

ระบบการจัดเส้นทางรถสำหรับการจัดส่งอาหารรายวันด้วยรถมอเตอร์ไซค์



นาย ธนัท สวนศิลป์พงศ์

สถาบันวิทยบริการ

จุฬาลงกรณ์มหาวิทยาลัย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

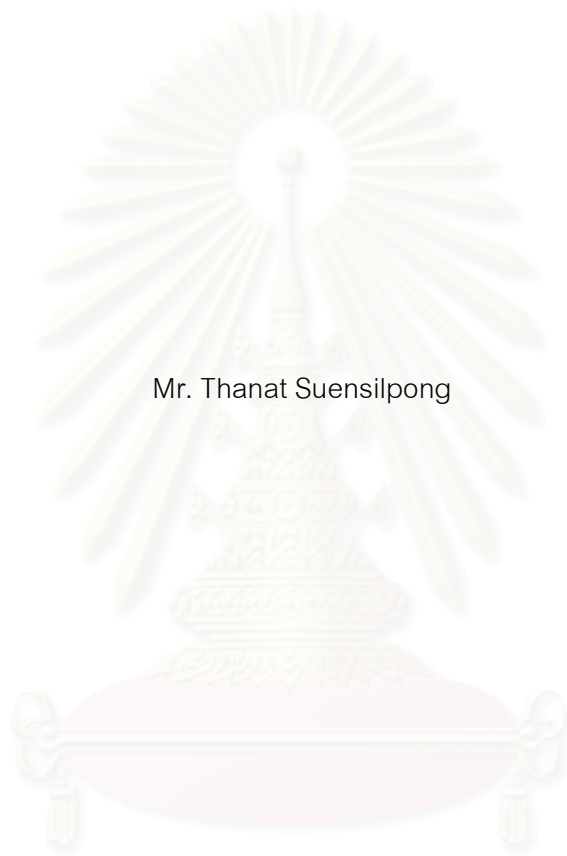
สาขาวิชาการจัดการทางวิศวกรรม ศูนย์ระดับภูมิภาคทางวิศวกรรมระบบการผลิต

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2550

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

VEHICLE ROUTING SYSTEM FOR DAILY MEAL DELIVERY WITH
MOTORCYCLES



Mr. Thanat Suensilpong

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

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

Regional Centre for Manufacturing Systems Engineering

Faculty of Engineering


Chulalongkorn University

Academic Year 2007

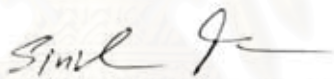
Copyright of Chulalongkorn University

| | |
|----------------|--|
| Thesis Title | VEHICLE ROUTING SYSTEM FOR DAILY MEAL DELIVERY WITH MOTORCYCLES |
| By | Mr. Thanat Suensilpong |
| Field of Study | Engineering Management |
| Thesis Advisor | Assistant Professor Manop Reodecha, Ph. 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 Boonsom Lerdhirunwong, Dr.Ing.)

THESIS COMMITTEE


..... Chairman
(Professor Sirichan Thongprasert, Ph.D.)


..... Thesis Advisor
(Assistant Professor Manop Reodecha, Ph. D.)


..... Member
(Assistant Professor Paveena Chaovalitwongse, Ph.D.)

จุฬาลงกรณ์มหาวิทยาลัย

ธนัท สวณศิลป์พงศ์ : ระบบการจัดเส้นทางรถสำหรับการจัดส่งอาหารรายวันด้วยรถมอเตอร์ไซด์. (VEHICLE ROUTING SYSTEM FOR DAILY MEAL DELIVERY WITH MOTORCYCLES) อ. ที่ปรึกษา : ผศ.ดร. มานพ เรียวเดชะ, 86 หน้า.

วิทยานิพนธ์นี้รายงานการพัฒนากระบวนการจัดเส้นทางรถของธุรกิจส่งอาหารเข้าประจำวัน ซึ่งวางแผนที่จะใช้รถมอเตอร์ไซด์รับจ้างในการจัดส่ง ในช่วงเวลาดังแต่ 5.00 น. ถึง 6.30 น. โดยมีการจ่ายเงินเป็นค่าจ้างตายตัวรายสัปดาห์ และค่าจ้างตามระยะทางจัดส่งที่บริษัทเป็นผู้กำหนดเส้นทาง ระบบการจัดเส้นทางรถที่พัฒนามีเป้าหมายให้ใช้จำนวนรถที่น้อยและระยะทางโดยรวมสั้น เพื่อให้ต้นทุนของการจัดส่งต่ำ ขณะที่ต้องคำนึงถึงข้อจำกัดทางด้านความจุและระยะเวลาที่ใช้เดินทางถึงลูกค้าของรถแต่ละคัน ขั้นตอนการพัฒนากระบวนการรวมถึงการหาวิธีแก้ปัญหา การสร้างแบบจำลองปัญหา การจัดทำโปรแกรมคอมพิวเตอร์ การจัดเตรียมข้อมูล การจัดทำคู่มือวิธีการใช้งานระบบ และการประเมินทดสอบระบบ

ระบบที่พัฒนาได้ประยุกต์ใช้ Savings algorithm กับ A* search algorithm ในระบบ โดยดัดแปลงจากโปรแกรมการจัดเส้นทางรถที่มีอยู่สำหรับปัญหาการจัดเส้นทางรถของรถบรรทุก จากการทดสอบกับปัญหาตัวอย่างซึ่งได้มีการเผยแพร่ไว้ พิสูจน์ได้ว่าระบบที่ได้พัฒนามีประสิทธิภาพที่ยอมรับได้ และสามารถจัดเส้นทางรถได้โดยใช้จำนวนรถไม่มากและระยะทางจัดส่งโดยรวมสั้น

ในการทดสอบกับข้อมูลที่สำรวจได้ว่าจะเป็นลูกค้าจำนวน 78 รายในย่านถนนพระรามเก้า ผลลัพธ์ที่ได้มีเส้นทางรถเดินทางโดยใช้รถ 4 คัน ซึ่งจัดส่งอาหาร 16, 29, 22, และ 24 กล่องซึ่งไม่เกินความสามารถบรรทุกของรถ และใช้เวลาถึงลูกค้าคนสุดท้าย 44 นาที, 1 ชม.19 นาที, 1 ชม., และ 56 นาทีตามลำดับซึ่งอยู่ในกรอบเวลาที่ต้องจัดส่ง นอกจากนั้น ยังใช้ระบบที่ได้พัฒนาในการทดลองเพื่อปรับปรุงผล ซึ่งทำให้เห็นถึงความเป็นไปได้ในการลดจำนวนรถที่ใช้ได้จาก 4 คันเป็น 3 คัน ด้วยการผ่อนปรนกรอบจำกัดเวลาในการเดินทาง หรือการลดเวลาที่ใช้ส่งของถึงมือลูกค้าแต่ละราย

ศูนย์ระดับภูมิภาคทางวิศวกรรมระบบการผลิต

Thanat Suanriping

สาขาวิชา...การจัดการทางวิศวกรรม.....

ลายมือชื่ออาจารย์ที่ปรึกษา.....

ปีการศึกษา 2550

4771602221 : MAJOR ENGINEERING MANAGEMENT

KEY WORD: VEHICLE ROUTING PROBLEM

THANAT SUENSILPONG : VEHICLE ROUTING SYSTEM FOR DAILY MEAL DELIVERY WITH MOTORCYCLES. THESIS ADVISOR : ASST. PROF. MANOP REODECHA, PH.D., 86 pp.

This thesis reports the development of a vehicle routing system for a daily breakfast delivery business. The business plans to hire motorcycles for the delivery operation which operates between 5:00 am to 6:30 am. The cost payable to each motorcycle employed consists of a fixed weekly fee and a charge that varies with the distance it travels in its delivery route specified by the company. The vehicle routing system aims to minimize the fleet size and the total travelling distance, which determine the cost of the delivery, with constraints on limited capacity and travelling time frame of each vehicle. The development process includes selection of the appropriate solution approaches, problem modelling, developing a computer program to solve the problem, data preparation, developing an operation procedure, and testing the system.

The developed system applied the Savings algorithm and the A* search algorithm. The program was modified from an existing vehicle routing program for truck routing. The modified program was tested with published instances of the problem to verify that the developed system is effective and gives routing with small fleet size and short total delivery distance.

In testing with survey data of 78 potential customers in the area of Rama IX Road, the system specified the use of four motorcycles. Each delivered 16, 29, 22 and 24 units respectively which did not exceed its capacity of 40 units. Each motorcycle delivered the meals to the last customer in 44 minutes, 1 hour 19 minutes, 1 hour, and 56 minutes respectively which are within the specified time frame. The system was also used to experiment for the possibility to improve results. It was found that it is possible to reduce the fleet size from four to three vehicles by relaxing the time frame or reducing the time spent at each drop off point.

Regional Centre for Manufacturing Systems Engineering Student's signature.....
Field of study...Engineering Management...Advisor's signature.....
Academic year 2007.....

Thanat Suensilpong
[Signature]

ACKNOWLEDGEMENTS

Firstly, the author wishes to thank Assistant Professor Manop Reodecha, thesis advisor, for his invaluable advice with warm encouragement. The author is also grateful to the members of thesis committees, Professor Sirichan Thongprasert and Assistant Professor Paveena Chaovalitwongse for their kind and helpful comments.

Furthermore, the author would like to thank Mr. Viroj Putvithee, members of Water Pacific Part., Ltd. and National Electronics and Computer Technology Center for supporting the vehicle routing software used in this thesis.

Finally, the author would like to thank to his families, friends, and teachers for their supports, understanding and encouragement throughout the study course.



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

CONTENTS

| | Page |
|--|------|
| THAI ABSTRACT | iv |
| ENGLISH ABSTRACT | v |
| ACKNOWLEDGEMENT | vi |
| CONTENTS | vii |
| LIST OF TABLES..... | x |
| LIST OF FIGURES..... | xi |
| | |
| CHAPTER I INTRODUCTION | 1 |
| 1.1 Background | 1 |
| 1.2 Thesis Objectives..... | 2 |
| 1.3 Scope of the Research and Assumptions..... | 2 |
| 1.4 Methodology..... | 4 |
| 1.5 Expected Benefit..... | 4 |
| | |
| CHAPTER II LITERATURE REVIEW..... | 5 |
| 2.1 Shortest Path Problem..... | 5 |
| 2.1.1 Solutions Techniques..... | 6 |
| 2.1.1.1 Single-source Algorithms..... | 6 |
| ● Dijkstra's Algorithm | 6 |
| ● Bellman-Ford Algorithm..... | 7 |
| ● A* search Algorithm | 7 |
| 2.1.1.2 All-pairs Algorithms..... | 9 |
| ● Floyd-Warshall Algorithm..... | 9 |
| ● Johnson's Algorithm..... | 13 |
| 2.2 Vehicle Routing Problem..... | 13 |
| 2.2.1 Solution Techniques..... | 15 |
| 2.2.1.1 Exact Approaches..... | 15 |
| ● Linear Programming | 15 |

| | Page |
|--|------|
| • Dynamic Programming..... | 16 |
| • Branch and Bound..... | 16 |
| 2.2.1.2 Heuristic Approaches..... | 16 |
| 2.2.1.2.1 Constructive Methods..... | 16 |
| • Nearest Neighbor Algorithm..... | 16 |
| • Nearest Insertion Algorithm..... | 17 |
| • Savings Algorithm..... | 17 |
| 2.2.1.2.2 Multi-route Improvement Heuristics | 19 |
| • Thompson and Psaraftis | 19 |
| • Van Breedam..... | 19 |
| • Kinderwater and Savelsbergh | 19 |
| 2.2.1.2.3 2-Phase Algorithms..... | 20 |
| • Cluster-First, Route-Second | 20 |
| • Route-First, Cluster-Second | 20 |
| 2.2.1.3 Metaheuristic Approaches..... | 20 |
| • Ant Colony Optimization (ACO)..... | 20 |
| • Genetic Algorithms..... | 20 |
| • Simulated Annealing..... | 21 |
| • Tabu Search..... | 22 |
| 2.3 Applications of Theories..... | 23 |
| 2.4 Conclusion..... | 25 |

CHAPTER III PROBLEM FORMULATION AND MODELLING, AND SOFTWARE

| | |
|------------------------------------|----|
| DEVELOPMENT | 26 |
| 3.1 Problem Description | 26 |
| 3.2 Problem Modelling Concept..... | 27 |
| 3.3 Model..... | 30 |
| 3.3.1 Inputs | 30 |

| | Page |
|---|------|
| • Customers' locations..... | 30 |
| • Customers' demands..... | 30 |
| 3.3.2 Computation | 30 |
| 3.3.3 Outputs..... | 32 |
| 3.4 Software Development | 32 |
| 3.5 System Validation..... | 38 |
| 3.6 Procedure of Operating the Vehicle Routing System | 43 |
| 3.7 Conclusion..... | 48 |
| CHAPTER IV SYSTEM EVALUATION..... | 49 |
| 4.1 Evaluation Procedure..... | 49 |
| 4.2 Test Input Data | 50 |
| 4.3 Results and Discussion | 53 |
| 4.3.1 Results..... | 53 |
| 4.3.2 Discussion | 55 |
| 4.4 Conclusion..... | 59 |
| CHAPTER V CONCLUSION AND RECOMMENDATIONS..... | 61 |
| 5.1 Conclusion..... | 61 |
| 5.2 Recommendations | 64 |
| REFERENCES | 66 |
| APPENDIX Route Details from Routing Program | 70 |
| BIOGRAPHY..... | 86 |

| | Page |
|---|------|
| Table 3.1: Results from running the developed system with A set of the instances with comparison..... | 40 |
| Table 3.2: Results from running the developed system with B set of the instances with comparison..... | 41 |
| Table 3.3: Results from running the developed system with P set of the instances with comparison..... | 42 |
| Table 4.1: Customer's data used for testing..... | 50 |
| Table 4.2: Routes summary of result from the test | 53 |
| Table 4.3: Routes summary of result from the time frame adjustment..... | 56 |
| Table 4.4: Routes summary of result from the first drop-off time adjustment..... | 58 |
| Table 4.5: Routes summary of result from the second drop-off time adjustment | 59 |
| Table 4.6: Traveling time to last customer, re-calculated from the test with drop-off time of 1 minute 45 seconds..... | 59 |
| Table A.1: Route details from a run of the routing program with initial conditions | 71 |
| Table A.2: Route details from a run of the routing program with relaxation of time frame or drop-off time of 1:30 minute | 76 |
| Table A.3: Route details from a run of the routing program with drop-off time of 1:45 minute..... | 81 |

LIST OF FIGURES

| | Page |
|--|------|
| Figure 1.1: Depot's location..... | 1 |
| Figure 2.1: A weighted network with negative arc | 5 |
| Figure 2.2: Dijkstra's algorithm | 7 |
| Figure 2.3: A* search algorithm | 7 |
| Figure 2.4: An example of shortest path problem | 8 |
| Figure 2.5: Concept of Floyd-Warshall algorithm..... | 9 |
| Figure 2.6: An example of shortest path problem | 11 |
| Figure 2.7: RCM procedure | 11 |
| Figure 2.8: RCM procedure (cont.)..... | 12 |
| Figure 2.9: An example of VRP | 14 |
| Figure 2.10: An example of VRP solution | 14 |
| Figure 2.11: Merging routes of Savings algorithm | 18 |
| Figure 2.12: Problematic case for Savings algorithm..... | 18 |
| Figure 2.13: Example of 3-cyclic, 2-transfer | 19 |
| Figure 2.14: Pseudo code of tabu search..... | 22 |
| Figure 3.1: Flow chart of finding solution for the vehicle routing system | 29 |
| Figure 3.2: Initial route for each customer..... | 31 |
| Figure 3.3: Calculate saving value of the existing software..... | 33 |
| Figure 3.4: Calculate saving value of the existing software for route with 2 customers | 34 |
| Figure 3.5: Case of having several customers in same edge | 35 |
| Figure 3.6: Problem of selecting start node and target node from direct distance | 36 |
| Figure 3.7: Problem of ignoring distance from road intersection to customers..... | 36 |
| Figure 3.8: Consideration of alternative node of edge and distance between road intersection and delivery customer for shortest path finding..... | 37 |
| Figure 3.9: Calculating distance between customers in same edge | 38 |
| Figure 3.10: Example of input file of problem instances..... | 39 |
| Figure 3.11: Road map and delivery point editor dialog..... | 44 |
| Figure 3.12: Customer information dialog for adding customer | 45 |

| | Page |
|--|------|
| Figure 3.13: Added customer icon..... | 45 |
| Figure 3.14: Delete Customer dialog | 46 |
| Figure 3.15: Vehicle routing dialog..... | 46 |
| Figure 3.16: Dialog for assigning parameters | 47 |
| Figure 3.17: Order information page..... | 47 |
| Figure 4.1: Distribution of Demand..... | 49 |
| Figure 4.2: Location of potential customers used for testing..... | 52 |
| Figure 4.3: Example of entering customer order | 54 |
| Figure 4.4: Running the system after entering the orders | 54 |
| Figure 4.5: Result of the software | 55 |
| Figure 4.6: Displayed the result in the Bangkok road map..... | 55 |
| Figure 4.7: Arrival time for each customer, generated by the program | 57 |

CHAPTER I

INTRODUCTION

Vehicle Routing Problem (VRP) is an important problem in operation management which has been studied both in theory and practice. Vehicle routing for delivering daily meal is a sub-classified problem which involves applying vehicle routing theories to the food delivery business.

1.1 Background

A company plans to start a breakfast delivery business in Bangkok. A critical success factor of the business besides the quality of the food is its delivery performance which must be punctual and efficient. It is necessary to have an effective vehicle routing system. Although vehicle can be routed by hand, the result sometimes is not good enough compared with the one from computerized system.

The company has set up a plan of using motorcycles for the delivery operation. The purpose of applying the motorcycles is that it is flexible and suitable to the congested condition of the traffic in Bangkok city. The company is able to hire any number of motorcycles with weekly agreements that include weekly fixed charge for the agreed working time and additional charge that is based on the distance of delivery route. Each morning, the hired motorcycles will come to the company's depot to fix containers of boxed meals on them for delivery on the route prescribed by the company.



Figure 1.1: Depot's location

The company's depot is located in the area near Rama IX Road and Srinakarintra Road as shown in Figure 1.1. The delivery covers an area of approximately 5 kilometre radius around the depot. The routing distance can be calculated from the map image of Google Map®. The time frame of the delivery operation is from 5:00 am to 6:30 am by which the last customer must receive his meal box. Therefore, motorcycles must reach to the depot within 4:45 am in order to fix the containers on the motorcycles. Each container can be loaded up to 40 meal boxes. The agreed payment includes weekly fixed charge of Baht 1,750 and Baht 1.10 per kilometre for the additional charge.

The factors that affect effectiveness of the vehicle routing system include: the appropriate size of the vehicle fleet, and the total travelling distance of all deliveries. The system also requires accurate data and procedure, with proper tools, to determine effective routing.

1.2 Thesis Objectives

The purpose of the thesis is to develop a vehicle routing system for the case company to minimize the fleet size and the total travel distance for the vehicles in order to deliver daily meals to all customers within the defined time frame. The delivery has to be done under the conditions that each vehicle must not deliver more than a specified number of points and must not take longer travelling time than the specified time frame.

1.3 Scope of the Research and Assumptions

The scope of this thesis includes:

1. Finding a method for problem solving.
2. Modelling the problem.
3. Implementation of the model and the solution in computerized simulation program.

4. Data preparation for the system.
5. Formulating the procedure for the company to operate the program.
6. Testing validity of the model and the solution with publicized benchmark.

The thesis has the following assumptions:

1. Delivery points may be changed weekly but not during the week.
2. Motorcycles are hired for one trip per day.
3. Every motorcycle has the maximum capacity of 40 orders.
4. The company is able to find the motorcycles with no limitation.
5. Expenditures for motorcycles include Baht 1,750 for weekly fixed charge and Baht 1.10 per kilometre for delivery charge related to the transportation distance which is calculated from the map image of Google Map®.
6. Irregularities, such as accidents and closing routes, are neglected from the computational model.
7. The routes exclude the expressway and the motorway due to traffic regulations.
8. Small roads have two-way traffic.
9. Main streets have dividing islands with one-way traffic for either side.

10. Since deliveries use motorcycles and delivery time is in the morning (5 am – 6.30 am), the conditions of traffic may be neglected from calculating travelling time. The average velocities of motorcycles are 25 kilometre per hour along small roads and 45 kilometre per hour along main streets.
11. Time spent with each customer is 2 minutes.

