González, J. J. (2010). A branch-and-cut algorithm for the pickup and delivery traveling salesman problem with LIFO loading. *Networks*, 55(1), 46-59. doi:10.1002/net.20312
- Cordeau, J.-F., & Laporte, G. (2003). A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological*, 37(6), 579-594. doi:[http://dx.doi.org/10.1016/S0191-2615\(02\)00045-0](http://dx.doi.org/10.1016/S0191-2615(02)00045-0)
- Cordeau, J.-F., Laporte, G., & Ropke, S. (2008). *Recent models and algorithms for one-to-one pickup and delivery problems*: Springer.

- Dantzig, G. B., & Ramser, J. H. (1959). The Truck Dispatching Problem. *Manage. Sci.*, 6(1), 80-91. doi:10.1287/mnsc.6.1.80
- Detti, P., Papalini, F., & Lara, G. Z. M. d. A multi-depot dial-a-ride problem with heterogeneous vehicles and compatibility constraints in healthcare. *Omega*. doi:<http://dx.doi.org/10.1016/j.omega.2016.08.008>
- Doerner, K. F., & Salazar-González, J.-J. Chapter 7: Pickup-and-Delivery Problems for People Transportation *Vehicle Routing* (pp. 193-212).
- Dumitrescu, I., Ropke, S., Cordeau, J.-F., & Laporte, G. (2008). The traveling salesman problem with pickup and delivery: polyhedral results and a branch-and-cut algorithm. *Mathematical Programming*, 121(2), 269-305. doi:10.1007/s10107-008-0234-9
- Eglese, R., & Bektaş, T. Chapter 15: Green Vehicle Routing *Vehicle Routing* (pp. 437-458).
- Ekvitthayavechnukul, C. (2016). Thailand 2016: Top trends to look out for in nascent but fast-growing e-commerce market (Online Article). Available from Deal Street Asia Retrieved January 6, 2016
<http://www.dealstreetasia.com/stories/thailand-2016-top-trends-look-nascent-fast-growing-e-commerce-market-24059/>
- Fabry, J. (2007). Dynamic Messenger Problem. *Communications*, 9(4), 66-69.
- Fabry, J. (2015). INSERTION METHOD FOR MULTIPLE MESSENGER PROBLEM WITH MULTIPLE DEPOTS. *RE-AGGREGATION HEURISTICS FOR THE LARGE LOCATION PROBLEMS WITH LEXICOGRAPHIC MINIMAX OBJECTIVE*, 11.
- Fábry, J., & Kobzareva, M. (2012). Multiple Messenger Problem. *Proc. of Mathematical Methods in Economics '12, Karvina*, 141-147.
- Golden, B. L., Kovacs, A. A., & Wasil, E. A. Chapter 14: Vehicle Routing Applications in Disaster Relief *Vehicle Routing* (pp. 409-436).
- Hernández-Pérez, H., & Salazar-González, J.-J. (2003). The one-commodity pickup-and-delivery travelling salesman problem *Combinatorial Optimization—Eureka, You Shrink!* (pp. 89-104): Springer.