1.4 Methodology

Methodology for the thesis is described as following:

1. Study related materials regarding the vehicle routing problem, methods for solving the problem and select a method (CHAPTER II)
2. Define computational model for the shortest path algorithm and the selected method (CHAPTER III)
3. Design and implement simulation program and test validity (CHAPTER III)
4. Develop the program and procedures for the company (CHAPTER III)
5. Evaluate the program for validity of solutions (CHAPTER IV)

1.5 Expected Benefit

It is expected that the system will help the company to operate its business with low cost. Furthermore, it is hoped that the system will be beneficial to other businesses that have similar delivery/collection problems

CHAPTER II

LITERATURE REVIEW

This chapter describes theory related to the problem are being studied. It also describes some recent applications of the theories.

Vehicle routing system in this thesis is implemented with real road network. The problem consists of two important computational problems which are the shortest path problem and vehicle routing problem.

2.1 Shortest Path Problem

Shortest Path Problem, as described by Paul A. Jensen [8], Jesper Larsen and Jens Clausen [9], is a problem associated with network consisted of nodes and arcs. As shown in Figure 2.1, nodes, called as vertices in [9], in the network are connected by arcs, or edges [9]. The arcs are weighted and can be both directed and undirected. The weight of each arc can be either positive number or negative number. However, because roads have only positive length, networks which are used for transportation problems include only nonnegative arcs. The objective of the problem is to find the shortest path from a source node to a destination node.

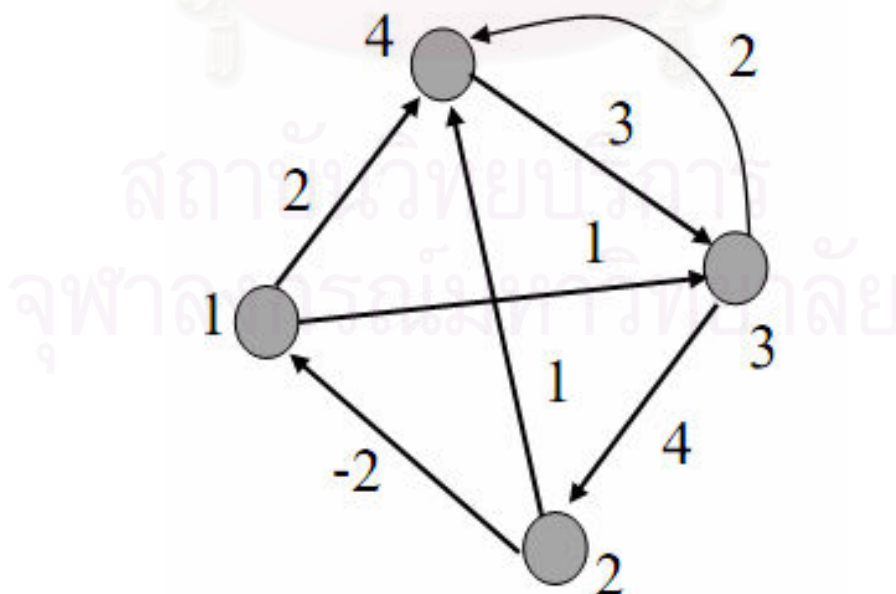


Figure 2.1: A weighted network with negative arc [10]

There are many researches that describe techniques of finding a solution for the problem. According to the reviewed literature, solution techniques for this problem are described in the following section.

2.1.1 Solution Techniques

There are many techniques used to find a solution for the shortest path problem. The techniques [11] are classified by type output into single-source algorithms and all-pairs algorithms.

2.1.1.1 Single-source algorithms

Single-source algorithms are used to find a solution from a source node to all other node the network [11]. The well-known algorithms for this category are as follows [11]:

- Dijkstra's Algorithm

The algorithm is named after its discoverer, Edsger Dijkstra [13]. It is a greedy algorithm which solves the single-source shortest path problem for a directed graph with non negative edge weights.

As described in [13], the algorithm works by keeping the cost $d[v]$ of each node v . These costs represent the shortest path between source node s and v . Procedures of the algorithm are described as following:

1. Set the cost for source node s to 0 and infinity for all other nodes.
2. Select a node which has lowest cost from the nodes that has not been scanned.
3. For each neighbor v of u , check to see whether the cost for node v , which includes node u in the path, is better than the current value. If this new path is better, update the cost with the new lower value.

According to [13] and [14], the original Dijkstra's algorithm uses an ordinary linked list or array to store un-scanned nodes and linear search with running time of $O(n^2)$, where n is the number of nodes, is required to find the lowest cost. For

more efficient implementation of the algorithm, different data structures are used such as binary heap, pairing heap, or Fibonacci heap. In addition, Dial (1969), cited in [14] and [15], is the first one who implement the Dijkstra's algorithm using the bucket data structure. As a result of his method, Dijkstra's algorithm is one of the fastest algorithms which solve the shortest path problem, as described in [15] and [16].

- Bellman-Ford Algorithm

The algorithm has an advantage over Dijkstra's algorithm that it can solve the problem with negative edge weights, according to [17]. It is described that the algorithm should be used only the problems which have negative edge weights because running time of this algorithm is higher than one of the Dijkstra's algorithm when solve the same problem. However, Bellman-Ford algorithm cannot solve the problems which contain a cycle of total negative weight.

Principle of Bellman-Ford algorithm is similar to Dijkstra's algorithm, but it simply scans all edges for $n-1$ times, where n is the number of nodes, instead of choosing the node with minimum cost to process. Therefore, the algorithm runs in $O(nxm)$, where n and m are the number of nodes and edges respectively.

- A* search Algorithm

A* search algorithm uses the greedy concept which is similar to the one used by Dijkstra's algorithm. However, according to [18] and [19], instead of

using only the cost from source node to node v , it also uses estimation of cost from the node v to the

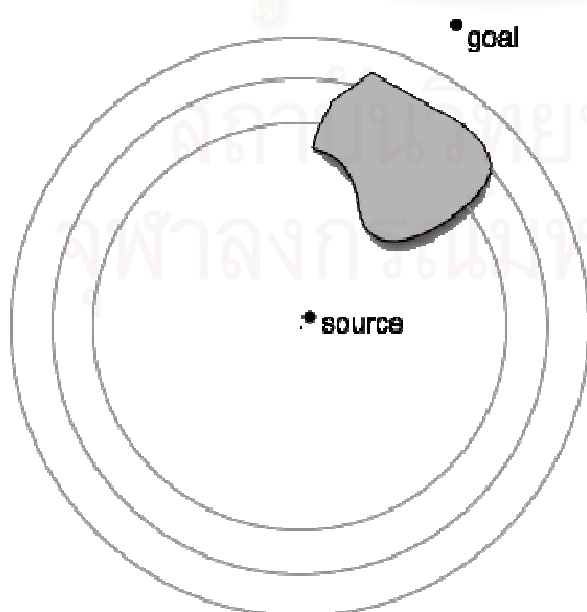


Figure 2.2: Dijkstra's algorithm [19]

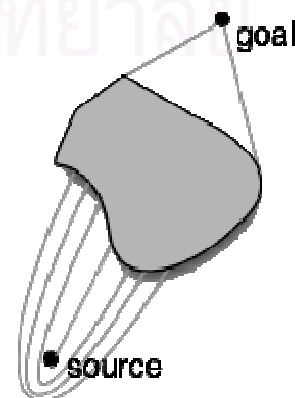


Figure 2.3: A* search algorithm [19]

destination node in the node selection. As described in [19], this algorithm can often do better than Dijkstra's Algorithm because Dijkstra does not take the destination location into account like A* search, as shown in Figure 2.2 and Figure 2.3.

The most important part of A* algorithm is estimating the cost to the destination. Heuristic function is commonly used for calculation of the value. Calculating the value depends on type of problem. Particularly, for the shortest path problem on road network, similar to the problem shown in Figure 2.4, the straight-line distance to the goal is used as the heuristic value, as described in [19] and [20].

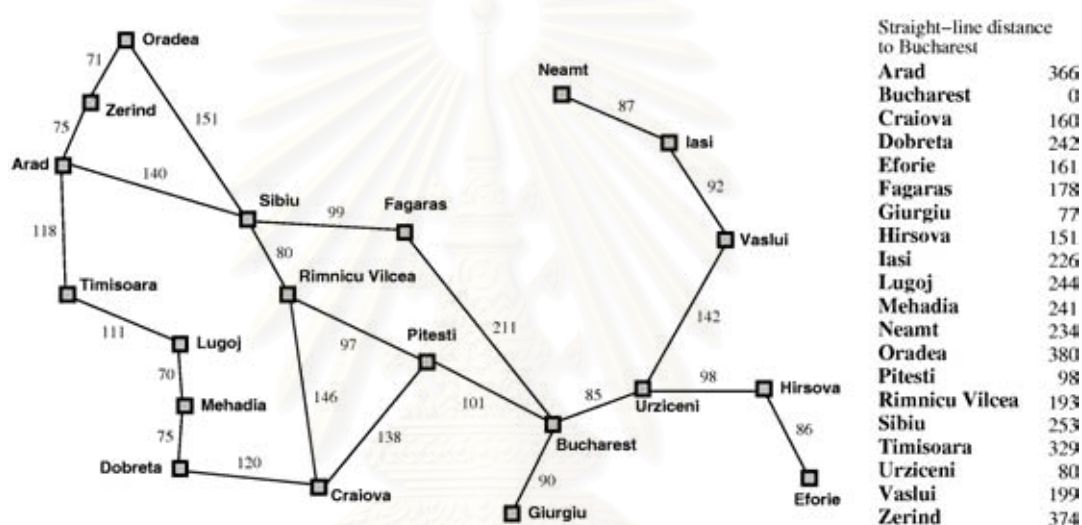


Figure 2.4: An example of shortest path problem [20]

Furthermore, it is described in [18] and [19] as follow:

- *A heuristic function is admissible if it never overestimates the distance to the goal vertex. For example, when the heuristic function is applied to the goal vertex itself, it must return zero. Admissibility ensures that when the goal vertex is first reached by the search, the path used is an optimum path. In the MapQuest example, Euclidean distance (distance as the crow flies) between the given vertex and the goal is admissible.*
- *A heuristic function is monotonic if the combined distance+heuristic from the initial vertex never decreases along any path. This is actually a stronger property than admissibility: if a function is monotonic, it must also be admissible.*

The time complexity of the algorithm depends on the calculation of heuristic [18] which can range from polynomial to exponential of the length of the solution. It is polynomial when the heuristic function h meets the condition, $|h(x) - h^*(x)| \leq O(\log h^*(x))$ where h^* is the optimal heuristic, i.e. the exact

cost to get from x to the goal. It is reinforced by Russell and Norvig's statement (2003), cited in [18], that the error of the heuristic function should not grow faster than the logarithm of the "perfect heuristic" that returns the true distance from a node to the destination.

In addition, its consumption of memory is enormous [18] which can be an exponential number of nodes. Therefore, there are many variants developed to cope with this problem such as iterative deepening A* (IDA), memory-bound A* (MA), and etc.

2.1.1.2 All-pairs Algorithms

Shortest path between every pair of nodes in the network is the output from all-pairs algorithms. Beside running single-source algorithm several times, the well-known algorithms for this category are as follow:

- Floyd-Warshall Algorithm

Floyd-Warshall algorithm is an example of dynamic programming [21]. The idea is that the shortest path between node i and node j has to pass intermediate nodes including nodes 1 to k , [10] and [21]. We have to consider whether to choose the node k in the path by comparing the distance between node i and node j , which intermediate nodes include nodes 1 to $k-1$, with the distance from node i to node k and distance from node k to node j which includes nodes 1 to $k-1$ in the intermediate nodes. Figure 2.5 illustrate the concept of Floyd-Warshall.

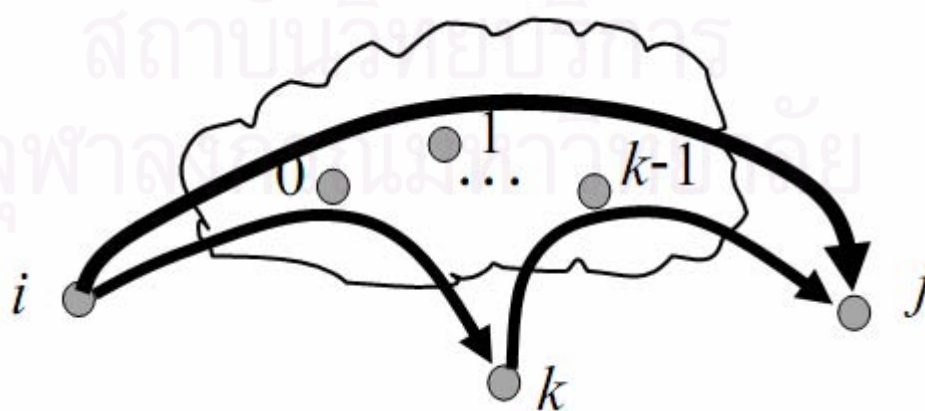


Figure 2.5: Concept of Floyd-Warshall algorithm [10]

Consider a function $shortestPath(i,j,k)$ as the shortest possible path from i to j using only vertices 1 through k as intermediate points along the way. The concept of Floyd-Warshall can be shown in following formula [21]:

$$shortestPath(i, j, k) = \min(shortestPath(i, j, k-1), shortestPath(i, k, k-1) + shortestPath(k, j, k-1));$$

$$shortestPath(i, j, 0) = edgeCost(i, j);$$

For the time complexity, this algorithm can solve the problem within $O(n^3)$ which is consider as a good algorithm for all-pairs shortest path problem [21].

In addition, the algorithm is called “Revised Cascade Method (RCM)” in [3] and [6]. The authors also describe how to find all the shortest paths by hands as following:

1. Assign numbers to each node from 1 to n
2. Let D^0 represent the metric of cost from node i to node j (d_{ij}) where $d_{ii} = 0$ and $d_{ij} = \infty$ if there is no edge from node i to node j . Let R^0 the metric of next node in shortest path and j is initially assigned for each value of r_{ij}^0 . The k value initially equals to 1.
3. In D^{k-1} , draw 2 lines vertically cross the column k and horizontally cross the row k . If there is a d_{ik} which equals to infinity in the vertical line, draw a horizon line cross the row i . Similarly, if there is an infinity in the horizontal line, a vertical line has to be drawn cross that infinity. For the rest of d_{ij}^{k-1} , the value of d_{ij}^k can be calculated by comparison between d_{ij}^{k-1} and $d_{ik}^{k-1} + d_{kj}^{k-1}$. Therefore, value of d_{ij}^k can be divided in 2 cases as following:

- If $d_{ik}^{k-1} + d_{kj}^{k-1}$ is lower than d_{ij}^{k-1} , let $d_{ij}^k = d_{ik}^{k-1} + d_{kj}^{k-1}$ and $r_{ij}^k = k$.
- If d_{ij}^{k-1} is lower than $d_{ik}^{k-1} + d_{kj}^{k-1}$, let $d_{ij}^k = d_{ij}^{k-1}$ and $r_{ij}^k = r_{ij}^{k-1}$.

- Increase the value of k by 1. If $k < n$, go back to step 3 until $k = n$.

D^n and R^n are the solution resulted from the method. Furthermore, example of shortest path problem is shown as below.

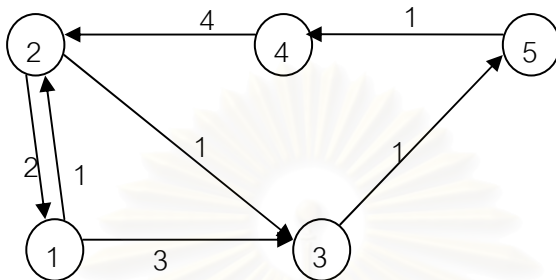


Figure 2.6: An example of shortest path problem (adapted from [3])

It can be transformed into metrics as following.

| | | | |
|---|--|---|---|
| $D^0 = \begin{matrix} & 1 & 2 & 3 & 4 & 5 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 0 & 1 & 3 & \alpha & \alpha \\ 2 & 0 & 1 & \alpha & \alpha \\ \alpha & \alpha & 0 & \alpha & 2 \\ \alpha & 4 & \alpha & 0 & \alpha \\ \alpha & \alpha & \alpha & 1 & 0 \end{bmatrix} \end{matrix}$ | $R^0 = \begin{matrix} & 1 & 2 & 3 & 4 & 5 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix} \end{matrix}$ | $m=3$ $D^2 = \begin{matrix} & 1 & 2 & 3 & 4 & 5 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 0 & 1 & 2 & \alpha & \alpha \\ 2 & 0 & 1 & \alpha & \alpha \\ \alpha & \alpha & 0 & \alpha & 2 \\ 6 & 4 & 5 & 0 & \alpha \\ \alpha & \alpha & \alpha & 1 & 0 \end{bmatrix} \end{matrix}$ | $d_{45}^3 = \min \{d_{45}^2, d_{43}^2 + d_{35}^2\}$ $= \min \{\infty, 2+2\} = 4$ $d_{15}^3 = 4, r_{15}^3 = 3$ $d_{25}^3 = \min \{d_{25}^2, d_{23}^2 + d_{35}^2\}$ $= \min \{\infty, 1+2\} = 3$ $d_{35}^3 = 3, r_{35}^3 = 3$ $d_{45}^3 = \min \{d_{45}^2, d_{43}^2 + d_{35}^2\}$ $= \min \{\infty, 5+2\} = 7$ $d_{45}^3 = 7, r_{45}^3 = 3$ |
| $m=1$ $D^0 = \begin{matrix} & 1 & 2 & 3 & 4 & 5 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 0 & 1 & 3 & \alpha & \alpha \\ 2 & 0 & 1 & \alpha & \alpha \\ \alpha & \alpha & 0 & \alpha & 2 \\ \alpha & 4 & \alpha & 0 & \alpha \\ \alpha & \alpha & \alpha & 1 & 0 \end{bmatrix} \end{matrix}$ | $d_{23}^1 = \min \{d_{23}^0, d_{21}^0 + d_{13}^0\}$ $= \min \{1, 2+3\} = d_{23}^0$ $r_{23}^1 = r_{23}^0 = 3$ $D^1 = D^0, R^1 = R^0$ | | |
| $m=2$ $D^1 = \begin{matrix} & 1 & 2 & 3 & 4 & 5 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 0 & 1 & 3 & \alpha & \alpha \\ 2 & 0 & 1 & \alpha & \alpha \\ \alpha & \alpha & 0 & \alpha & 2 \\ \alpha & 4 & \alpha & 0 & \alpha \\ \alpha & \alpha & \alpha & 1 & 0 \end{bmatrix} \end{matrix}$ | $d_{13}^2 = \min \{d_{13}^1, d_{12}^1 + d_{23}^1\}$ $= \min \{3, 1+1\} = 2$ $d_{13}^2 = d_{12}^1 + d_{23}^1 = 2$ $r_{13}^2 = 2$ $d_{41}^2 = \min \{d_{41}^1, d_{42}^1 + d_{21}^1\}$ $= \min \{\infty, 4+2\} = 6$ $d_{41}^2 = 6, r_{41}^2 = 2$ $d_{43}^2 = \min \{d_{43}^1, d_{42}^1 + d_{23}^1\}$ $= \min \{\infty, 4+1\} = 5$ $d_{43}^2 = 5, r_{43}^2 = 2$ | $m=3$ $D^3 = \begin{matrix} & 1 & 2 & 3 & 4 & 5 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 0 & 1 & 2 & \alpha & \alpha \\ 2 & 0 & 1 & \alpha & \alpha \\ 3 & \alpha & \alpha & 0 & \alpha & 2 \\ 6 & 4 & 5 & 0 & 7 \\ \alpha & \alpha & \alpha & 1 & 0 \end{bmatrix} \end{matrix}$ | $R^3 = \begin{matrix} & 1 & 2 & 3 & 4 & 5 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 1 & 2 & 2 & 4 & 3 \\ 1 & 2 & 3 & 4 & 3 \\ 1 & 2 & 3 & 4 & 5 \\ 2 & 2 & 2 & 4 & 3 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix} \end{matrix}$ |
| $D^2 = \begin{matrix} & 1 & 2 & 3 & 4 & 5 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 0 & 1 & 2 & \alpha & \alpha \\ 2 & 0 & 1 & \alpha & \alpha \\ \alpha & \alpha & 0 & \alpha & 2 \\ 6 & 4 & 5 & 0 & \alpha \\ \alpha & \alpha & \alpha & 1 & 0 \end{bmatrix} \end{matrix}$ | $R^2 = \begin{matrix} & 1 & 2 & 3 & 4 & 5 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 1 & 2 & 2 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \\ 2 & 2 & 2 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix} \end{matrix}$ | $m=4$ $D^3 = \begin{matrix} & 1 & 2 & 3 & 4 & 5 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 0 & 1 & 2 & \alpha & \alpha \\ 2 & 0 & 1 & \alpha & \alpha \\ 3 & \alpha & \alpha & 0 & \alpha & 2 \\ 6 & 4 & 5 & 0 & 7 \\ \alpha & \alpha & \alpha & 1 & 0 \end{bmatrix} \end{matrix}$ | $d_{51}^4 = \min \{d_{51}^3, d_{54}^3 + d_{41}^3\}$ $= \min \{\infty, 1+6\} = 7$ $d_{51}^4 = 7, r_{51}^4 = 4$ $d_{52}^4 = \min \{d_{52}^3, d_{54}^3 + d_{42}^3\}$ $= \min \{\infty, 1+4\} = 5$ $d_{52}^4 = 5, r_{52}^4 = 4$ $d_{53}^4 = \min \{d_{53}^3, d_{54}^3 + d_{43}^3\}$ $= \min \{\infty, 1+5\} = 6$ $d_{53}^4 = 6, r_{53}^4 = 4$ |