- Hernández-Pérez, H., & Salazar-González, J.-J. (2004). Heuristics for the one-commodity pickup-and-delivery traveling salesman problem. *Transportation science*, *38*(2), 245-255.
- Hernández-Pérez, H., & Salazar-González, J.-J. (2009). The multi-commodity one-to-one pickup-and-delivery traveling salesman problem. *European Journal of Operational Research*, *196*(3), 987-995.
- Irnich, S., Toth, P., & Vigo, D. Chapter 1: The Family of Vehicle Routing Problems *Vehicle Routing* (pp. 1-33).
- Marković, N., Nair, R., Schonfeld, P., Miller-Hooks, E., & Mohebbi, M. (2015). Optimizing dial-a-ride services in Maryland: Benefits of computerized routing and scheduling. *Transportation Research Part C: Emerging Technologies*, *55*, 156-165. doi:<http://dx.doi.org/10.1016/j.trc.2015.01.011>
- Parragh, S. N., Doerner, K. F., & Hartl, R. F. (2008a). A survey on pickup and delivery problems. Part I: transportation between customers and depot. *Journal für Betriebswirtschaft*, *58*, 21-51.
- Parragh, S. N., Doerner, K. F., & Hartl, R. F. (2008b). A survey on pickup and delivery problems. Part II: transportation between pickup and delivery locations. *Journal für Betriebswirtschaft*, *58*, 81-117.
- Ropke, S., & Cordeau, J.-F. (2009). Branch and Cut and Price for the Pickup and Delivery Problem with Time Windows. *Transportation science*, *43*(3), 267-286. doi:doi:10.1287/trsc.1090.0272
- Ropke, S., Cordeau, J.-F., & Laporte, G. (2007). Models and branch-and-cut algorithms for pickup and delivery problems with time windows. *Networks*, *49*(4), 258-272. doi:10.1002/net.20177
- Ropke, S., & Pisinger, D. (2006). An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows. *Transportation science*, *40*(4), 455-472. doi:doi:10.1287/trsc.1050.0135
- Savelsbergh, M., & Sol, M. (1998). Drive: Dynamic Routing of Independent Vehicles. *Operations Research*, *46*(4), 474-490. doi:doi:10.1287/opre.46.4.474
- Savelsbergh, M. W., & Sol, M. (1995). The general pickup and delivery problem. *Transportation science*, *29*(1), 17-29.

The Thailand Life. (2013). The Lucrative Business of Being A Motorbike Taxi in Thailand (Photo). Blog Retrieved from <http://www.thethailandlife.com/the-business-of-motorbike-taxis-in-thailand>

Yuen, J. (2015). Opportunities in Thailand's Logistics Market (Online Article). Available from Hong Kong Trade Development Council Research Retrieved June 8, 2015 <http://economists-pick-research.hktdc.com/business-news/article/Research-Articles/Opportunities-in-Thailand-s-Logistics-Market/rp/en/1/1X000000/1X0A2LSH.htm>





APPENDIX

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

GENERAL ALGEBRAIC MODELING SYSTEM (GAMS)

Coding Template (Six Requests)

(1) SETS

\$ontext

Static One-to-One Multiple Vehicle Pickup and Delivery Problem with Multiple Depots

\$offtext

```
set v          vertices          /node1*node18/;
set i(v)       nodes (from)     /node1*node15/;
set j(v)       nodes (to)       /node4*node18/;
set depoti(i)  starting depots  /node1*node3/;
set depotj(j)  ending depots    /node16*node18/;
set customer(v) pickup-delivery /node4*node15/;
set pickup(v)  pickup nodes     /node4*node9/;
set delivery(v) delivery nodes  /node10*node15/;

set RTS        no. of routes    //;

scalar N       no. of requests  //;
scalar CMB     max links        //;
```

(2) INSTANCE DATA (PARAMETERS)

```
*-----*
* Data
*-----*

table nodedata(v,*)
      xpoint      ypoint      pair
node1              0
node2              0
node3              0
node4              1
node5              2
node6              3
node7              4
node8              5
node9              6
node10             Node
node11             Coordinates
node12             3
node13             4
node14             5
node15             6
node16             0
node17             0
node18             0
;
```

```

parameter xc(v);
xc(v) = nodedata(v,'xpoint');

parameter yc(v);
yc(v) = nodedata(v,'ypoint');

parameter pair(v);
pair(v) = nodedata(v,'pair');

parameter s(*,*);
s(i,j) = SQRT(SQR(xc(i)-xc(j))+SQR(yc(i)-yc(j)));