Figure 2.7: RCM procedure [3]

$$D^4 = \begin{matrix} & 1 & 2 & 3 & 4 & 5 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 0 & 1 & 2 & \alpha & 4 \\ 2 & 0 & 1 & \alpha & 3 \\ \alpha & \alpha & 0 & \alpha & 2 \\ 6 & 4 & 5 & 0 & 7 \\ 7 & 5 & 6 & 1 & 0 \end{bmatrix} \end{matrix}$$

$$R^4 = \begin{matrix} & 1 & 2 & 3 & 4 & 5 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 1 & 2 & 2 & 4 & 3 \\ 1 & 2 & 3 & 4 & 3 \\ 1 & 2 & 3 & 4 & 5 \\ 2 & 2 & 2 & 4 & 3 \\ 4 & 4 & 4 & 4 & 5 \end{bmatrix} \end{matrix}$$

$$m = 5 \quad D^4 = \begin{matrix} & 1 & 2 & 3 & 4 & 5 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 0 & 1 & 2 & \alpha & 4 \\ 2 & 0 & 1 & \alpha & 3 \\ \alpha & \alpha & 0 & \alpha & 2 \\ 6 & 4 & 5 & 0 & 7 \\ 7 & 5 & 6 & 1 & 0 \end{bmatrix} \end{matrix}$$

$$d_{12}^5 = \min \{d_{12}^4, d_{15}^4 + d_{52}^4\} \\ = \min \{1, 4+5\} = 1$$

$$d_{12}^5 = 1, r_{12}^5 = r_{12}^4 = 2$$

$$d_{13}^5 = \min \{d_{13}^4, d_{15}^4 + d_{53}^4\} \\ = \min \{2, 4+6\} = 2$$

$$d_{13}^5 = 2, r_{13}^5 = r_{13}^4 = 2$$

$$d_{14}^5 = \min \{d_{14}^4, d_{15}^4 + d_{54}^4\} \\ = \min \{\infty, 4+1\} = 5$$

$$d_{14}^5 = 5, r_{14}^5 = 5$$

$$d_{21}^5 = d_{21}^4, d_{23}^5 = d_{23}^4$$

$$d_{24}^5 = \min \{d_{24}^4, d_{25}^4 + d_{54}^4\} \\ = \min \{\infty, 3+1\} = 4$$

$$d_{24}^5 = 4, r_{24}^5 = 5$$

$$d_{31}^5 = \min \{d_{31}^4, d_{35}^4 + d_{51}^4\} \\ = \min \{\infty, 2+7\} = 9$$

$$d_{31}^5 = 9, r_{31}^5 = 5$$

$$d_{32}^5 = \min \{d_{32}^4, d_{35}^4 + d_{52}^4\} \\ = \min \{\infty, 2+5\} = 7$$

$$d_{32}^5 = 7, r_{32}^5 = 5$$

$$d_{34}^5 = \min \{d_{34}^4, d_{35}^4 + d_{54}^4\} \\ = \min \{\infty, 2+1\} = 3$$

$$d_{34}^5 = 3, r_{34}^5 = 5$$

$$d_{41}^5 = d_{41}^4; d_{42}^5 = d_{42}^4; d_{43}^5 = d_{43}^4$$

$$D^5 = \begin{matrix} & 1 & 2 & 3 & 4 & 5 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 0 & 1 & 2 & 5 & 4 \\ 2 & 0 & 1 & 4 & 3 \\ 9 & 7 & 0 & 3 & 2 \\ 6 & 4 & 5 & 0 & 7 \\ 7 & 5 & 6 & 1 & 0 \end{bmatrix} \end{matrix}$$

$$R^5 = \begin{matrix} & 1 & 2 & 3 & 4 & 5 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 1 & 2 & 2 & 5 & 3 \\ 1 & 2 & 3 & 5 & 3 \\ 3 & 3 & 3 & 5 & 5 \\ 2 & 2 & 2 & 4 & 3 \\ 4 & 4 & 4 & 4 & 5 \end{bmatrix} \end{matrix}$$

Figure 2.8: RCM procedure (cont.) [3]

As shown in Figure 2.8, D^5 and R^5 are the result from the method. D^5 tells the minimum cost between node i and node j , while R^5 tells nodes in that shortest path.

For example, shortest path from 3 to 2 costs 7. The path starts from $r_{32} = 5$, $r_{52} = 4$, $r_{42} = 2$. It is therefore 3, 5, 4, 2. Additionally, cost of path from 2 to 5 is 3. The path start from $r_{25} = 3$, $r_{35} = 5$, so it is 2, 3, 5.

- Johnson's Algorithm

Johnson's algorithm is an all-pairs shortest path algorithm. The algorithm is suitable for a sparse graph with some negative edge weights [22] because Bellman-Ford algorithm is a part of the algorithm. Its time complexity is $O(V^2 \log V + VE)$, where V and E is the number nodes and edges respectively. The procedures of its implementation [21] are as follows:

First, it adds a new node with zero weight edge from it to all other nodes, and runs the Bellman-Ford algorithm to check for negative weight cycles and find $h(v)$, the least weight of a path from the new node to node v . Next it reweights the edges using the nodes' $h(v)$ values. Finally for each node, it runs Dijkstra's algorithm and stores the computed least weight to other nodes, reweighted using the nodes' $h(v)$ values, as the final weight.

2.2 Vehicle Routing Problem

The Vehicle Routing Problem (VRP) [2] is a problem of determining a set of routes for a fleet of vehicle based at one of several depots for a number geographically dispersed cities or customers. The objective of the VRP is to deliver goods to a set of customers with minimum cost which depend on vehicle routes originating and terminating at a depot. An example of VRP problem and one of its possible outputs are shown in the figures below.

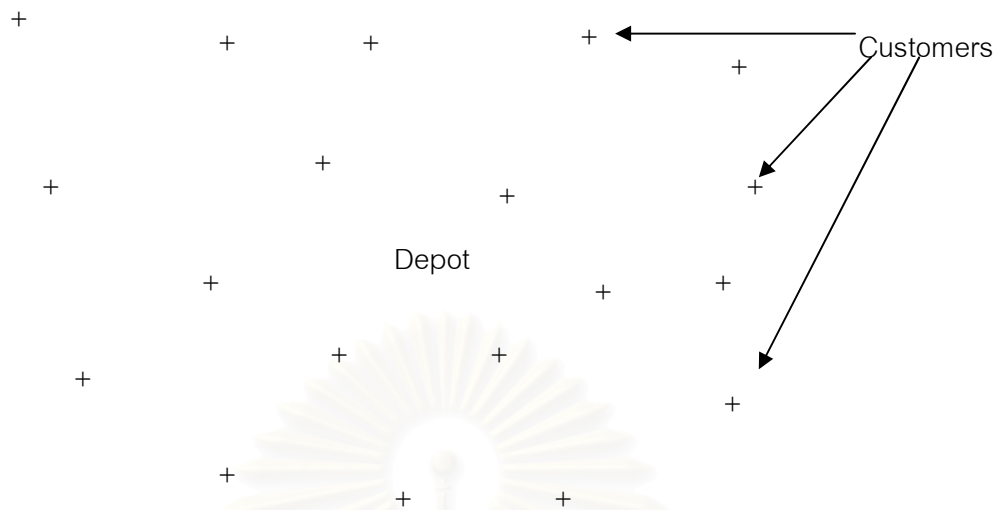


Figure 2.9: An example of VRP, adapted from [2]

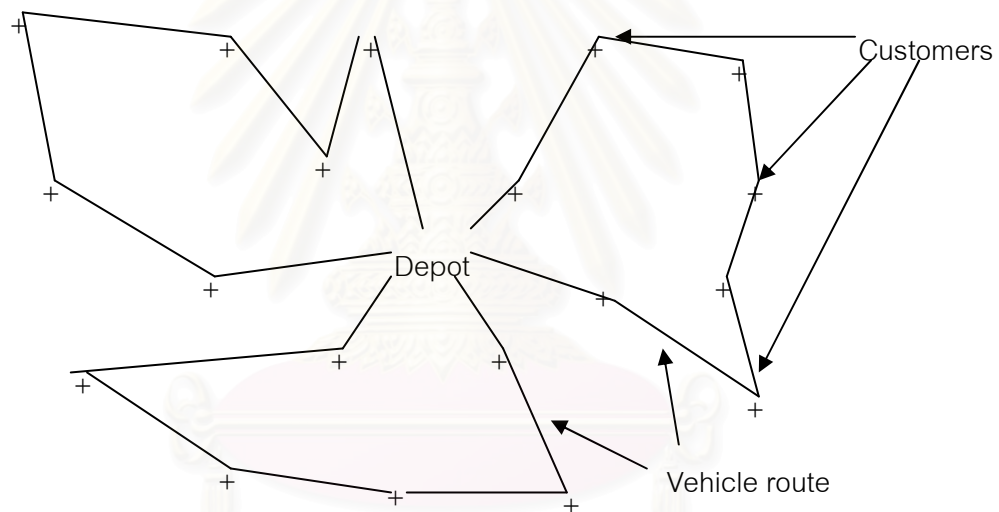


Figure 2.10: An example of VRP solution, adapted from [2]

According to [2], the VRP is a well known NP Hard integer programming problem which means that the computational time required for solving the problem increases exponentially with the problem size. In order to save computational time and get sufficient accuracy, approximate solutions are usually obtained.

VRPs in the real world have many side constraints such as capacity, delivery time, demands. These constraints derive VRP into many variants. Some of the most important restrictions [2] are:

- Every vehicle has a limited capacity (Capacitated VRP – CVRP)
- Every customer has to be supplied within a certain time window (VRP with time window – VRPTW)
- The vendor uses many depots to supply the customers (Multiple Depot VRP – MDVRP)
- Customers may return some goods to the depot (VRP with Pick-Up and Delivering – VRPPD)
- The customers may be served by different vehicles (Split Delivery VRP – SDVRP)
- Some value (like number of customers, their demands, serve time or travel time) are random (Stochastic VRP – SVRP)
- The deliveries may be done in some days (Periodic VRP – PVRP)

2.2.1 Solution Techniques

VRP is a computational problem which has been studied for over 40 years. Because the minimum solution cannot be guaranteed to be found within reasonable computing time [2], there is no further study on exact algorithms. Therefore, the techniques used to find a solution for VRP mostly are heuristics and metaheuristics [2]. The solution techniques can be classified as following:

2.2.1.1 Exact Approaches

The best solution is given by these approaches. However, the techniques consume too much time, so they can be used for VRP with certain number of customers. This depends on technique used to find the solution as following:

- Linear Programming

This technique has been used to solve many problems including VRP. As described in [1, 3, 4], the approach solves only small-sized VRP which can be

formulated into linear equation. The maximum problem size which can be solved by this approach is 42 cities. The technique which can solve the problem with size of 42 cities was proposed by Dantzig, Fulkerson and Johnson (1954, cited in [1]). However, this method can be used with only symmetric problems.

- Dynamic Programming

This technique was developed by Bellman Gonzales Zubieta (1962, cited in [1, 3]). As described by Pongpaut Totrakool [4], this technique finds a best solution by determining a solution for each node added to the routes. On the other hand, Held and Krap (1962, cited in [1, 3]) mentioned that this method is effective but it cannot solve the problem with size more than 13 customers due to limitation of computer memory and computation time.

- Branch and Bound

As described in Wikipedia [5], this method was first proposed by A. H. Land and A. G. Doig in 1960. According to Bernabé Dorransoro Diaz [2] and Wikipedia [5], the concept of this technique is to divide a problem into subproblems and then optimizes each subproblem individually. Krisakrai Manimmanakorn [6] described that this approach is suitable for problem with up to 25 cities. However, Fisher (1994, cited in [2]) proposed a K-tree method which succeeds in solving a problem with 71 cities.

2.2.1.2 Heuristic Approaches

The approaches were developed from the idea of saving computational time which is much lower than the exact approaches. Heuristic approaches are classified in 2 categories as follows:

2.2.1.2.1 Constructive Methods

Constructive methods [2] build a feasible solution gradually with computation of solution cost. However, the methods do not include the improvement phase. Algorithms in this category are described as following:

- Nearest Neighbor Algorithm

As described by Sean L. Forman [7] and Orawan Tunsitjareankun [1], this method is a greedy algorithm. It begins at the depot and then

selects the closet node to visit. It then travel from the last visited node to the closet node from the remaining unvisited node and repeat the procedure until the tour is completed.

Effectiveness of this algorithm can be shown as following:

$$\frac{\text{Length of nearest neighbor tour}}{\text{Length of optimal tour}} \leq \frac{1}{2} [\log_2 n] + \frac{1}{2}$$

- Nearest Insertion Algorithm

The method was first introduced by Rosenkrantz, Stearns and Lewis (1974), cited in [1] and [24]. The algorithm chooses a node closest to any node in the sub-tour to insert in the tour. Procedures of the method [1] [24] can be described as following:

1. Start a sub-graph with one node
2. Find node j with minimum cost from i to j and form sub-tour, $i-j-i$.
3. Choose node k which is closest to any node j in the sub-tour.
4. Find an edge (i,j) which minimizes $d_{ik}+d_{kj}-d_{ij}$ and insert k between i and j .
5. Repeat step 3 and 4 until all nodes are included in the tour.

The ratio of solution from this method to the optimal solution is as follow.

$$\frac{\text{Length of nearest insertion tour}}{\text{Length of optimal tour}} \leq 2$$

- Savings Algorithm

This method was introduced by Clarke and Wright (1964), as described in [1] [2] [3] [4] [6]. As described in [3] and [6], it is a simple algorithm which is easy to understand. Due to its character, it is one of the most known heuristic for VRP [2]. The idea of this method is merging two routes $(0, \dots, i, 0)$ and $(0, j, \dots, 0)$ into a single route $(0, \dots, i, j, \dots, 0)$ with distance saving $s_{ij} = c_{i0} + c_{0j} - c_{ij}$. Figure 2.11 shows the idea

of the method. According to the literature review, procedures of the method can be summarized as following:

1. Calculate $s_{ij} = c_{i0} + c_{0j} - c_{ij}$ for $i, j = 1, \dots, n, i \neq j$ and 0 refer to depot
2. Create n vehicle routes $(0, i, 0)$ for $i = 1, \dots, n$
3. Sort s_{ij} in descending order
4. Create sub-tour using point i and j with highest s_{ij}
5. Repeat step 4 until finish the tour

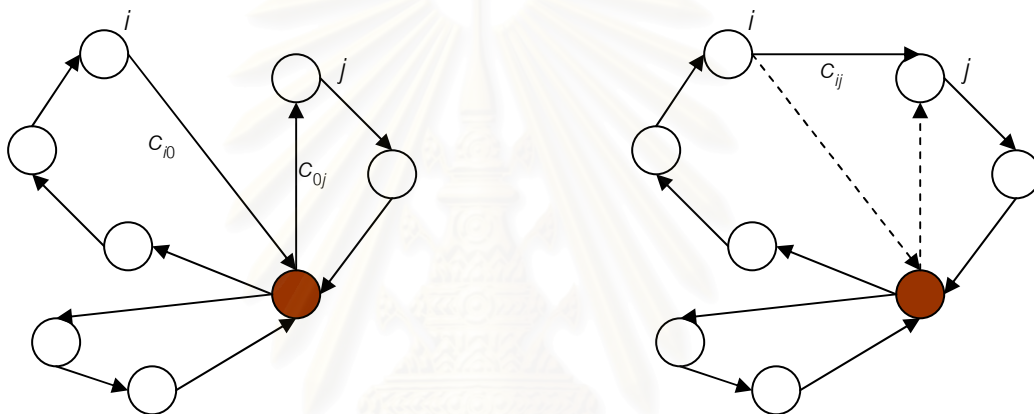


Figure 2.11: Merging routes of Savings algorithm

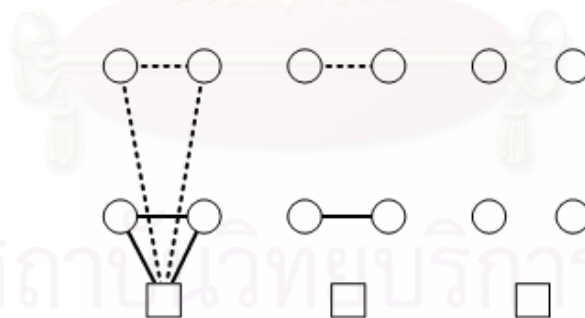


Figure 2.12: Problematic case for Savings algorithm [25]

In addition, there are some further developments of Savings algorithm which solve problems found in the traditional algorithm. For example, Matching Based Savings Algorithm, which was introduced by M. Desrochers and T. W. Verhoog, cited in [2] and [25], is designed according to the idea that the traditional one just combine two routes from point i to point j rather than rearrange the whole set in order to optimize overall cost. The case which has the problem is shown in Figure 2.12.

Another variant, which was introduced by K. Altinkemer, and B.Gavish (1991), cited in [2] and [25], was developed from the problem of un-merged routes of the savings algorithm.

2.2.1.2.2 Multi-route Improvement Heuristics

As described in [2], algorithms in this category improve a feasible solution by exchanging edges or node within or between vehicle routes. The algorithms are described in three researches as follows.

- Thompson and Psaraftis

The authors described the concept of " b -cyclic, k -transfer" [2]. The concept is that b routes are selected and k customers from each selected route are shifted to the next route of the cyclic permutation. The 3-cyclic 2-transfer operator is shown in the Figure 2.13.

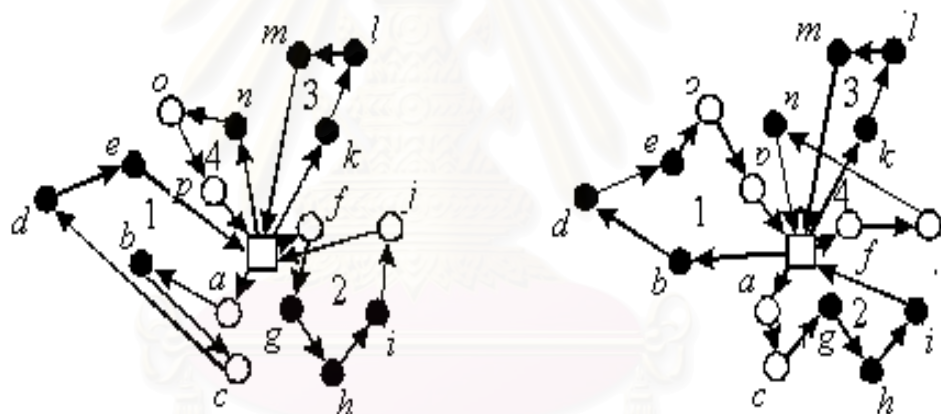


Figure 2.13: Example of 3-cyclic, 2-transfer [2]

- Van Breedam

Van Breedam classifies the improvement operations as "string cross", "string exchange", "string relocation", and "string mix", which can all be viewed as special cases of 2-cyclic exchanges, as described in [2].

- Kinderwater and Savelsbergh

The authors also described methods of improvement using exchanging nodes. Three main operations are customer relocation, crossover, and customer exchange [2]. Tours are not considered in isolation. Therefore, both edges and nodes are exchanged between different tours.

2.2.1.2.3 2-Phase Algorithms

2-Phase Algorithm is divided into 2 types as following:

- Cluster-First, Route-Second

These algorithms divide delivery points into groups then find a solution for each group. As described in [2], the methods include Fisher and Jaikumar's method, Sweep algorithm, Petal algorithm and Taillard's algorithm.

- Route-First, Cluster-Second

As described in [2], these methods construct a giant tour of TSP then divide it into vehicle routes in the second phase. Orawan Tunsitjareankun [1] describes the School Bus Routing Approach which is one of the Route-First, Cluster-Second methods.

2.2.1.3 Metaheuristic Approaches

Metaheuristic approaches, as described in [2], emphasizes on performing a deep exploration of the most promising regions of the solution space. It is also stated in [2] that quality of the solutions resulted from metaheuristics is much better than that obtained from classical heuristics. Commonly used metaheuristics for vehicle routing problem are described as following:

- Ant Colony Optimization (ACO)

Ant Colony Optimization (ACO), according to Marco Dorigo [26], is a class of optimization algorithms inspired from the behavior of real ant colonies. The first algorithm in this class was introduced by Marco Dorigo in 1992, and was called "Ant System" (AS). The algorithm was initially applied to the travelling salesman problem, and to the quadratic assignment problem. As described in [2], the algorithm has two phases: construction of vehicle routes and trail update. Additionally, more algorithms in ant colony optimization were developed [26] such as Ant Colony System (ACS), MAX-MIN Ant System (MMAS), and particularly hybrids version of ant colony optimization with local search.

- Genetic Algorithms

Genetic algorithm (GA) is one of well-known metaheuristic approaches. Particularly, as described in [2], it is likely to be the most widely known

type of the category. According to Bräysy (2001) [12], its basic concept are developed by Holland (1975) and practical application was demonstrated by De Jong (1975) and Goldberg (1989), as described in [2] and [12]. The core idea of the algorithm is about “survival of the fittest” which includes mechanics of natural selection and natural genetics used to evolve solutions to problems. In order to use the genetic algorithm, solution must be formed in string similar to chromosome. For example, a solution for vehicle routing problem can be as “7 → 8 → 9 → 11 → 12 → 2 → 3 → 1 → 4 → 5 → 6 → 10” where numbers represent delivery point.

According to the literature, [2], [12], [29] and [30], the operation of genetic algorithm is consisted of three operators: selection, reproduction and mutation. Firstly, individuals within a population are selected based on their fitness to reproduce new offspring for the new population which combines characteristics of their parents. According to Bräysy [12], combining characteristics potentially create a child with better fitness which ensures the survival of their descendants. In addition, mutation is performed with low probability [12]. With mutation, chromosomes are manipulate randomly. K.C. Tan et al. [30] stated that the main objective of mutation is to avoid becoming homogeneous of the population which leads to local optimum. Randomness of mutation process increases the variance of the population.

There are many researches in genetic algorithm due to the problem of time consumption the method, resulting in the creation of many variants of the method. The researches, described in [2], [29] and [30], focus on all aspects of the algorithm including representation of solution, selection process, reproduction process and mutation process.

- Simulated Annealing

As described in [2], Simulated Annealing (SA) is a stochastic relaxation technique originated in statistical mechanics. It is based on characteristic of annealing process of solids, where a solid is heated to a high temperature and gradually cooled in order for it to crystallize in a low energy configuration. The method uses a multi-routes improvement heuristics to find another solution. If the new solution is better than the existing one, it will use it with no condition. On the other hand, if the new one is

worse, it depends on a value T which is described in [2] as the temperature of the annealing. The probability of refusing the worse solution is $p = e^{-\Delta/T}$, where $\Delta = f(x) - f(x')$.

- Tabu Search

Similar to Simulated Annealing, Tabu Search also uses the improvement methods to find a solution, as described in [2], [27] and [28]. Hertz et al. [27] states that It was first presented in its present form in 1986 by Glover. As described in [2], [27] and [28], tabu search is one of well-known metaheuristics, and there are many further researches and developments of the algorithm. The concept of tabu search [28] is to help Local search, or neighborhood search, to escape from being trapped in local optimum by allowing non-improving moves. It additionally prevents cycling back to previously visited solution by using a list called “tabu list” which records the recent history of the search.

The tabu search algorithm associates with two terminologies: search space and neighborhood structure [28]. Search space is the space of all possible solutions. For the neighborhood structure, the neighborhood of current solution

Notation

- S , the current solution,
- S^* , the best-known solution,
- f^* , value of S^* ,
- $N(S)$, the neighborhood of S ,
- $\tilde{N}(S)$, the “admissible” subset of $N(S)$ (i.e., non-tabu or allowed by aspiration).

Initialization

Choose (construct) an initial solution S_0 .
Set $S := S_0$, $f^* := f(S_0)$, $S^* := S_0$, $T := \emptyset$.

Search

While *termination criterion not satisfied* do

- Select S in $\underset{S' \in \tilde{N}(S)}{\operatorname{argmin}} [f(S')]$;
- if $f(S) < f^*$, then set $f^* := f(S)$, $S^* := S$;
- record tabu for the current move in T (delete oldest entry if necessary);

endwhile.

Figure 2.14: Pseudo code of tabu search, from [28]

S is the set of solutions obtained by applying the local transformations to S . Denoted as $N(S)$, the neighborhood of S is a subset of the search space.

With basic concept of tabu search, pseudo code of the algorithm, described in [28], is shown in Figure 2.14. The author supposes that the problem is to minimize the function $f(S)$ over some domain.

Additionally, Heuristic approaches seem to be more appropriate due to low computational time with acceptable solution. Some results from the heuristic approaches are not good enough to be used practically, such as results from Nearest Neighbor Algorithm or Local Search (or Hill climbing) as described by Pongpaut Totrakool [4]. However, the approaches can be used in combination with other heuristic. For example, as cited in [12], Thangiah (1995) describes a method called GIDEON which used genetic algorithm to partition the customers in sectors, then routes each sector with the cheapest insertion method of Golden and Stewart (1985) and the routes are improved using λ -exchanges introduced by Osman (1993). F Potvin and Bengio (1996) use cheapest insertion heuristic of Solomon (1987) to create the initial population for genetic algorithm. Furthermore, Berger *et al.* (1998) use heuristic approaches in some parts of genetic algorithm, including generating the initial population and crossover operator. The saving algorithm of Clarke and Wright (1964) is also used to generate individuals of the initial population for genetic algorithm. In addition, Bräysy *et al.* (2000) also use several local search and route construction heuristics, which are inspired from the studies of Solomon (1987) and Taillard *et al.*, with genetic algorithm to solve the Vehicle Routing Problem with Time Windows (VRPTW).

2.3 Application of Theories

Beside theories of shortest path problem and vehicle routing problem, there are researches focus on applying those theories with case studies in many industries as follows.

Orawan Tunsitijareankun (1991) [1] used School Bus Routing Approach to find a solution for trucks to collect solid waste in Bang Khen area. She used branch and bound method to find a giant route in routing phase, then clustered it in to several

routes. She also used Dijkstra's algorithm to find the shortest path between each pair of collecting point. Additionally, operation procedures were proposed for the reasons of size of vehicle, employees' safety, costs, and time. For example, in case of collecting waste from one side of roads, route should be re-route in counter clockwise form for the truck to turn only left and avoid right turn which wastes the time and may obstruct the traffic.

Krisakrai Manimmanakorn (1995) [6] describes using Floyd-Warshall algorithm and Savings Algorithm with a case study of gasoline delivery. Fleet in the case study includes three types of vehicle. Constraint of the case study are prohibiting trucks to enter some roads in Bangkok for specific period of time and company's policy which customers can order only full filled truck. Author gives the reason of using Saving Algorithm that it is simple and uses less time than other methods while gives a solution which is similar to ones from other methods.

Naruporn Kanchanarat (1999) [3] develop a vehicle routing system for transporting furniture. The problem has three types of vehicle which are used to deliver goods in different distances. The goods in this research are different from two researches above due to variety of size and design. Furthermore, they are probably be damaged if they are not properly arranged. The algorithm for vehicle routing problem in this research is Savings algorithm with the reasons as same as stated by Krisakrai Manimmanakorn (1995) [6]. Floyd-Warshall method was also used in this research to find the shortest paths.

Pongpaut Totrakool (2004) [4] used Savings Algorithm, 2-OPT Algorithm and Anti-Intersection Algorithm to find a solution for the problem of medical supplies distribution system. Characteristics of the problem include uncertainty of demands, time windows for delivery, uncertainty of density of routes for different period of time, stability of operation system, and variety of products. The proposed approach was validated with sets of test data, which are described in his report. The results are shown in coordinated graphs.

Vipada Supavita and Duangpan Krichcharnchai (2006) [31] developed a prototype of transportation management system for a frozen food manufacturer. The vehicle routing is performed daily from late afternoon and it has to be finished within 9.00 am. Because the manual operations are not flexible enough for changes of orders, Savings Algorithm is chosen for the system. Data of distance between delivery point in the research is retrieved from digital map program and Mappoint Asia's website.

Viroj Putvithee, et al [32] also developed a vehicle routing system using Forward algorithm, a modification of A* search algorithm, to find the shortest path between each pair of delivery points. Savings algorithm is used to find a solution with farthest assigned customer and applying of saving value in the procedure.

2.4 Conclusion

Related theories and researches were described in this chapter. In next chapter, the problem formulation and modelling, and software development are described.



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

CHAPTER III

PROBLEM FORMULATION AND MODELLING, AND SOFTWARE DEVELOPMENT

This thesis aims to study vehicle routing problem of daily meals delivery by using motorcycles of the case company. Manually routing requires route familiarity which is uncertain in each driver. It consequently affects on costs and delivery time. Therefore, a mathematic modelling theory was applied in a vehicle routing software in order to optimize costs of delivery, and satisfy the time and capacity constraints. In this chapter, the concept of approaches which are used in the software and validation of the system are described.

3.1 Problem Description

The problem involves a plan for breakfast delivery service business in Bangkok. It hires a fleet of motorcycles to deliver meals to its customers from its depot located near Rama IX Road and Srinakarindra Road. The company can find the motorcycles with no limitation. The delivery area covers an area of approximately 5 km radius around the depot. Locations of customers can be retrieved from the survey of potential customers and are available on a map.

Expenditures for hiring each motorcycle include the weekly charge of Baht 1,750 per week and transportation charge which is agreed to Baht 1.10 per kilometre. Therefore, the company wishes to minimize its delivery cost by trying to use the least number of motorcycles and the least total travel distance by the motorcycles.

Limitations of the problem are capacity of each motorcycle and time which every vehicle has to reach its last customer. Each motorcycle will be attached with a container loaded with meal boxes during the delivery. Due to limitations of size of motorcycle and the container, each vehicle cannot be loaded more than 40 boxes. Additionally, the delivery process has limited time because the customers have to have their meals in the same specific time window. Therefore, time used to deliver all meal boxes should not longer than 1 hour 30 minutes with the average velocities of 25 km per

hour along small road and 45 km per hour along main streets, and drop-off time of 2 minutes.

3.2 Problem Modelling Concept

The objective of this problem is to operate the food delivery service by trying to find a routing which minimizes the number of vehicles employed for delivery and the total distance used to deliver the products. Savings algorithm and Forward algorithm, a modification of A* search algorithm, were chosen to find solutions for the problem.

Savings algorithm was chosen for the routing process of the problem because of its simplicity and widely usages, as described in Chapter 2. The procedure of the algorithm is described as following:

1. Initially build a set of routes, of which number is equal to the number of customers. Each route has only one customer.
2. Calculate Saving value (S_{ij}) between customer i and customer j , where $S_{ij} = c_{i0} + c_{0j} - c_{ij}$ and $i, j = 1, \dots, n$
3. Sort the saving value in descending order
4. Select the highest S_{ij} which customer i and j are not in the same route, and demand and travelling time of the merging of both routes are still satisfy capacity and time constraint
5. Merge the both routes. The merging of two routes is shown in Figure 2.11.
6. Repeat step 4 and 5 until every saving value is checked

Furthermore, Forward algorithm is used to find the shortest path between each pair of delivery points because it takes shorter time and the road network has no negative weighted edge, as mentioned before in Chapter 2, A* search algorithm.

Although all-pair shortest path algorithms are used in some researches as described by Krisakrai Manimmanakorn [6] and Naruporn Kanchanarat [3], the all-pair algorithms have too many unnecessary calculations compared to the Single-source algorithms, especially with road network which has up to 1000 unused nodes. The procedure of the algorithm implemented in the system can be described as following:

1. Add each edge, which is connected with the start node, in a path and put it in a "Priority Queue"
2. Calculate estimated cost of each path in the "Priority Queue" and sort all paths in ascending order. The estimated cost (C) of a path can be calculated as equation:

$$C = D_p + \sum d_{it}, \forall (i \text{ is a node in path } p)$$

where

C = the cost of path p

D_p = Total distance of path p

d_{it} = Linear distance from node i to target node

3. Remove the path, p , with lowest cost from the queue
4. If the last node of p is already visited, remove another path with lowest cost from the queue and check again.
5. If the last node of p is not the target node, make new paths by adding each edge, which is connected with the last node of p except the one included in p , in the path p and put them in the queue. Repeat the procedure from step 2 until the last node of p is the target node.

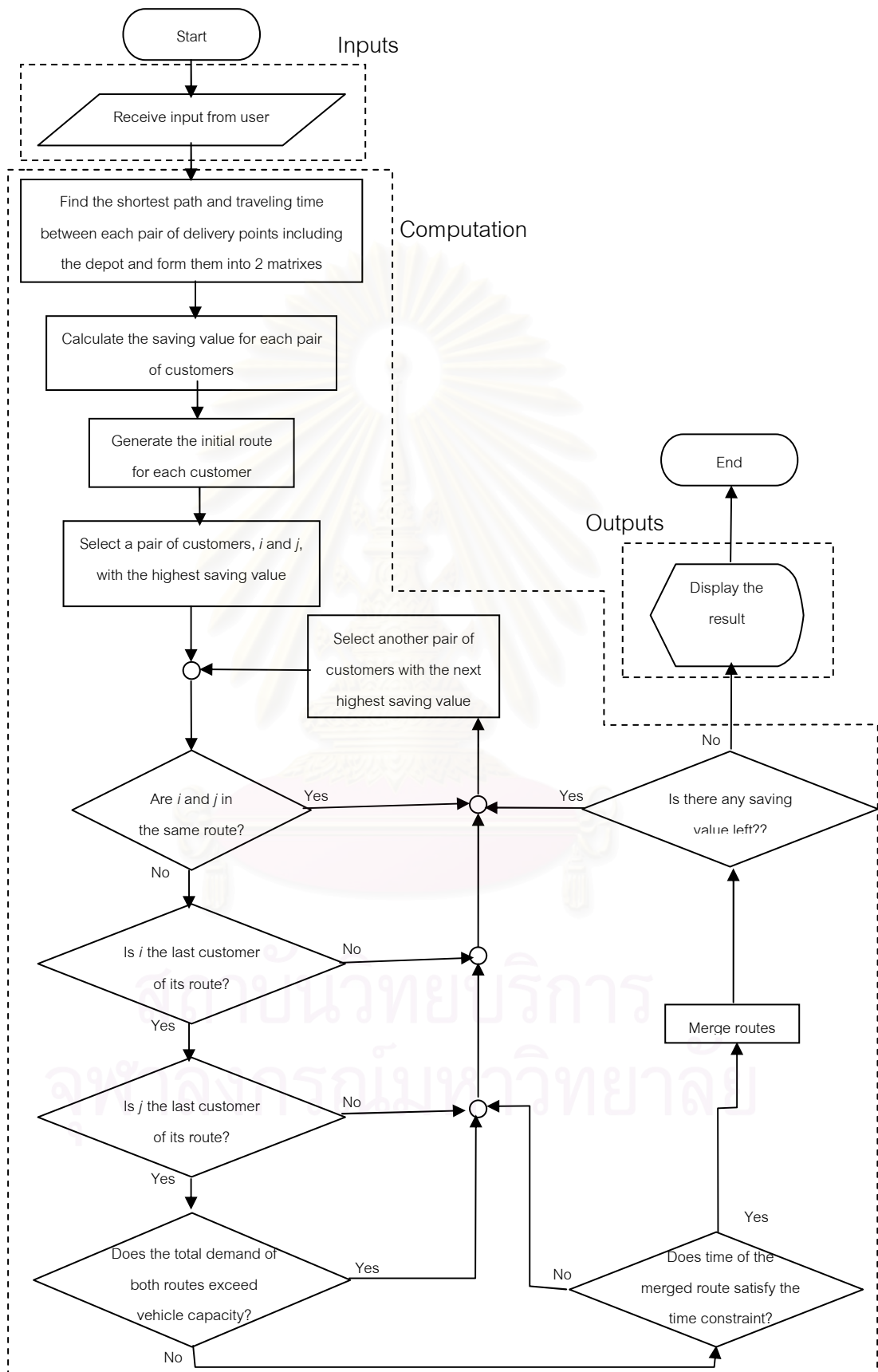


Figure 3.1: Flow chart of finding solution for the vehicle routing system

3.3 Model

Model of the vehicle routing system, of which flow chart is shown in Figure 3.1, is classified into 3 parts: inputs, computation, and outputs.

3.3.1 Inputs

Input of the system is used in the computational part to find a set of routes for the company to achieve the objectives. The input is consisted of:

- Customers' locations

Customers' locations are pointed in real road network. The company has to put the location of each customer to the system.

- Customers' demands

Demand of each customer is an important input for computational part to find a set of routes for motorcycles to deliver the boxes of meals. Although there are many recipes of food for the meals, they are packed into standard boxes loaded into the vehicles.

3.3.2 Computation

Computational part uses the input from the input part to find a solution of vehicle routes for the company to achieve the objectives. The parameters of the problem used in finding a solution for the company are vehicle capacity and time within which each motorcycle must reach its last customer.

Procedure of the computational part can be described as following:

1. Find the shortest path and time of each pair of delivery points including the depot and form them into 2 matrixes.
2. Calculate saving value of each pair of customers as shown in the equation, $S_{ij} = c_{i0} + c_{0j} - c_{ij}$ where

$$i \text{ and } j = \text{Customer number } (1, 2, \dots, n), i \neq j$$

S_{ij} = Saving value of (i, j)

c_{i0} = Distance from customer i to the depot

c_{0j} = Distance from the depot to customer j

c_{ij} = Distance from customer i to customer j

3. Make the initial route $(0, i, 0)$ for each customer as shown in Figure 3.2



Figure 3.2: Initial route for each customer

4. Select a pair of customers, i and j , which has the highest saving value.
5. If i and j are in the same route, select another pair of customers with the next highest saving value and check again
6. If i is not the last customer of its route or j is not the first customer of its route, select another pair of customers with the next highest saving value and go back to step 5
7. If total demand of both routes exceed the vehicle capacity, select another pair of customers with the next highest saving value and go back to step 5
8. Calculate time for the merging of both routes. The calculation can be shown as $T_m = T_1 + lt_i + T_2 - t_{0j} + t_{ij}$ where

T_m = Travelling time of merged route

T_1 = Travelling time of route which passes customer i

T_2 = Travelling time of route which passes customer j

lt_i = Drop-off time for customer i

t_{0j} = Travelling time from the depot to customer j

t_{ij} = Travelling time from customer i to customer j

9. If time of the merged route exceed the time constraint, choose another pair of customers with the next highest saving value and go back to step 5
10. Merge both routes by adding path from customer i to customer j , and deleting path from the depot to customer j and path from customer i to the depot, as shown in Figure 2.11
11. If there is remain saving value, select another pair of customer with the next highest saving value and repeat the procedure from step 5 to step 11 until every pair of customers is chosen

After the procedure is processed, a solution which includes a set of routes is ready to use. It will be sent to the next part, the output part.

3.3.3 Outputs

This part is responsible for displaying the result to user. The model finds the real path of each route and display the solution on the real road network with a list of customers in each route.

3.4 Software Development

Software for the vehicle routing system was developed by modifying a vehicle routing software which is developed by Viroj Putvithee et al [32] with a copyright of National Electronic and Computer Technology Center (NECTEC), member of National

Science and Technology Development Agency (NSTDA), and Water Pacific Part., Ltd. The software was developed with Java programming language with an advantage that each module of the software is separated in a file. It is used to find solutions for trucks routing. The fleet of trucks is allowed to have different types of vehicles. Finding solution procedure starts with using Savings algorithm to assign customers for each truck. The trucks are allowed to operate several rounds in order to serve all customers. The procedure of savings algorithm used in the program can be described as following:

1. Select the vehicle with highest capacity to form the first route
2. Select the customer which has longest direct distance from the depot to be the first customer added to the route
3. Calculate added distance for each of remain customers whose demands are able to be loaded in current truck. Added distances are calculated from only direct distance in every path of current route, as shown in Figure 3.3 and Figure 3.4.