```

(3) EQUATIONS

```

*-----*
* Problem Model
*-----*

OPTION RESLIM = 3600;
OPTION ITERLIM = 100000;
OPTION WORK = 1000;
OPTION OPTCR = 0.01;

variable tcost                objective variable: total cost of all routes
variable xp(*,*,RTS)         link-path selection
binary variable xp            if path 'i-j' is used
integer variable vseq(v)     for sequence of 'v' nodes

```

EQUATIONS

objective	minimizes total cost for all routes
sttdepot(RTS)	each route should have one starting depot
enddepot(RTS)	each route should have one ending depot
conone(RTS)	(1) route should end in the starting origin depot
contwo(RTS)	(2) route should end in the starting origin depot
conthree(RTS)	(3) route should end in the starting origin depot
combination(RTS)	forces the combination goal
frlimit(customer)	limits to only one outgoing link for customer nodes
tolimit(customer)	limits to only one incoming link for customer nodes
conservation(customer,RTS)	keeps the incoming and outgoing link in one route
onepair(RTS)	keeps the pickup-delivery nodes in one route
subtour(customer,j)	avoids sub-tour for the routes
precedence	makes sure that pickup comes before its corresponding delivery node
vtov	avoids vertex to vertex link
ditodj	avoids depoti to depotj link
djtodi	avoids depotj to depoti link
ditodel	avoids depoti to delivery link
picktodj	avoids pickup to depotj link
dtopick	avoids delivery to pickup link
;	


```

objective..
stdepot (RTS) ..
enddepot (RTS) ..

conone (RTS) ..
contwo (RTS) ..
conthree (RTS) ..

combination (RTS) ..
flimit (customer) ..
tolimit (customer) ..

conservation (customer, RTS) ..
onepair (RTS) ..

subtour (customer, j) ..
precedence (customer, customer+n) ..

vtov (v, v, RTS) ..
ditodj (depoti, depotj, RTS) ..
djtodi (depotj, depoti, RTS) ..
ditodel (depoti, delivery, RTS) ..
picktodj (pickup, depotj, RTS) ..
dtopick (customer, customer-n, RTS) ..

tcost =e= sum((i,j,RTS),xp(i,j,RTS)*s(i,j));
sum((depoti,pickup),xp(depoti,pickup,RTS)) =e= 1;
sum((delivery,depotj),xp(delivery,depotj,RTS)) =e= 1;

sum((pickup),xp('node1',pickup,RTS))-sum((delivery),xp(delivery,'node16',RTS)) =e= 0;
sum((pickup),xp('node2',pickup,RTS))-sum((delivery),xp(delivery,'node17',RTS)) =e= 0;
sum((pickup),xp('node3',pickup,RTS))-sum((delivery),xp(delivery,'node18',RTS)) =e= 0;

sum((i,j),xp(i,j,RTS)) =e= CMB;
sum((j,RTS),xp(customer,j,RTS)) =l= 1;
sum((i,RTS),xp(i,customer,RTS)) =l= 1;

sum(j,xp(customer,j,RTS))-sum(i,xp(i,customer,RTS)) =e= 0;
sum((pickup,j),xp(pickup,j,RTS)*SQRT(pair(pickup)))-sum((delivery,j),xp(delivery,j,RTS)*SQRT(pair(delivery))) =e= 0;

sum(RTS,vseq(customer)-vseq(j)+(2*N+1)*xp(customer,j,RTS)) =l= 2*N;
vseq(customer)-vseq(customer+n) =l= 0;

xp(v,v,RTS) =e= 0;
xp(depoti,depotj,RTS) =e= 0;
xp(depotj,depoti,RTS) =e= 0;
xp(depoti,delivery,RTS) =e= 0;
xp(pickup,depotj,RTS) =e= 0;
xp(customer,customer-n,RTS) =e= 0;