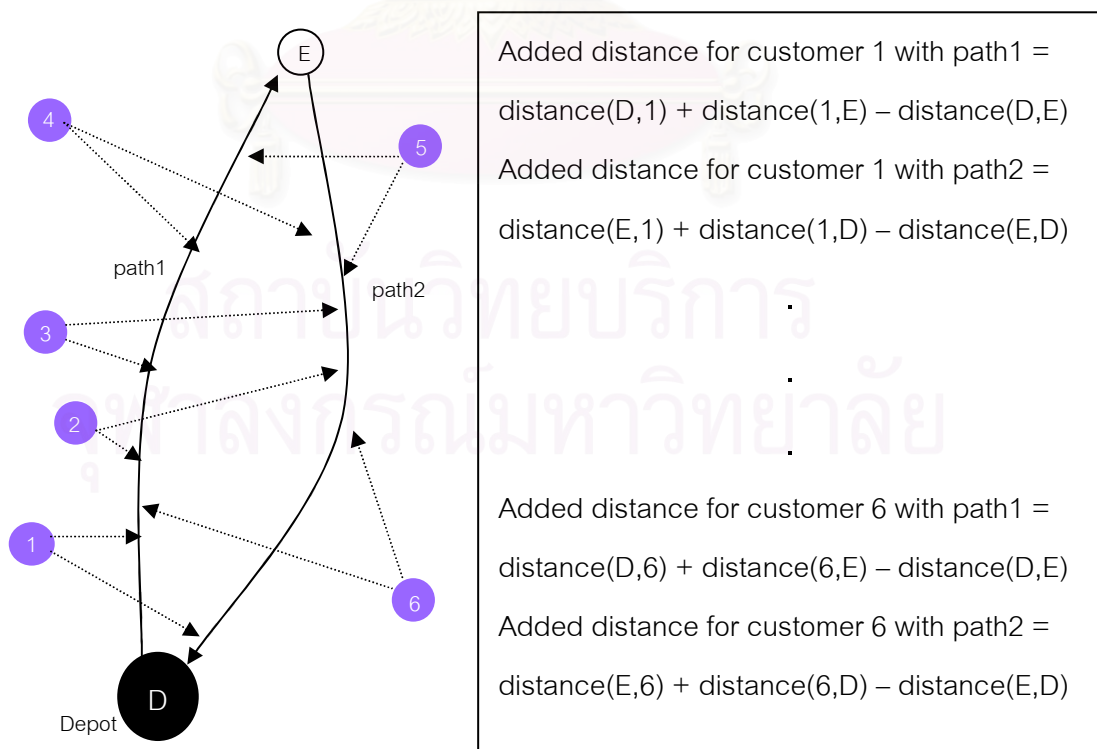
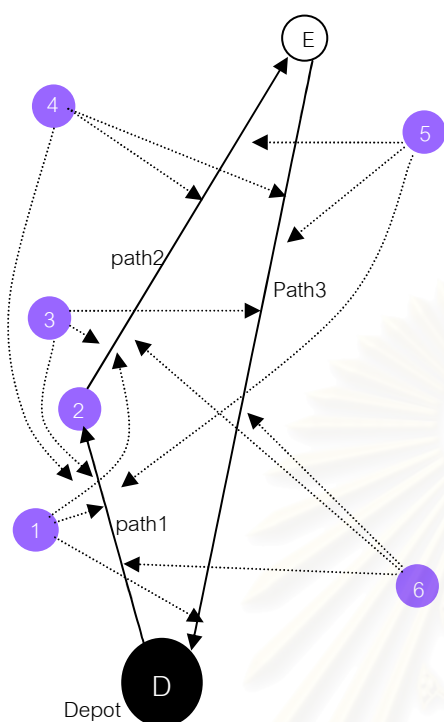


Figure 3.3: Calculate saving value of the existing software



Added distance for customer 1 with path1 =
 $\text{distance}(D,1) + \text{distance}(1,2) - \text{distance}(D,2)$

Added distance for customer 1 with path2 =
 $\text{distance}(2,1) + \text{distance}(1,E) - \text{distance}(2,E)$

Added distance for customer 1 with path3 =
 $\text{distance}(E,1) + \text{distance}(1,D) - \text{distance}(E,D)$

.

.

.

Added distance for customer 6 with path1 =
 $\text{distance}(D,6) + \text{distance}(6,2) - \text{distance}(D,2)$

Added distance for customer 6 with path2 =
 $\text{distance}(2,6) + \text{distance}(6,E) - \text{distance}(2,E)$

Added distance for customer 1 with path3 =
 $\text{distance}(E,6) + \text{distance}(6,D) - \text{distance}(E,D)$

Figure 3.4: Calculate saving value of the existing software for route with 2 customers

4. Add the customer with lowest added distance in the path which leads to the lowest distance of the route.
5. If every customer is assigned to a route, stop the process.
6. If there is remain capacity of the truck, repeat the procedure from step 3 until cannot fill any customer.
7. If there are trucks left, change to next highest capacity truck and repeat the process from step 2.
8. If there are remain customers but no truck left, reset all trucks for next round and repeat the process from step1.

After the routes are assigned with customers, the software uses Forward algorithm to find the shortest path between delivery points. The edges represent roads and the nodes represent road intersections. Procedure of the algorithm is described in 3.2 Problem Modelling Concept.

The software has many disadvantages for the meal delivery business. The software can solve the vehicle routing problem of trucks with capacity constraint, but without time constraint. The procedure, which finds the routes before shortest path calculation, is not practical for small scale problem like the delivery using motorcycles. Furthermore, because it was developed based on big scale problem, the shortest path was calculated without consideration of the case that there are several customers located in the same edge as explained in Figure 3.5. Additionally, as shown in Figure 3.6, the calculation uses only node with shorter direct distance to the target customer as the start node and node with shorter direct distance to the start customer as the end node. Consequently, the calculation also ignores distance from road intersection to delivery points. As a result, distances from the customers located in the same edge are not different, as shown in Figure 3.7.

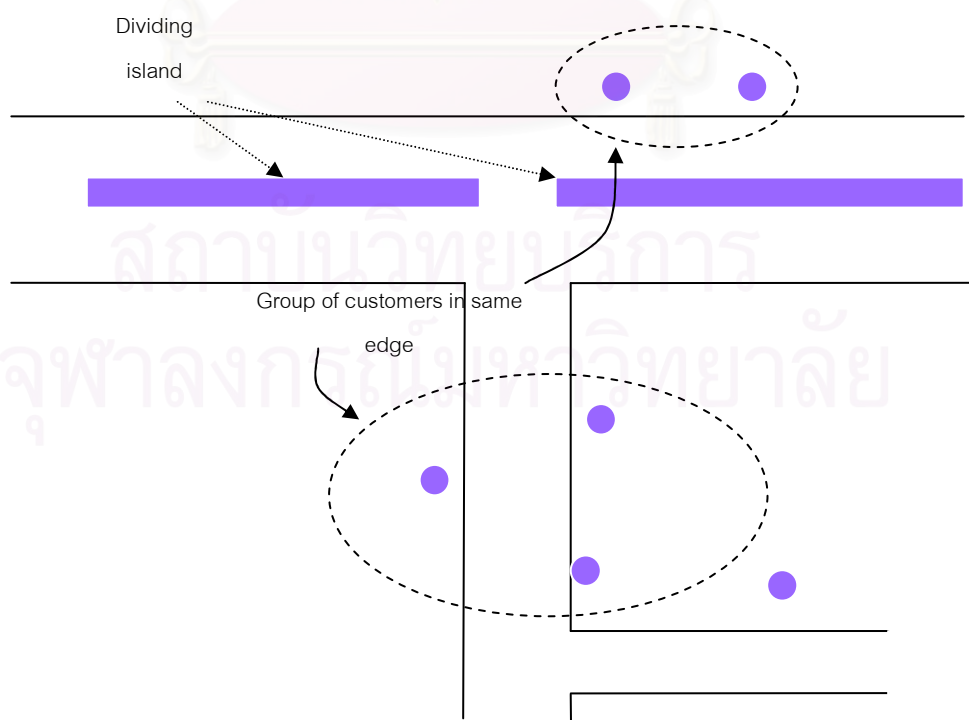


Figure 3.5: Case of having several customers in same edge

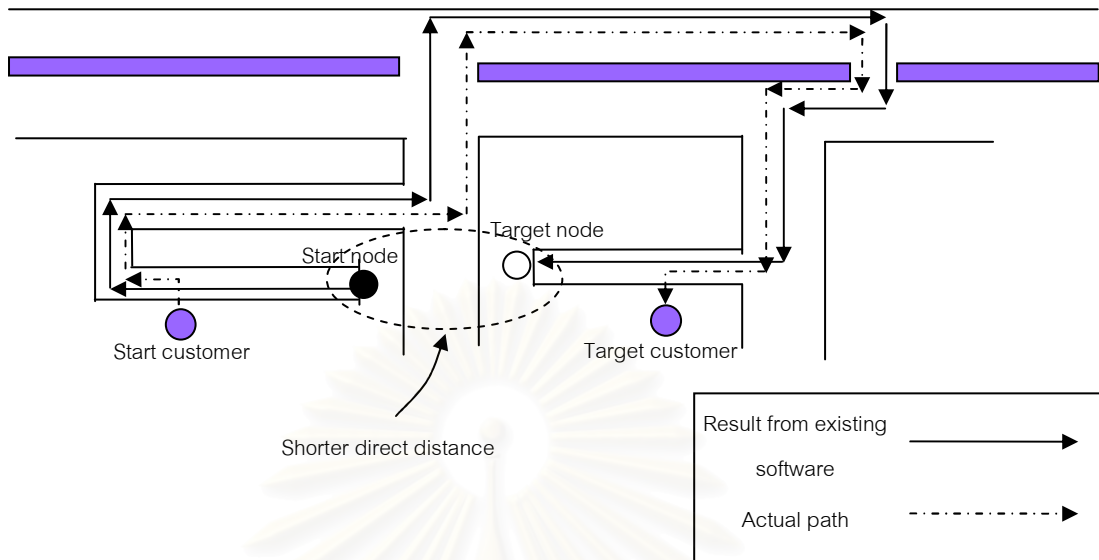


Figure 3.6: Problem of selecting start node and target node from direct distance

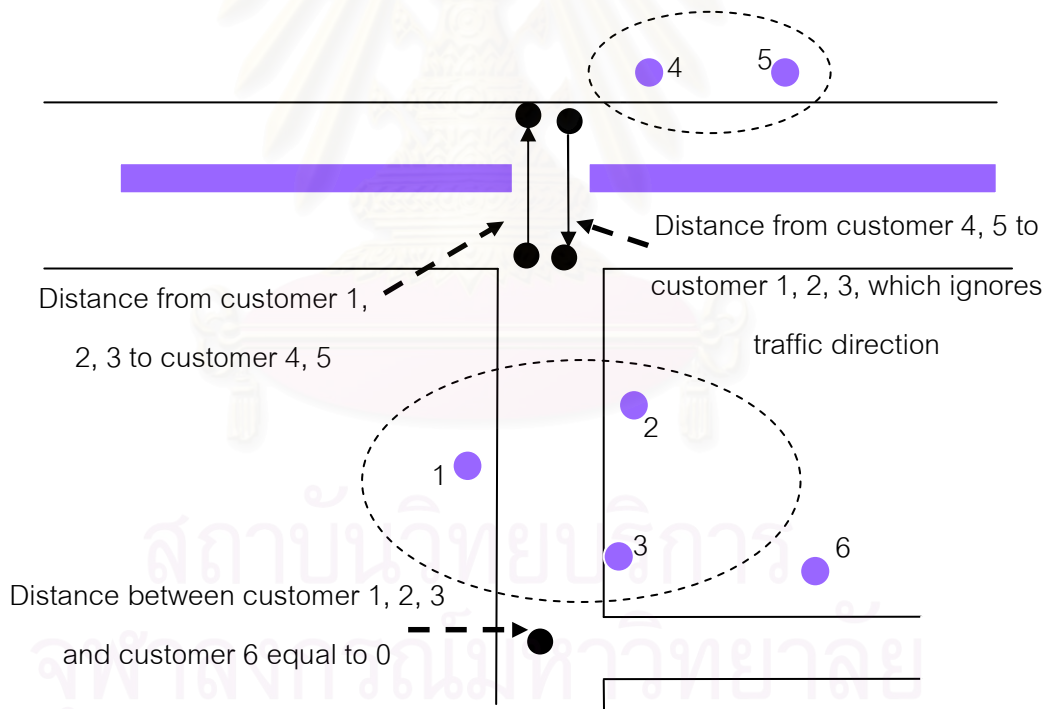


Figure 3.7: Problem of ignoring distance from road intersection to customers

The modifications of the software were made in 2 parts: shortest path finding control module and routing process. The shortest path finding was modified by adding the calculation of customers located in same edge, consideration of alternative

nodes of the edges which the delivery points are located, consideration of distance from road intersection in the shortest path finding control module, as explained in Figure 3.8 and Figure 3.9. The routing process was modified with a new developed module using savings algorithm of which the procedure is described in 3.3.2. The solution procedure was also modified as starting with shortest path finding between delivery points, and following with the routing process. The shortest path computation module, user interface and output display module of the software are not modified. The user interface uses Thai language for practicality.

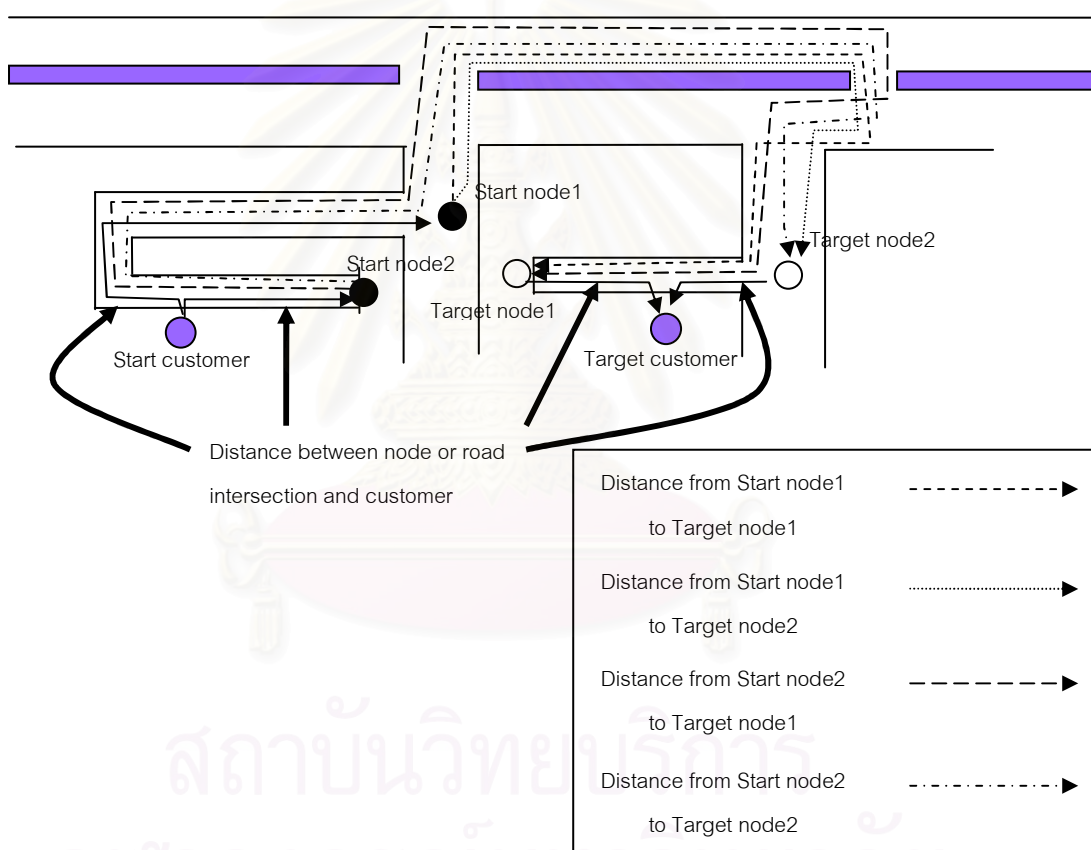


Figure 3.8: Consideration of alternative node of edge and distance between road intersection and delivery customer for shortest path finding

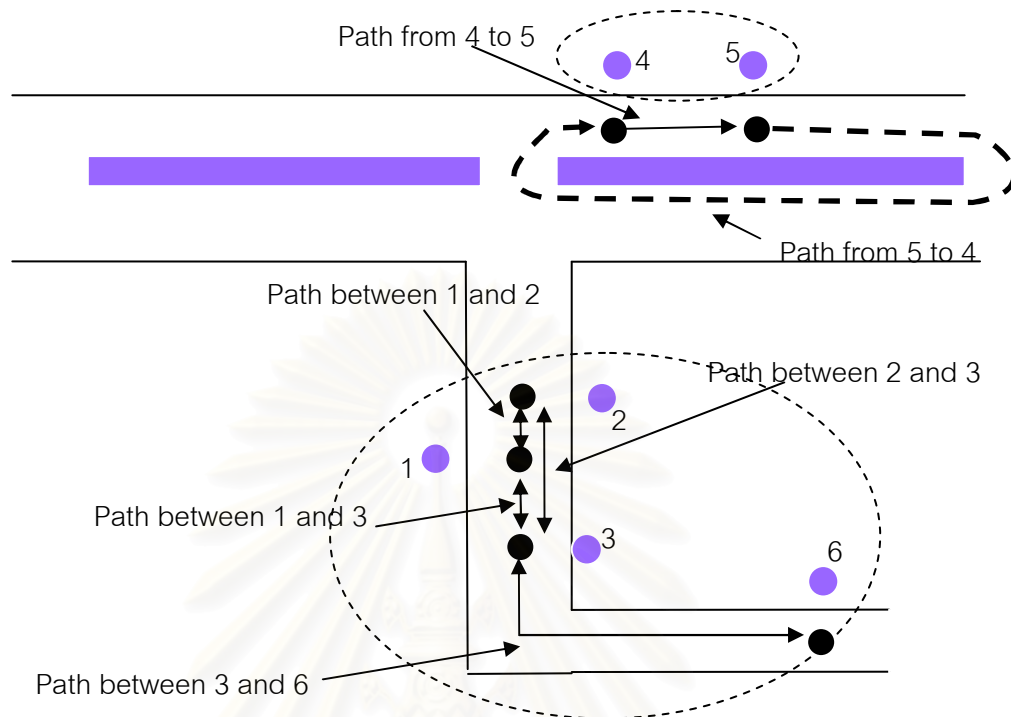


Figure 3.9: Calculating distance between customers in same edge

3.5 System Validation

Since the developed vehicle routing system has new software which uses the added savings algorithm, it needs proving that it can effectively find a solution for the problem. It was run with the instances of capacitated vehicle routing problem authored by Augerat, et al, which can be acquired from [2]. The example of input file of the instances is shown in Figure 3.10. The instances are divided into three sets: set A, set B, and set P. The results from the new software were compared with the known best results. The results and comparison of each set are shown on Table 3.1 to Table 3.3 respectively.

| | | |
|--|----------------|---------------|
| NAME : A-n32-k5 | 23 5 42 | 20 24 |
| COMMENT : (Augerat et al, Min no of trucks: 5, Optimal value: 784) | 24 42 9 | 21 8 |
| TYPE : CVRP | 25 61 62 | 22 12 |
| DIMENSION : 32 | 26 9 97 | 23 4 |
| EDGE_WEIGHT_TYPE : EUC_2D | 27 80 55 | 24 8 |
| CAPACITY : 100 | 28 57 69 | 25 24 |
| NODE_COORD_SECTION | 29 23 15 | 26 24 |
| 1 82 76 | 30 20 70 | 27 2 |
| 2 96 44 | 31 85 60 | 28 20 |
| 3 50 5 | 32 98 5 | 29 15 |
| 4 49 8 | DEMAND_SECTION | 30 2 |
| 5 13 7 | 1 0 | 31 14 |
| 6 29 89 | 2 19 | 32 9 |
| 7 58 30 | 3 21 | DEPOT_SECTION |
| 8 84 39 | 4 6 | 1 |
| 9 14 24 | 5 19 | -1 |
| 10 2 39 | 6 7 | EOF |
| 11 3 82 | 7 12 | |
| 12 5 10 | 8 16 | |
| 13 98 52 | 9 6 | |
| 14 84 25 | 10 16 | |
| 15 61 59 | 11 8 | |
| 16 1 65 | 12 14 | |
| 17 88 51 | 13 21 | |
| 18 91 2 | 14 16 | |
| 19 19 32 | 15 3 | |
| 20 93 3 | 16 22 | |
| 21 50 93 | 17 18 | |
| 22 98 14 | 18 19 | |
| | 19 1 | |

Figure 3.10: Example of input file of problem instances

Although some results from the developed system are very different from the known best results especially the results in P set, some results are better than the best results. The overall average of differences of distance is 6.17% which indicates that the performance of the developed system is close to the most known effective approach. Furthermore, the number of routes which is mostly as same as the best result shows the ability of the system to minimize the fleet size.