```

(4) OPTIONS AND OUTPUT

```
model pdpmultiple
/objective, sttdepot, enddepot, conone, contwo, conthree, combination, frlimit, tolimit, conservation, onepair, subtour, precedence,
vtov, ditodj, djtodj, ditodel, picktodj, dtopick/;

pdpmultiple.solprint = 2;
pdpmultiple.limrow = 0;
pdpmultiple.limcol = 0;
pdpmultiple.solveLink = 2;
pdpmultiple.dictfile = 0;
pdpmultiple.optfile = 1;

solve pdpmultiple using MIP minimizing tcost;

*-----
* Solution Results
*-----

display xp.l, vseq.l, tcost.l;

parameter CostPerRoute(*);
CostPerRoute(RTS) = sum((i,j), s(i,j)*xp.l(i,j,RTS));

parameter ReqDistance(*);
ReqDistance(customer+n) = s(customer, customer+n);

parameter BetDepots(*,*);
BetDepots(depoti, depotj) = s(depoti, depotj);

display s, CostPerRoute, ReqDistance, BetDepots;
```

GENERAL ALGEBRAIC MODELING SYSTEM (GAMS)

Coding Template (Twelve Requests)

(1) SETS

\$ontext

Static One-to-One Multiple Vehicle Pickup and Delivery Problem with Multiple Depots

\$offtext

```
set v          vertices      /node1*node30/;
set i(v)       nodes (from)  /node1*node27/;
set j(v)       nodes (to)    /node4*node30/;
set depot1(i)  starting depots /node1*node3/;
set depotj(j)  ending depots  /node28*node30/;
set customer(v) pickup-delivery /node4*node27/;
set pickup(v)  pickup nodes   /node4*node15/;
set delivery(v) delivery nodes /node16*node27/;

set RTS        no. of routes  //;

scalar N       no. of requests //;
scalar CMB     max links      //;
```

(2) INSTANCE DATA (PARAMETERS)

```
table nodedata(v,*)
      xpoint      ypoint      pair
node1
node2
node3
node4
node5
node6
node7
node8
node9
node10
node11
node12
node13
node14
node15
node16
node17
node18
node19
node20
node21
node22
node23
node24
node25
node26
node27
node28
node29
node30
;
```

Node
Coordinates

```

parameter xc(v);
xc(v) = nodedata(v,'xpoint');

parameter yc(v);
yc(v) = nodedata(v,'ypoint');

parameter pair(v);
pair(v) = nodedata(v,'pair');

parameter s(*,*);
s(i,j) = SQRT(SQR(xc(i)-xc(j))+SQR(yc(i)-yc(j)));

```

(3) EQUATIONS

```

*-----*
* Problem Model
*-----*

OPTION RESLIM = 3600;
OPTION ITERLIM = 100000;
OPTION WORK = 1000;
OPTION OPTCR = 0.01;

variable tcost                objective variable: total cost of all routes
variable xp(*,*,RTS)         link-path selection
binary variable xp            if path 'i-j' is used
integer variable vseq(v)     for sequence of 'v' nodes