Table 3.1: Results from running the developed system with set A of the instances with comparison

| Problem Name | Results from the developed system | | The known best results | | Difference of distance (%) |
|--------------|-----------------------------------|---------------|------------------------|---------------|----------------------------|
| | distance | No. of routes | distance | No. of routes | |
| A-n32-k5 | 844 | 5 | 784 | 5 | 7.61% |
| A-n33-k5 | 694 | 5 | 661 | 5 | 4.95% |
| A-n33-k6 | 776 | 7 | 742 | 6 | 4.62% |
| A-n34-k5 | 807 | 5 | 778 | 5 | 3.77% |
| A-n36-k5 | 831 | 5 | 799 | 5 | 3.99% |
| A-n37-k5 | 705 | 5 | 669 | 5 | 5.41% |
| A-n37-k6 | 980 | 6 | 949 | 6 | 3.23% |
| A-n38-k5 | 795 | 6 | 730 | 5 | 8.86% |
| A-n39-k5 | 902 | 5 | 822 | 5 | 9.73% |
| A-n39-k6 | 883 | 6 | 831 | 6 | 6.21% |
| A-n44-k6 | 1021 | 6 | 937 | 6 | 8.95% |
| A-n45-k6 | 1013 | 7 | 944 | 6 | 7.30% |
| A-n45-k7 | 1200 | 7 | 1146 | 7 | 4.71% |
| A-n46-k7 | 940 | 7 | 914 | 7 | 2.82% |
| A-n48-k7 | 1115 | 7 | 1073 | 7 | 3.90% |
| A-n53-k7 | 1093 | 8 | 1010 | 7 | 8.21% |
| A-n54-k7 | 1202 | 7 | 1167 | 7 | 2.97% |
| A-n55-k9 | 1111 | 9 | 1073 | 9 | 3.50% |
| A-n60-k9 | 1377 | 9 | 1408 | 9 | -2.18% |
| A-n61-k9 | 1103 | 10 | 1035 | 9 | 6.58% |
| A-n62-k8 | 1353 | 8 | 1290 | 8 | 4.87% |
| A-n63-k10 | 1359 | 10 | 1315 | 10 | 3.34% |
| A-n63-k9 | 1691 | 10 | 1634 | 9 | 3.48% |
| A-n64-k9 | 1482 | 9 | 1402 | 9 | 5.71% |
| A-n65-k9 | 1243 | 10 | 1177 | 9 | 5.64% |
| A-n69-k9 | 1209 | 9 | 1168 | 9 | 3.55% |
| A-n80-k10 | 1837 | 10 | 1764 | 10 | 4.13% |
| Average | | | | | 5.03% |

| | |
|---|-----|
| Total routes | 198 |
| Difference of routes | 7 |
| Number of problems | 27 |
| Number of problems with higher number of routes | 7 |

| |
|-------|
| 191 |
| 3.66% |

=

Table 3.2: Results from running the developed system with set B of the instances with comparison

| Problem Name | Results from the developed system | | The known best results | | Difference of distance (%) |
|--------------|-----------------------------------|---------------|------------------------|---------------|----------------------------|
| | distance | No. of routes | distance | No. of routes | |
| B-n31-k5 | 685 | 5 | 672 | 5 | 1.88% |
| B-n34-k5 | 809 | 5 | 788 | 5 | 2.63% |
| B-n35-k5 | 980 | 5 | 955 | 5 | 2.66% |
| B-n38-k6 | 834 | 6 | 805 | 6 | 3.57% |
| B-n39-k5 | 569 | 5 | 549 | 5 | 3.61% |
| B-n41-k6 | 900 | 7 | 829 | 6 | 8.55% |
| B-n43-k6 | 763 | 6 | 742 | 6 | 2.78% |
| B-n44-k7 | 938 | 7 | 909 | 7 | 3.24% |
| B-n45-k5 | 757 | 5 | 751 | 5 | 0.82% |
| B-n45-k6 | 733 | 7 | 678 | 6 | 8.18% |
| B-n50-k7 | 750 | 7 | 741 | 7 | 1.16% |
| B-n50-k8 | 1354 | 8 | 1313 | 8 | 3.12% |
| B-n51-k7 | 1127 | 8 | 1032 | 7 | 9.17% |
| B-n52-k7 | 766 | 7 | 747 | 7 | 2.60% |
| B-n56-k7 | 737 | 7 | 707 | 7 | 4.19% |
| B-n57-k7 | 1240 | 8 | 1153 | 7 | 7.53% |
| B-n57-k9 | 1656 | 9 | 1598 | 9 | 3.65% |
| B-n63-k10 | 1598 | 10 | 1537 | 10 | 3.98% |
| B-n64-k9 | 922 | 10 | 861 | 9 | 7.08% |
| B-n66-k9 | 1425 | 10 | 1374 | 9 | 3.72% |
| B-n67-k10 | 1105 | 11 | 1033 | 10 | 6.96% |
| B-n68-k9 | 1320 | 9 | 1304 | 9 | 1.22% |
| B-n78-k10 | 1274 | 10 | 1266 | 10 | 0.63% |
| Average | | | | | 4.04% |

| | | | |
|---|-----|---|-------|
| Total routes | 172 | = | 165 |
| Difference of routes | 7 | | 4.24% |
| Number of problems | 23 | | |
| Number of problems with higher number of routes | 7 | | |

Table 3.3: Results from running the developed system with set P of the instances with comparison

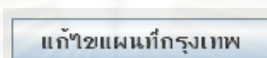
| Problem Name | Results from the developed system | | The known best results | | Difference of distance (%) |
|---|-----------------------------------|---------------|------------------------|--|----------------------------|
| | distance | No. of routes | distance | No. of routes | |
| P-n101-k4 | 772 | 4 | 681 | 4 | 13.36% |
| P-n16-k8 | 479 | 9 | 435 | 8 | 10.06% |
| P-n19-k2 | 238 | 2 | 212 | 2 | 12.21% |
| P-n20-k2 | 234 | 2 | 220 | 2 | 6.36% |
| P-n21-k2 | 236 | 2 | 211 | 2 | 11.94% |
| P-n22-k2 | 239 | 2 | 216 | 2 | 10.88% |
| P-n22-k8 | 591 | 9 | 603 | 8 | -2.05% |
| P-n23-k8 | 539 | 9 | 554 | 8 | -2.62% |
| P-n40-k5 | 518 | 5 | 458 | 5 | 13.18% |
| P-n45-k5 | 573 | 5 | 510 | 5 | 12.42% |
| P-n50-k10 | 734 | 11 | 696 | 10 | 5.51% |
| P-n50-k7 | 595 | 7 | 554 | 7 | 7.34% |
| P-n50-k8 | 667 | 7 | 649 | 8 | 2.75% |
| P-n51-k10 | 791 | 11 | 745 | 10 | 6.17% |
| P-n55-k10 | 732 | 10 | 669 | 10 | 9.45% |
| P-n55-k15 | 978 | 17 | 856 | 15 | 14.26% |
| P-n55-k7 | 617 | 7 | 524 | 7 | 17.79% |
| P-n55-k8 | 641 | 7 | 576 | 8 | 11.24% |
| P-n60-k10 | 789 | 10 | 706 | 10 | 11.72% |
| P-n60-k15 | 1024 | 16 | 905 | 15 | 13.15% |
| P-n65-k10 | 862 | 11 | 792 | 10 | 8.81% |
| P-n70-k10 | 896 | 11 | 834 | 10 | 7.45% |
| P-n76-k4 | 684 | 5 | 589 | 4 | 16.18% |
| P-n76-k5 | 698 | 5 | 631 | 5 | 10.54% |
| | | | | | 9.50% |
| Total routes | | 184 | | 175 | |
| Difference of routes | | 9 | = | 5.14% | |
| Number of problems | | 24 | | | |
| Number of problems with higher number of routes | | 10 | | Number of problems with lower number of routes | 2 |

3.6 Procedure of Operating the Vehicle Routing System


After the vehicle routing system for the case company is developed and tested, procedure for using the system is formulated. The procedure is divided into 2 parts: Adding/Deleting customers and Vehicle Routing which can be described as following:

1. Adding/Deleting customers

1 At the main dialog of the system click at button



to enter to the road map and delivery point editor dialog and scroll to customer location as shown in Figure 3.11

2 Click button  at the tools bar to enter adding/deleting customer mode

3 Left click at the road which the customer is located and the system will ask the user to enter customer information of which loading time and Zone are 0.0334 hour and "1" respectively, as shown in Figure 3.12.

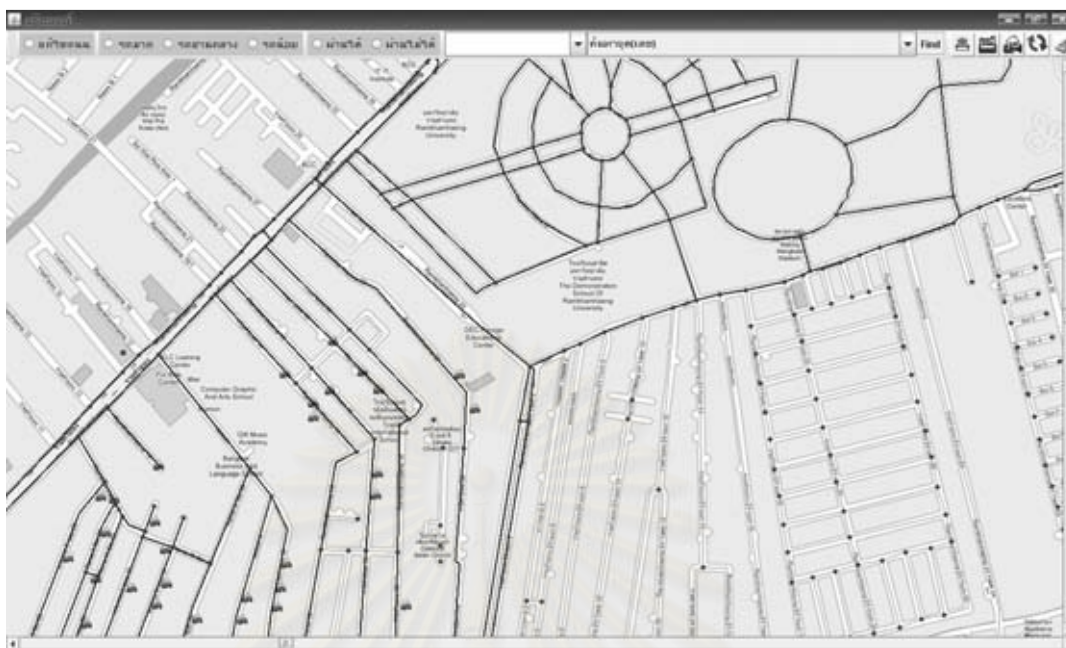
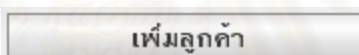


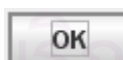
Figure 3.11: Road map and delivery point editor dialog

4 After finish enter the information, click button



to save the information and a customer icon will be added into road map, as shown in Figure 3.13. If there is remain customer to add, repeat step 3 and 4.

5 To delete a customer, right click at icon of customer which user want to delete and confirm deleting by



clicking button at the “Delete Customer” dialog, as shown in Figure 3.14.

สถาบันวิจัยและพัฒนา
จุฬาลงกรณ์มหาวิทยาลัย

เพิ่มลูกค้าใหม่

รหัส
รหัสลูกค้า [1234]

รายการลูกค้า

รายละเอียด

ชื่อถนน ช.รามคำแหง22

ชื่อลูกค้า [Test customer]

เวลายกสินค้าลง 0.0334 ชั่วโมง

เวลาส่งสินค้า 00.00 - 23.59

Zone & Note

Zone 1

Note

Priority h or n N

เพิ่มลูกค้า ยกเลิก

Figure 3.12: Customer information dialog for adding customer

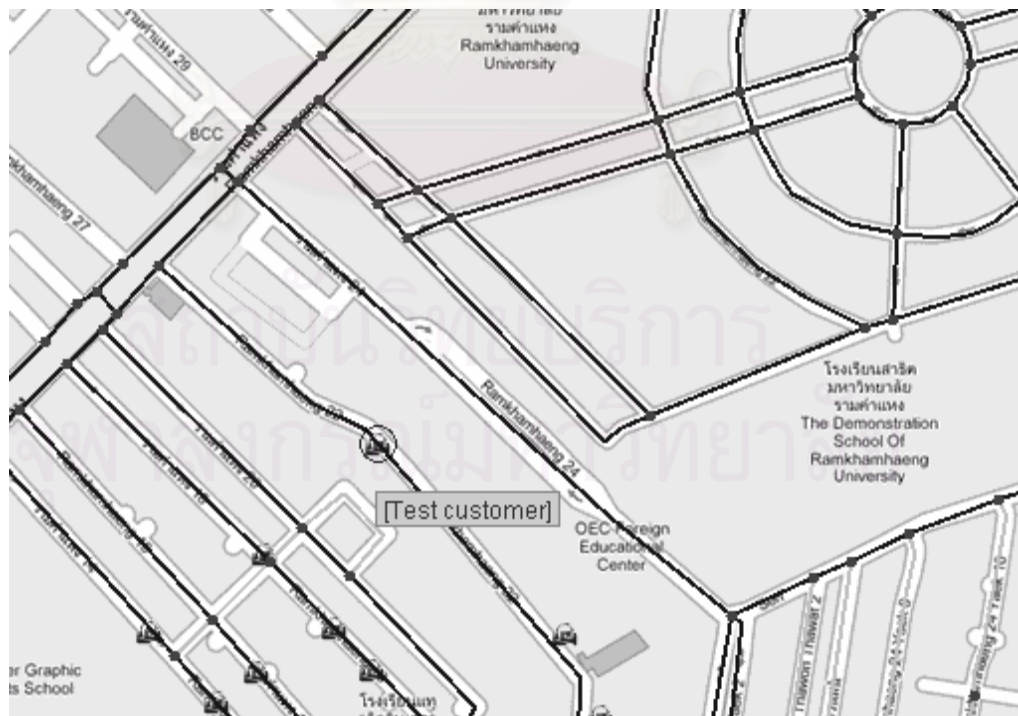


Figure 3.13: Added customer icon

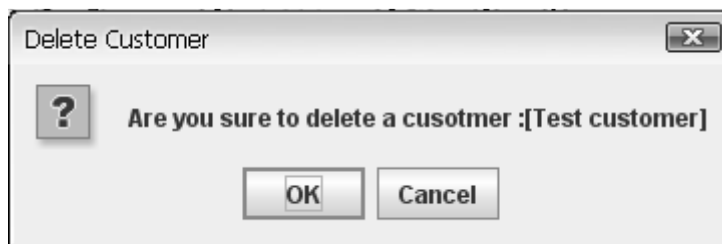


Figure 3.14: Delete Customer dialog

2. Vehicle Routing

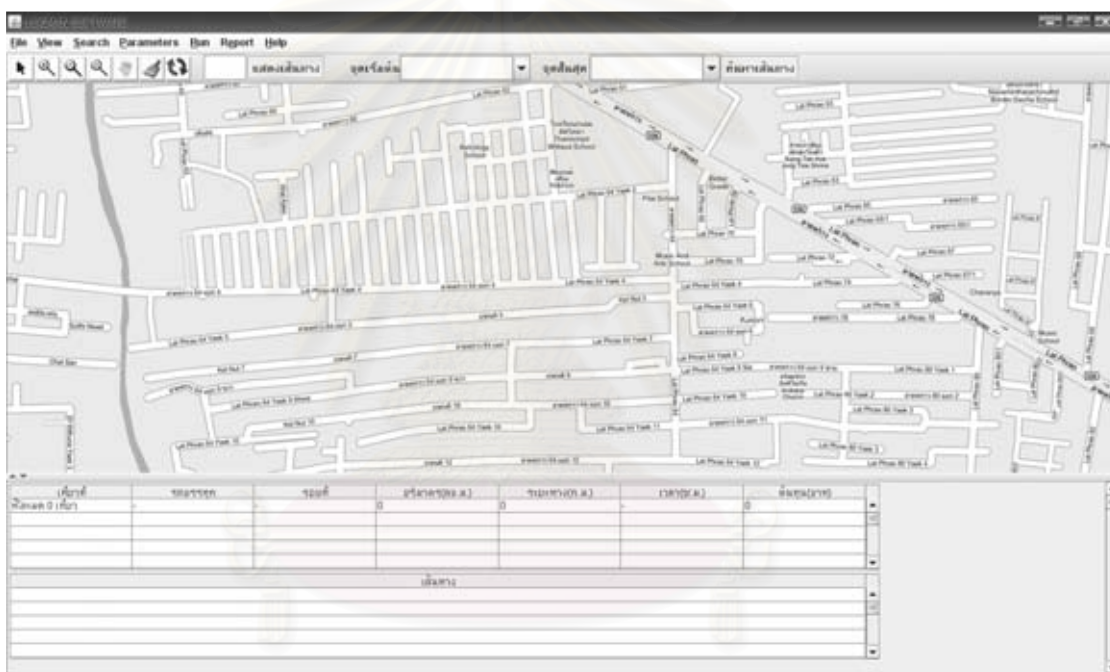
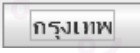



Figure 3.15: Vehicle routing dialog

- 1 At the main dialog, click button  to enter to the vehicle routing dialog which is shown in Figure 3.15
- 2 Choose menu Parameters->Assign Parameters to assigning the parameters for vehicle routing.
- 3 After the dialog, as shown in Figure 3.16, appears, click a radio button to choose a depot

- 4 Click tab  to enter customers' order information page

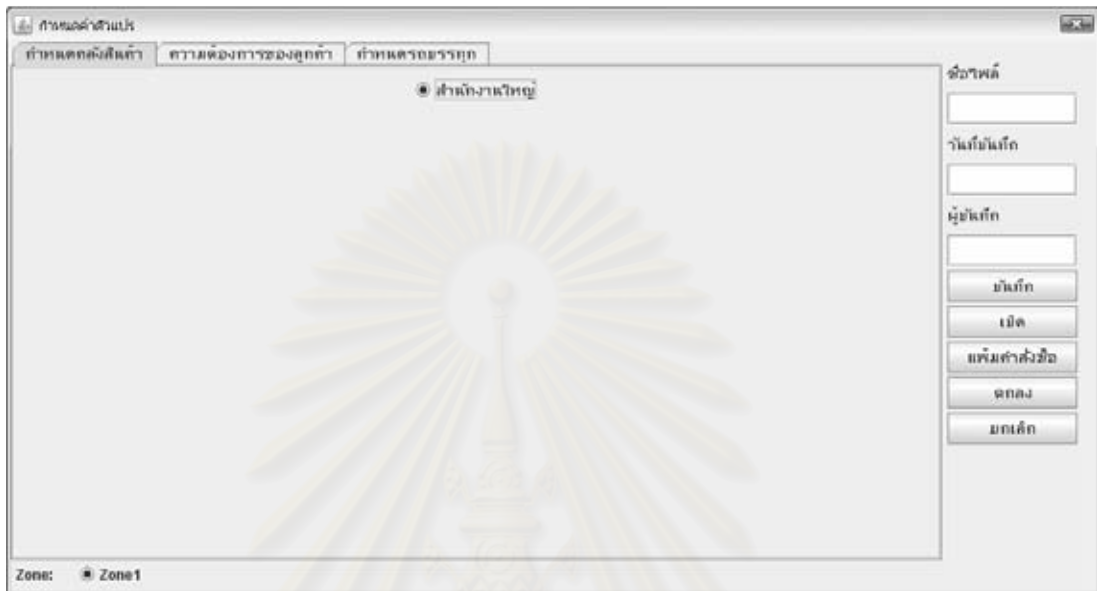


Figure 3.16: Dialog for assigning parameters

- 5 In the order information page, which is shown in Figure 3.17, choose customer(s) in the box number 1 and click button number 2 to add customer(s) to box number 3.

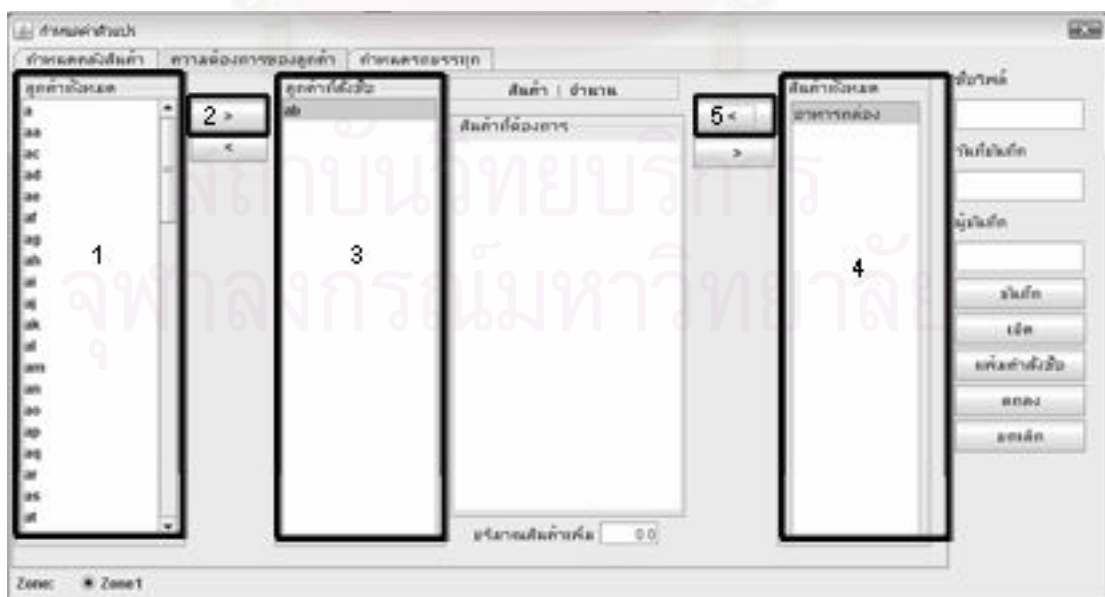
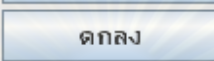


Figure 3.17: Order information page

- 6 Select a customer in box number 3, then choose a product in box number 4 and click button number 5 to add demand of the product to that customer. Dialog for enter number of demand will appear.
- 7 Repeat step 6 for each customer in box number 3 until a product is assigned to every customer. Click button  to confirm the parameters and exit the dialog.
- 8 In the vehicle routing dialog, choose menu “Run->Run Path Routing & Truck Scheduling” to run the vehicle routing process.
- 9 After the process finish, the solution will be displayed in road map and tables on the bottom of the dialog.

3.7 Conclusion

A software was developed to solve the vehicle routing problem. It was modified from a vehicle routing software which originally was used for large scale problems and ignored the time constraint by using Savings algorithm and A* search algorithm. The software was proved to be effective by testing it with published instances with best known solutions. A procedure to use the system was also developed to facilitate the use of the system.

CHAPTER IV

SYSTEM EVALUATION

After development and validation phases, the system was tested in order to prove that solutions resulted from the system are suitable for the problem. This chapter describes the system evaluation which includes the evaluation procedure, the information of the customers used for the test, results and discussion.

4.1 Evaluation Procedure

In order to prove that the system would work properly, it was tested with a set of 78 customers. Information of the customers, including locations of the customers was acquired from customer survey conducted in the delivery area. The demands of the customers varied from one to four units per customer. 90% of customers order 1 unit, 5% order 2 units, 4% order 3 units and 1% order 4 units, as shown in Figure 4.1. Capacity limit of each vehicle was set to 40 units and time constraint was set to 1 hour 30 minutes. Drop-off time for each customer was 2 minutes or 0.0334 hour. The test was run on a laptop with Intel Pentium Dual-Core 1.73 GHz, RAM 1014 MB and Windows XP operating system. Additionally, the operating procedure described in 3.6 was used to run the system test to prove the validity of the procedure.

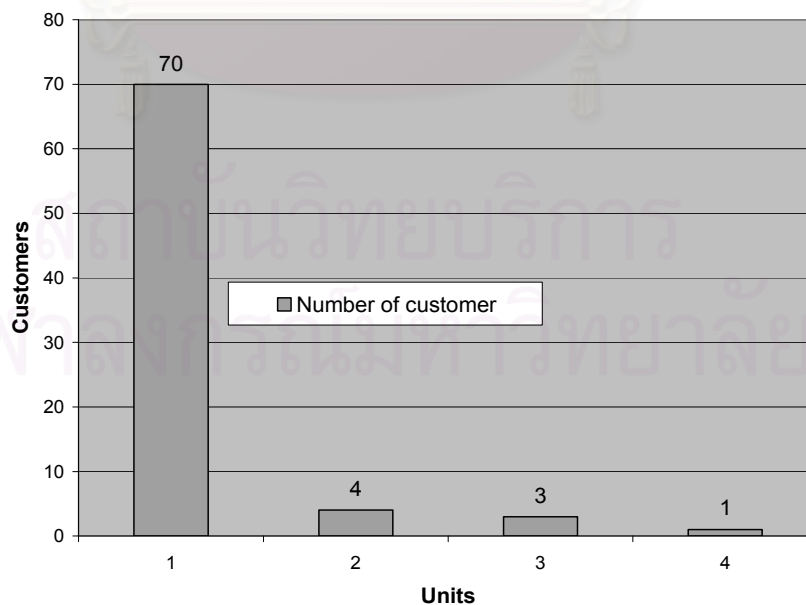


Figure 4.1: Distribution of demand

Table 4.1: Customer's data used for testing

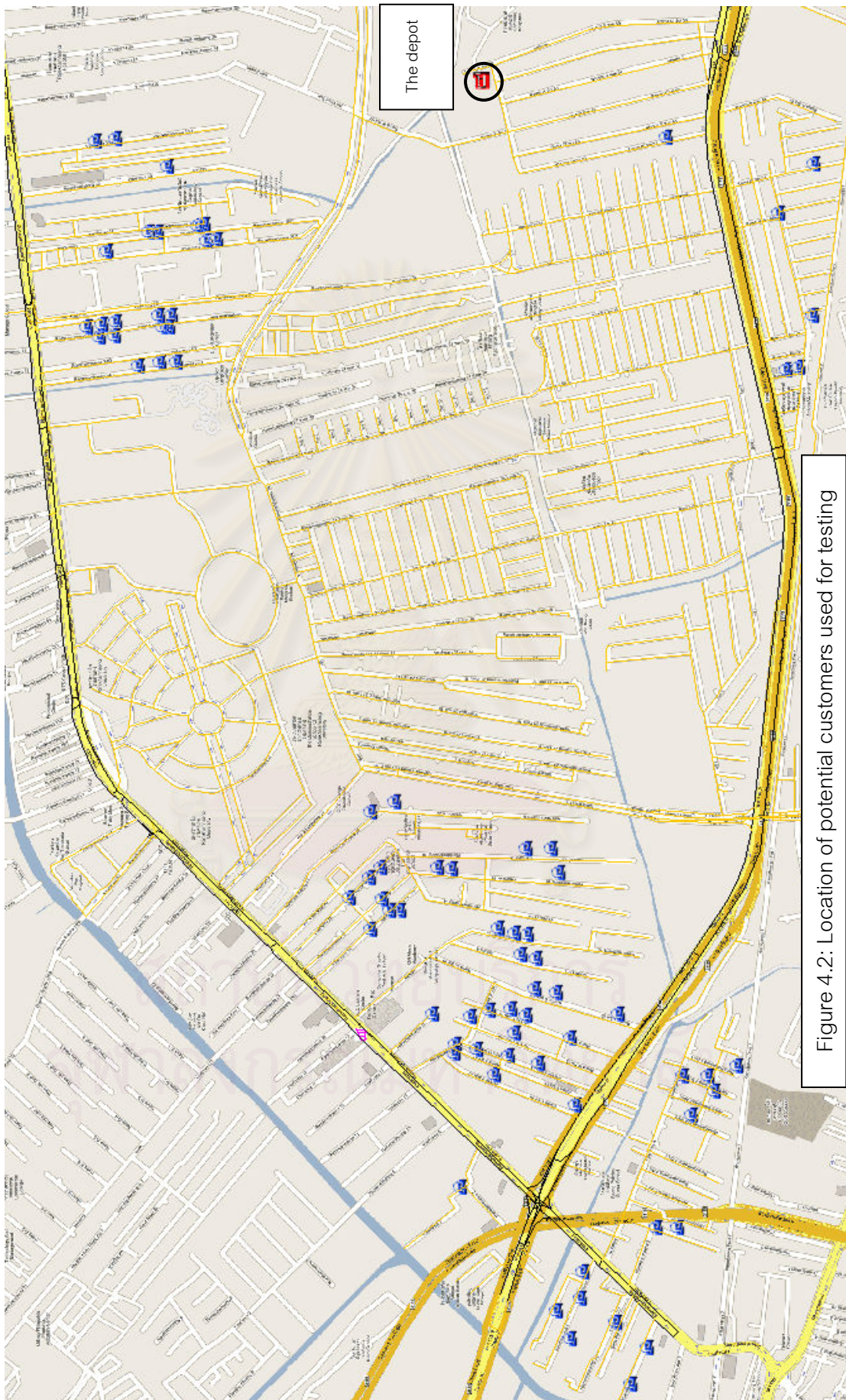
| ID | Name | Demand | EdgeID | X | Y | ID | Name | Demand | EdgeID | X | Y |
|----|------|--------|--------|------|------|----|------|--------|--------|------|------|
| 1 | a | 1 | 1365 | 2677 | 2188 | 40 | an | 1 | 127 | 1373 | 2994 |
| 2 | b | 1 | 1365 | 2680 | 2216 | 41 | ao | 1 | 113 | 1298 | 3058 |
| 3 | c | 1 | 1365 | 2684 | 2238 | 42 | ap | 1 | 113 | 1390 | 2842 |
| 4 | d | 2 | 1365 | 2679 | 2215 | 43 | aq | 1 | 274 | 1459 | 2805 |
| 5 | e | 1 | 1365 | 2697 | 2318 | 44 | ar | 3 | 103 | 1406 | 2845 |
| 6 | f | 1 | 1365 | 2701 | 2335 | 45 | as | 1 | 292 | 952 | 3069 |
| 7 | g | 1 | 1365 | 2701 | 2335 | 46 | at | 1 | 297 | 1152 | 2855 |
| 8 | h | 1 | 1365 | 2685 | 2244 | 47 | au | 1 | 480 | 1281 | 3266 |
| 9 | i | 1 | 1365 | 2700 | 2345 | 48 | av | 1 | 165 | 1080 | 3204 |
| 10 | j | 1 | 1368 | 2863 | 2406 | 49 | aw | 2 | 1385 | 2852 | 2314 |
| 11 | k | 1 | 1368 | 2863 | 2399 | 50 | ax | 1 | 1385 | 2851 | 2300 |
| 12 | l | 1 | 1383 | 2838 | 2420 | 51 | ay | 1 | 41 | 3014 | 2204 |
| 13 | m | 1 | 1383 | 2836 | 2405 | 52 | az | 1 | 41 | 3020 | 2239 |
| 14 | n | 4 | 1370 | 2818 | 2234 | 53 | ba | 1 | 68 | 1668 | 2656 |
| 15 | o | 1 | 546 | 1811 | 2698 | 54 | bb | 1 | 78 | 1664 | 2715 |
| 16 | p | 1 | 546 | 1831 | 2737 | 55 | bc | 1 | 83 | 1703 | 2693 |
| 17 | q | 1 | 86 | 1750 | 3017 | 56 | bd | 1 | 83 | 1722 | 2714 |
| 18 | r | 1 | 86 | 1749 | 2972 | 57 | be | 1 | 79 | 1721 | 2814 |
| 19 | s | 1 | 80 | 1704 | 3028 | 58 | bf | 1 | 79 | 1719 | 2840 |
| 20 | t | 1 | 67 | 1653 | 2740 | 59 | bg | 1 | 65 | 1610 | 2695 |
| 21 | u | 1 | 279 | 1492 | 3026 | 60 | bh | 1 | 67 | 1644 | 2730 |
| 22 | v | 1 | 272 | 1469 | 2946 | 61 | bi | 1 | 312 | 1612 | 2922 |
| 23 | w | 1 | 272 | 1457 | 2971 | 62 | bj | 1 | 312 | 1605 | 2951 |
| 24 | x | 1 | 908 | 2611 | 3459 | 63 | bk | 1 | 312 | 1601 | 2974 |
| 25 | y | 1 | 908 | 2607 | 3435 | 64 | bl | 2 | 113 | 1351 | 2915 |
| 26 | z | 1 | 1408 | 2705 | 3482 | 65 | bm | 1 | 134 | 1000 | 3073 |
| 27 | aa | 1 | 1475 | 2974 | 3485 | 66 | bn | 1 | 134 | 932 | 3027 |
| 28 | ab | 1 | 2367 | 3021 | 3222 | 67 | bo | 1 | 51 | 866 | 3189 |
| 29 | ac | 1 | 1449 | 2886 | 3421 | 68 | bp | 1 | 51 | 801 | 3159 |
| 30 | ad | 1 | 37 | 2964 | 2332 | 69 | bq | 1 | 136 | 910 | 3131 |
| 31 | ae | 2 | 1362 | 2620 | 2319 | 70 | br | 3 | 166 | 1078 | 3242 |
| 32 | af | 3 | 1362 | 2614 | 2282 | 71 | bs | 1 | 505 | 1322 | 3308 |
| 33 | ag | 1 | 1362 | 2621 | 2348 | 72 | bt | 1 | 507 | 1348 | 3296 |
| 34 | ah | 1 | 236 | 1427 | 2951 | 73 | bu | 1 | 528 | 1367 | 3347 |
| 35 | ai | 1 | 268 | 1467 | 2905 | 74 | bv | 1 | 459 | 1348 | 3250 |
| 36 | aj | 1 | 279 | 1505 | 2988 | 75 | bw | 1 | 279 | 1459 | 3137 |
| 37 | ak | 1 | 305 | 1565 | 2963 | 76 | bx | 1 | 124 | 1396 | 2885 |
| 38 | al | 1 | 305 | 1559 | 2989 | 77 | by | 1 | 114 | 1384 | 2940 |
| 39 | am | 1 | 266 | 1417 | 3050 | 78 | bz | 1 | 112 | 1455 | 2872 |

4.2 Test Input Data

Data of customers used for the test includes locations and demands. The information was acquired from customer survey in the delivery area. Detail of the test data is shown on Table 4.1. The identification numbers for each customer are shown in "ID" column. The second column of the table shows the potential customers' names

which are currently not known. The location of each customer can be plotted in the Bangkok road map as shown in Figure 4.2. Each customer's location is represented in coordinate system, shown in column "X" and "Y", which must be associated with the road, or edge, it is located in. The column "EdgeID" shows the identification number of the edge. Demands of the customers are generated according to the distribution shown in Figure 4.1.

Customer's location used in the system is represented in coordinate system which must be processed with shortest path finding to calculate the distance between delivery points for routing process. Demand is the number of units. The constraints include capacity and total time to reach the last customer. On the other hand, the instances' input used for system validation in CHAPTER III also represents customer location in coordinate system with Euclidean distance calculation. Its demand is also the number of units. Capacity is the only constraint for the instance. With the reasons that the routing process uses only data of distance between delivery points, demand and constraints for its procedure, and time constraint is not too complex to be relaxed, the evaluation input is equivalent to the instances' input.



4.3 Results and Discussion

The results of the test are described in this section. Additionally, its discussion is presented to illustrate the advantages and disadvantages of the system.

4.3.1 Results

Result from the program includes routes summary and details of each route, as shown in Figure 4.3 to Figure 4.6. Each route of the result must satisfy both capacity and time constraints of the problem. The system test takes 1 hour 6 minutes 47 seconds. The summary of the result is shown on Table 4.2 and detail of each route is shown on Table A.1. The solution has 4 routes with the loaded demands of 16 units, 29 units, 22 units and 24 units respectively. The demands indicate that the solution does not violate the capacity constraint. Additionally, the travelling times to last customer for the routes are 44 minutes, 1 hour 19 minutes, 1 hour, and 56 minutes respectively. According to the result, it can be suggested that every motorcycle is able to deliver the meal to its customers within specific time. It can be summarized that the system can find solutions which satisfy both capacity and time constraints. The total cost of the result can be calculated from Baht 7,000 per week for 4 motorcycles, and transportation cost Baht 56.32 per day, which equal to Baht 7,394.24 per week.

Table 4.2: Routes summary of result from the test

| Route No. | Vehicle | Round No. | Loaded Demand (unit) | Travel Distance (kilometre) | Travelling time to last customer (hour) | Cost (baht) |
|------------|-----------|-----------|----------------------|-----------------------------|---|-------------|
| เที่ยวที่1 | Motorycle | 1 | 16 | 11.47 | 0:44 | 12.62 |
| เที่ยวที่2 | Motorycle | 1 | 29 | 17.16 | 1:19 | 18.88 |
| เที่ยวที่3 | Motorycle | 1 | 22 | 12.52 | 1:00 | 13.77 |
| เที่ยวที่4 | Motorycle | 1 | 24 | 10.05 | 0:56 | 11.05 |