```

EQUATIONS

objective	minimizes total cost for all routes
sttdepot(RTS)	each route should have one starting depot
enddepot(RTS)	each route should have one ending depot
conone(RTS)	(1) route should end in the starting origin depot
contwo(RTS)	(2) route should end in the starting origin depot
conthree(RTS)	(3) route should end in the starting origin depot
combination(RTS)	forces the combination goal
frlimit(customer)	limits to only one outgoing link for customer nodes
tolimit(customer)	limits to only one incoming link for customer nodes
conservation(customer,RTS)	keeps the incoming and outgoing link in one route
onepair(RTS)	keeps the pickup-delivery nodes in one route
subtour(customer,j)	avoids sub-tour for the routes
precedence	makes sure that pickup comes before its corresponding delivery node
vtov	avoids vertex to vertex link
ditodj	avoids depoti to depotj link
djtodi	avoids depotj to depoti link
ditodel	avoids depoti to delivery link
picktodj	avoids pickup to depotj link
dtopick	avoids delivery to pickup link
;	

```

objective..
tcost =e= sum((i,j,RTS),xp(i,j,RTS)*s(i,j));

stdepot(RTS)..
enddepot(RTS)..
sum((depoti,pickup),xp(depoti,pickup,RTS)) =e= 1;
sum((delivery,depotj),xp(delivery,depotj,RTS)) =e= 1;

conone(RTS)..
contwo(RTS)..
conthree(RTS)..
sum((pickup),xp('node1',pickup,RTS))-sum((delivery),xp(delivery,'node16',RTS)) =e= 0;
sum((pickup),xp('node2',pickup,RTS))-sum((delivery),xp(delivery,'node17',RTS)) =e= 0;
sum((pickup),xp('node3',pickup,RTS))-sum((delivery),xp(delivery,'node18',RTS)) =e= 0;

combination(RTS)..
f1limit(customer)..
t1limit(customer)..
sum((i,j),xp(i,j,RTS)) =e= CMB;
sum((j,RTS),xp(customer,j,RTS)) =l= 1;
sum((i,RTS),xp(i,customer,RTS)) =l= 1;

conservation(customer,RTS)..
onepair(RTS)..
sum(j,xp(customer,j,RTS))-sum(i,xp(i,customer,RTS)) =e= 0;
sum((pickup,j),xp(pickup,j,RTS)*SQRT(pair(pickup)))-sum((delivery,j),xp(delivery,j,RTS)*SQRT(pair(delivery))) =e= 0;

subtour(customer,j)..
precedence(customer,customer+n)..
sum(RTS,vseq(customer)-vseq(j)+(2*N+1)*xp(customer,j,RTS)) =l= 2*N;
vseq(customer)-vseq(customer+n) =l= 0;

vtov(v,v,RTS) =e= 0;
xp(depoti,depotj,RTS) =e= 0;
xp(depotj,depoti,RTS) =e= 0;
xp(depoti,delivery,RTS) =e= 0;
xp(pickup,depotj,RTS) =e= 0;
xp(customer,customer-n,RTS) =e= 0;

```

(4) OPTIONS AND OUTPUT

```
model pdpmultiple
/objective, stdepot, enddepot, conone, contwo, conthree, combination, frlimit, tolimit, conservation, onepair, subtour, precedence,
vtov, ditodj, djtodi, ditodel, picktodj, dtopick/;

pdpmultiple.solprint = 2;
pdpmultiple.limrow = 0;
pdpmultiple.limcol = 0;
pdpmultiple.solveLink = 2;
pdpmultiple.dictfile = 0;
pdpmultiple.optfile = 1;

solve pdpmultiple using MIP minimizing tcost;

*-----
* Solution Results
*-----

display xp.l, vseq.l, tcost.l;

parameter CostPerRoute(*);
CostPerRoute(RIS) = sum((i,j), s(i,j)*xp.l(i,j,RIS));

parameter ReqDistance(*);
ReqDistance(customer+n) = s(customer, customer+n);

parameter BetDepots(*,*);
BetDepots(depoti, depotj) = s(depoti, depotj);

display s, CostPerRoute, ReqDistance, BetDepots;
```

VITA

Paolo Ian Lucero was born on April 7, 1992, in Silang, Cavite, Philippines. He finished his elementary education in Silang Central School and was able to study high school in Cavite Institute through a scholarship. In 2009, he received St. La Salle Scholarship and was able to pursue college in De La Salle University in Manila. In 2013, he earned his degree, "Bachelor of Science in Civil Engineering with Specialization in Transportation Engineering". Before receiving AUN/SEED-Net's Scholarship, he worked as a field engineer in a construction company in the Philippines. He went to Thailand in 2014 to pursue his master's degree at Chulalongkorn University also in the field of transportation engineering under civil engineering department. He also participated in an exchange program in Hokkaido University in Japan for one semester in 2015 to 2016. After earning his master's degree, he plans to go back to his university in the Philippines and teach before eventually pursuing another degree. Crazy for back-tucks, he also loves Pokemon, volleyball, and has three siblings.