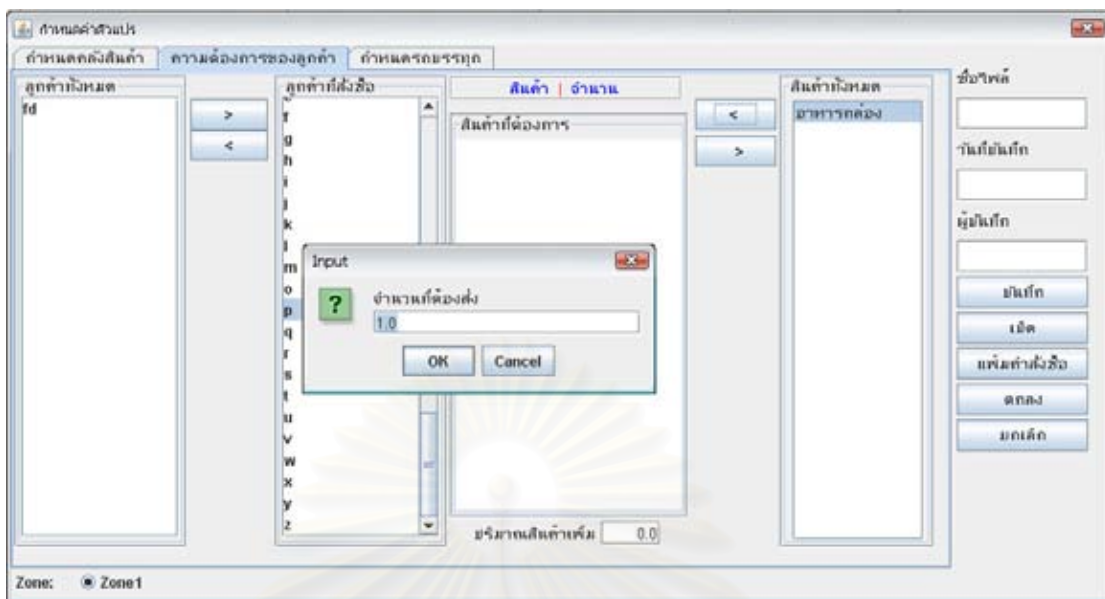


Figure 4.3: Example of entering customer order

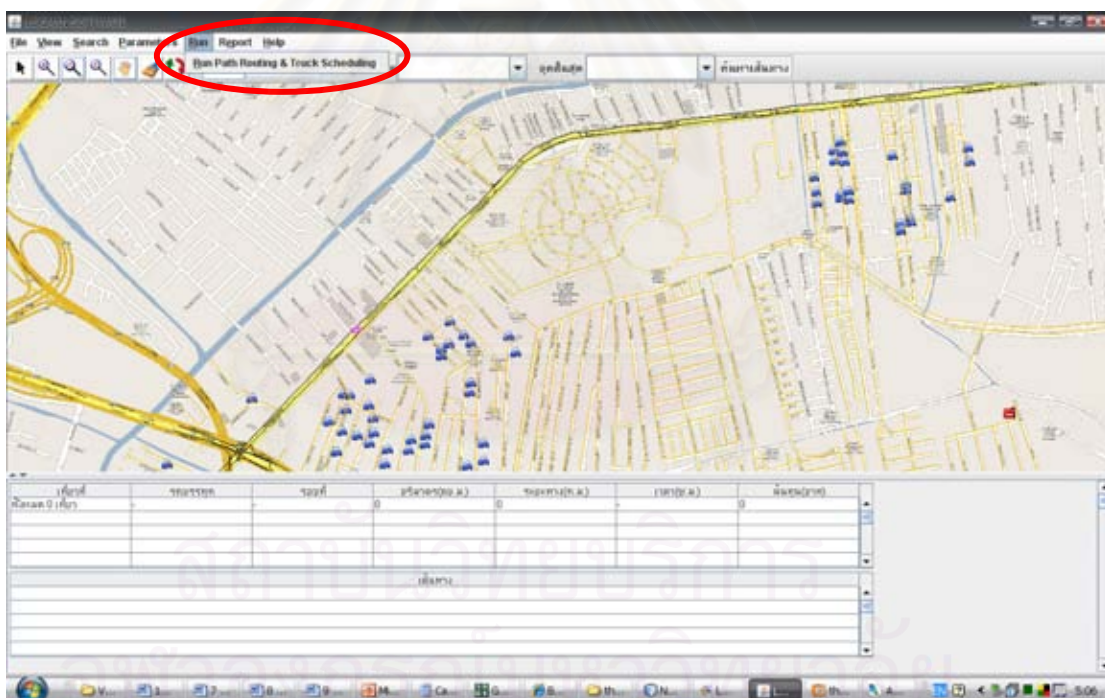


Figure 4.4: Running the system after entering the orders

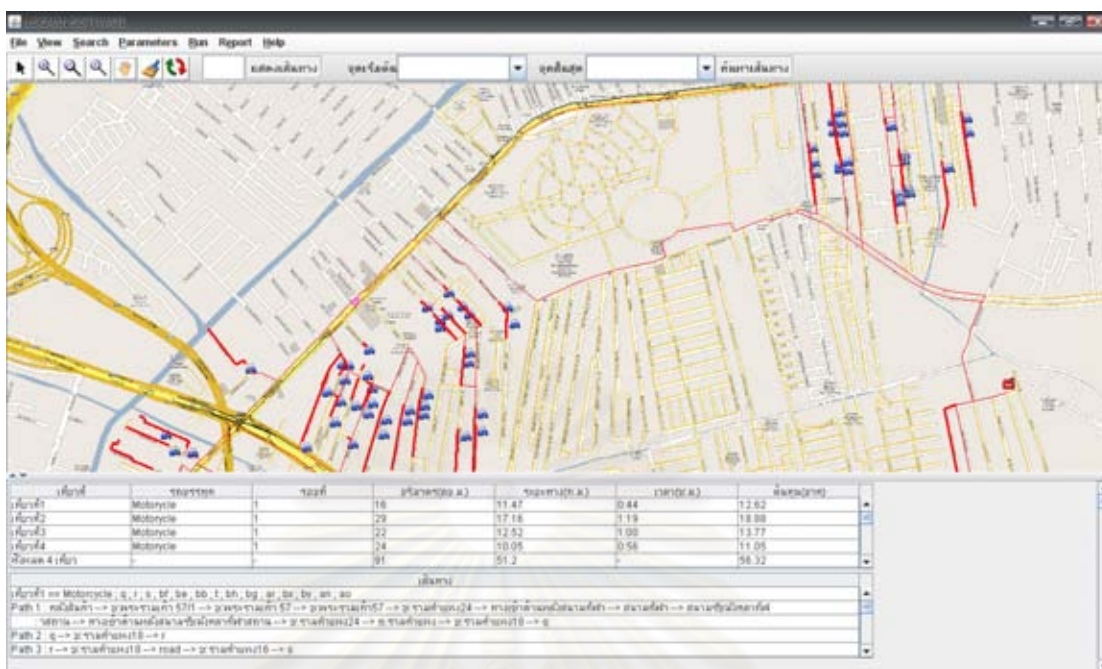


Figure 4.5: Result of the software

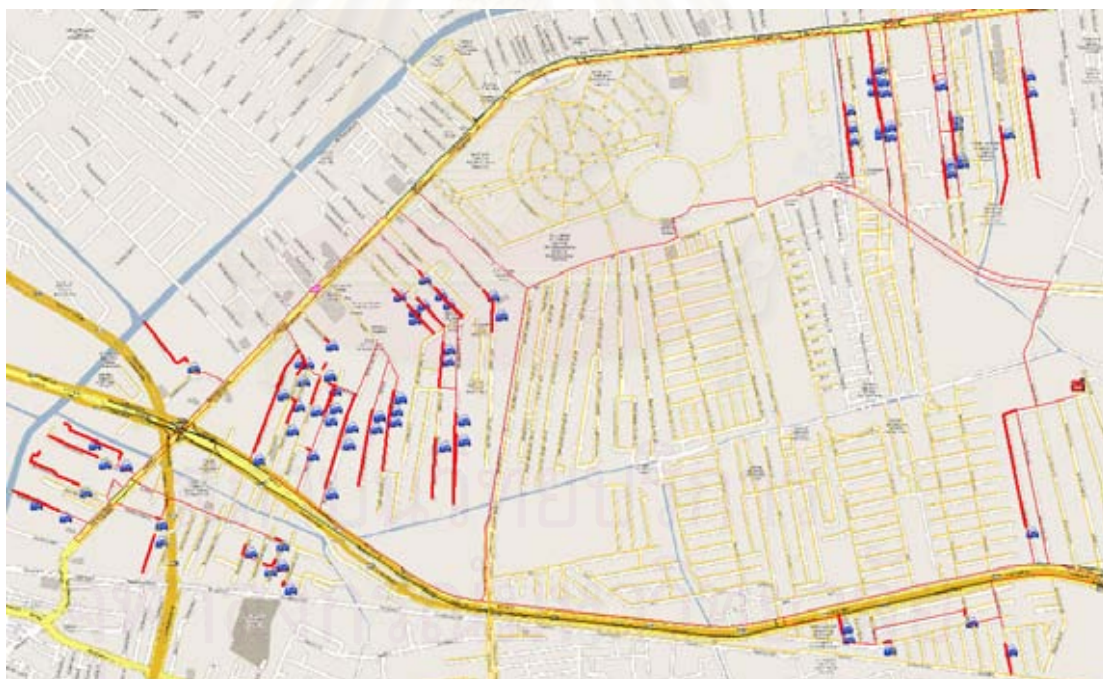


Figure 4.6: All routes displayed in the Bangkok road map

4.3.2 Discussion

It can be noticed that the result as obtained can be considered not an optimum level. There are still rooms of delivery that can be added. This depends entirely

on the strong marketing approach conducted by the company. It is possible for change of input data.

Beside the possibility for additional customer demand as described above, it is considered that the total demand of 91 units can be divided into 3 routes. Therefore, expenditures can be consequently reduced with the adjustments as following.

- Time Constraint

The results show that the time constraint affects the number of routes. Therefore, another test was conducted with the same input and parameters, but the time constraint was set to 1 hour and 45 minutes. The software was also modified to show the arrival time for each customer, as shown in Figure 4.7. The test takes 1 hour 5 minutes 22 seconds. Summary of the results is shown on Table 4.3 and detail of each route is shown on Table A.2, Appendix. The result has only 3 routes. The demands of the routes are 24 units, 37 units and 30 units with the travelling times of 1 hour 15 minutes, 1 hour 42 minutes, and 1 hour 4 minutes respectively. The total cost of the result can be calculated from Baht 5,250 per week for 3 motorcycles and transportation cost baht 51.06 per day, which equal to Baht 5,607.42 per week. 5 customers in the second route cannot receive the meal boxes within 1 hour 30 minutes.

Table 4.3: Routes summary of result from the time frame adjustment

| Route No. | Vehicle | Round No. | Loaded Demand (unit) | Travel Distance (kilometre) | Travelling time to last customer (hour) | Cost (baht) |
|------------|------------|-----------|----------------------|-----------------------------|---|-------------|
| เที่ยวที่1 | Motorcycle | 1 | 24 | 16.8 | 1:15 | 18.48 |
| เที่ยวที่2 | Motorcycle | 1 | 37 | 18.87 | 1:42 | 20.75 |
| เที่ยวที่3 | Motorcycle | 1 | 30 | 10.76 | 1:04 | 11.83 |

```

// the algorithm uses distance_matrix, time_matrix, and customers' demand to find a solution
// according to the design, the specific time which each vehicle must reach
// the last customer is assigned to 1.50
double m_time;
m_time = 1.5;
// For reaching, set the limited time to 1.75 and 2.00 respectively
//1.75
m_time = 1.75;

```

```

Output - logman2008 (run)
Route: 0: 1
0:13 reach a , 0:12 leave a
0:14 reach a , 0:14 leave a
0:17 reach af , 0:17 leave af
0:19 reach be , 0:22 leave be
0:22 reach bb , 0:25 leave bb
0:25 reach c , 0:27 leave c
0:27 reach bb , 0:29 leave bb
0:30 reach bg , 0:32 leave bg
0:33 reach ar , 0:36 leave ar
0:38 reach ba , 0:39 leave ba
0:39 reach by , 0:40 leave by
0:40 reach an , 0:42 leave an
0:44 reach ao , 0:46 leave ao
0:53 reach p , 0:55 leave p
0:55 reach s , 0:57 leave s
1:0 reach h1 , 1:3 leave h1
1:3 reach ap , 1:5 leave ap
1:5 reach ab , 1:9 leave ab
1:9 reach aa , 1:10 leave aa
1:15 reach ab , 1:17 leave ab
1:17 reach the depot
1:17 reach aa , 0:5 leave aa

```

Figure 4.7: Arrival time for each customer, generated by the program

According to the time frame adjustment, the overall cost was reduced by 24.16%. However, 5 customers will not receive the meal boxes within the time frame of 1 hour 30 minutes. The company may consider dispatching the second route 15 minutes earlier than the others. The company may pay Baht 2,000 for the fixed charge of the earlier dispatched route of which motorcyclist has to reach the depot earlier than others. It is a better choice as compared to hiring another motorcycle with additional fixed charge of Baht 1,750. Nevertheless, this can be done with the conditions that meal boxes must be ready for delivery earlier and the 5 first customers in that route can accept the delivery of meal boxes earlier than the agreed time.

- Drop-off Time

Because the time constraint affects the result, number of customers, of which each takes 2 minutes for dropping off the meal boxes, is another important variable. For example, if a motorcycle delivers 40 meal boxes to 40 customers, most of the time, or 80 minutes, will be used for dropping off the boxes and there will be only 10

minutes left for the transportation. Therefore, if the company can reduce the drop-off time, the fleet size should be reduced.

According to this assumption, another test was run with the adjustment of drop-off time. The drop-off time was initially reduced to 1 minute 30 seconds. The test takes 1 hour 23 minutes and 43 seconds. The result is the same as the test with time frame adjustment. The summary is shown on Table 4.4 and details of each route are the same as the one resulted from the test with time frame adjustment, as shown on Table A.2. Consequently, the total expenditure can be calculated equal to Baht 5,607.42 per week which is the same as the result of time frame adjustment. Therefore, it can be considered that reducing drop-off time can reduce the fleet size and total cost.

Table 4.4: Routes summary of result from the first drop-off time adjustment

| Route No. | Vehicle | Round No. | Loaded Demand (unit) | Travel Distance (kilometre) | Travelling time to last customer (hour) | Cost (baht) |
|-------------|------------|-----------|----------------------|-----------------------------|---|-------------|
| เที่ยวที่ 1 | Motorycle1 | 1 | 24 | 16.8 | 1:04 | 18.48 |
| เที่ยวที่ 2 | Motorycle1 | 1 | 37 | 18.87 | 1:25 | 20.75 |
| เที่ยวที่ 3 | Motorycle1 | 1 | 30 | 10.76 | 0:54 | 11.83 |

Additionally, the drop-off time adjustment was refined by increasing it to 1 minutes and 45 seconds. The test takes 1 hour 24 minutes and 9 seconds. The summary of results is shown on Table 4.5 and details of each route are shown on Table A.3. The weekly expenditures include fixed charge of Baht 5,250 per week and transportation charge of Baht 53.73 per day. The total weekly payment is equal to Baht 5,626.11 per week.

Table 4.5: Routes summary of result from the second drop-off time adjustment

| Route No. | Vehicle | Round No. | Loaded Demand (unit) | Travel Distance (kilometre) | Travelling time to last customer (hour) | Cost (baht) |
|------------|-----------|-----------|----------------------|-----------------------------|---|-------------|
| เที่ยวที่1 | Motorycle | 1 | 29 | 19.86 | 1:24 | 21.84 |
| เที่ยวที่2 | Motorycle | 1 | 30 | 10.76 | 0:59 | 11.83 |
| เที่ยวที่3 | Motorycle | 1 | 32 | 18.24 | 1:22 | 20.06 |

Because the longest time used to reach the last customer is 1 hour 24 minutes, it is possible for the drop-off time for each customer to be longer than 1 minute 45 seconds. Therefore, further refinement was done with the same routes by re-calculating only the travelling time to the last customer with longer drop-off time of 1 minute 50 seconds, 1 minute 55 seconds, and 2 minutes respectively. The results of the further refinement are shown on Table 4.6. The results show that the fleet size can be reduced with reduction of only 5 seconds for drop-off time.

Table 4.6: Travelling time to last customer, re-calculated from the test with drop-off time of 1 minute 45 seconds

| Route No. | Drop-off Time (hour:minute:second) | | | |
|-----------|------------------------------------|--------------|--------------|---------|
| | 1 min 45 sec | 1 min 50 sec | 1 min 55 sec | 2 min |
| 1 | 1:24:00 | 1:26:05 | 1:28:10 | 1:30:15 |
| 2 | 0:59:00 | 1:00:45 | 1:02:30 | 1:04:15 |
| 3 | 1:22:00 | 1:24:25 | 1:26:50 | 1:29:15 |

The drop-off time adjustment is seemed to be a better choice compared to dispatching earlier. However, the drop-off time of two minutes used in the system is just a fair estimate. To ensure that the time is shorter than 1 minute 55 seconds, the company has to find a practical method to reduce the drop-off time.

4.4 Conclusion

The system was tested with a set of 78 customers after its development. The test result does not violate both constraints. The result show the possibility for

additional demands. On the other hand, cost reduction can be done by time frame and drop-off time adjustments.



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

CHAPTER V

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

This thesis studied the vehicle routing problem of a planned breakfast delivery service in Bangkok. The business plans to use motorcycles for the delivery. The company has containers for loading up to 40 meal boxes which will be fasten with the motorcycles daily. The delivery expenditures include Baht 1,750 of weekly payment for salary of each delivery man, and Baht 1.10 per kilometre for transportation charge. The depot is located near Rama IX Road and Srinakarintra Road with the delivery area of approximately 5 kilometre radius around it. The problem has limitation of vehicle capacity and time frame which each vehicle has to reach its last customer. The purpose of developing the vehicle routing system is to find solutions for the case company to deliver meals to its customer with optimal fleet size and total travel distance under the condition that each vehicle must be loaded not more than its capacity and must take travelling time not longer than the specific time frame. The system was developed with assumptions including weekly changed delivery points, one trip per day for each motorcycle, ignorance of the condition of traffic and irregularities such as accidents and closing routes, excluding the expressway and the motorway from delivery routes, two-way traffic in small roads with average velocities of 25 kilometre per hour, one-way traffic for each side of main streets with dividing islands with average velocities of 45 kilometre per hour, and drop-off time of 2 minutes for each customer. The scope of thesis includes finding a method for problem solving, modelling the problem, implementation in computerized simulation program, data preparation for the system, formulating the operating procedure of the system for the company, and testing the system.

The research initially reviewed related theories and applications of theories, with the purposes of studying characteristic and solution approaches of the problem. Because only location of the customers can be used for the routing, the problem is consisted of shortest path problem and vehicle routing problem. Most of

reviewed researches use the Clarke-Wright's method, Savings algorithm, due to its simplicity.

The solution approaches used in the vehicle routing system includes Savings algorithm for the routing process and A* search algorithm for shortest path problem. Savings algorithm was chosen for the routing process because of its simplicity and widely usages. The procedure of the approach includes initially building a set of routes which each customer is assigned to each route, calculating of saving value between each pair of customers, selecting the highest saving value, and merging the routes to form the solution while feasible. A* search algorithm implemented in the system uses distance of the path and the distances from all routes in the path to target node for finding consideration. The developed system is divided into 3 parts:

- Inputs includes customers' locations and customers' demands
- Computation uses the input in the algorithms to find a solution for the problem
- Outputs are displayed in road map. Detail and summary of all routes are also shown in the system.

The system was developed by modifying a vehicle routing program developed by Viroj Putvithee [32] with a copyright of National Electronic and Computer Technology Center (NECTEC), member of National Science and Technology Development Agency (NSTDA), and Water Pacific Part., Ltd. The program originally used Forward algorithm, a modification of A* search algorithm, and a modification of savings algorithm. The program can solve the vehicle routing problem of trucks with capacity constraint, without time constraint. Because the software was developed for big scale problem, the shortest path finding ignores the case of having several customers on the same road, the distance from road intersection to customers, and choosing start and target nodes which give the actual shortest path. Additionally, the solution procedure starts from routing process using only direct distance between customers,

then finds the actual travelling path by using the shortest path module. Because of the development for truck routing, the software had to be modified to suit the small scale problem like motorcycles routing. The modification was made in 2 parts: shortest path finding control module and routing process. The modification of shortest path finding control module includes adding the case of having several customers located in same edge, using alternate node of edges to find actual shortest path, and consideration of distance from node to customer location. For the routing process, the savings algorithm module, of which procedure is already described in 3.3.2 Computation, was developed. Additionally, the solution finding was redesigned to use the shortest path in the routing process for its accuracy. The shortest path computation module, user interface and output display module of the software are not modified. The user interface uses Thai language for practicality. The software is tested with instances of capacitated vehicle routing problem in order to prove that it can effectively find a solution for the problem. The instances, authored by Augerat et al, can be found in [2] and is divided into 3 set: A, B, P. Results of set A are different from the known best results in average of 5.03%. The average difference of set B is 4.04% and that of set P is 9.50%. The overall average is 6.17% which indicates that the performance of the developed system is close to the most effective approach. The number of resulted routes which is mostly as same as the best result shows the ability of the system to minimize the fleet size. The procedure of operating the system was also formulated for the case company.

The system was tested with 78 customers. The customers' locations are retrieved from customer survey conducted in the delivery area. The drop-off time is 2 minutes and demands of the customers varied from one to four units per customer. 90% of customers order 1 unit, 5% order 2 units, 4% order 3 units and 1% order 4 units. The time constraint was 1 hour 30 minutes and maximum capacity was 40 units. Result of the test has 4 routes with demands of 16 units, 29 units, 22 units and 24 units, and the travelling time of 44 minutes, 1 hour 19 minutes, 1 hour, and 56 minutes respectively. The total operating expenditure is Baht 7394.24 per week. The result indicates that the system can find solutions which satisfy both constraints. According to the result, all routes of the result have a lot of room left for their capacity. Although the result has

flexibility, the number of routes can be reduced to 3 routes because the total demand is 91 units.

In order to explore the possibility of improving the result, another test with allowance to violate the time constraint was performed. The result has 3 routes of which one violates the time constraint. The cost was reduced by Baht 1,786.82 per week or 24.16% to Baht 5,607.42 per week. The company can choose to dispatch one vehicle 15 minutes earlier than initially planned or let 5 customers receive their meals up to 15 minutes late. The former alternative may be negotiated with the contracted motorcycle with some extra pay which obviously can be cheaper than hiring another motorcycle.

Another way to improve the result besides allowing more delivery time in a route is to reduce the drop-off time. The drop-off time was reduced from 2 minutes. The reduction of only 5 seconds of drop-off time can reduce the number of routes, and consequently the fleet size from four to three. However, the drop-off time of two minutes is a fair estimate. Therefore, the company has to find a practical method to reduce the drop-off time.

5.2 Recommendations

This section explains the implementation of the developed system in this study. It also recommends further work that can add value to this research.

A. Implementation of the system

- Because this thesis does not use real data, the company has to collect the real data of customers which can be plotted in the Bangkok roadmap of the system.
- The resulted fleet size in this thesis is just an example. The company should run the system with real data before starting of the operation.

B. Extension of the system

- The travelling time in the system is calculated with the average velocities of 25 kilometre per hour and 45 kilometre per hour. To make the system more realistic to the actual traffic condition, different and varying velocities may be considered to calculate the travelling time.
- In some situation, the vehicles must not arrive each customer sooner or later than the specific time window assigned by that customer. Therefore, the system should be modified to suit the problem conditions.
- The routing process can be modified with other approaches such as Simulated Annealing or applying routes improvement approaches with a constructive approach.



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

REFERENCES

- [1] Orawan Tunsitjareankun. A heuristic approach for solving the vehicle-routing problem in solid wastes collection in Bang Khen area. Master's thesis Department of Industrial Engineering Faculty of Engineering Chulalongkorn University, 1991.
- [2] Bernabé Dorronsoro Diaz. The VRP Web [Online]. Collaboration between AUREN and the Language and Computation Science department of the University of Málaga, Available from : <http://neo.lcc.uma.es/radi-aeb/WebVRP/>, [2007].
- [3] Naruporn Kanchanarat. A transportation routing system: a case study of transporting knock-down furniture. Master's thesis Department of Industrial Engineering Faculty of Engineering Chulalongkorn University, 1999.
- [4] Pongpaut Totrakool. A heuristic search method for a vehicle routing problem in a medical supplies distribution system. Master's thesis Department of Industrial Engineering Faculty of Engineering Chulalongkorn University, 2003.
- [5] Wikipedia. Branch and bound [Online]. Available from : http://en.wikipedia.org/wiki/Branch_and_bound, [2007].
- [6] Krisakrai Manimmanakorn. Vehicle routing for gasoline delivery. Master's thesis. Department of Industrial Engineering Faculty of Engineering Chulalongkorn University, 1995.
- [7] Sean L. Forman. Nearest Neighbor Algorithm [Online]. Mathematics and Computer Science Department, Saint Joseph's University. Available from : <http://www.baseball-reference.com/travel/class/nearest.html>, [2007].
- [8] Paul A. Jensen. Operations Research : Model & Method (Chapter 5) [Online]. University of Texas, Available from : <http://www.me.utexas.edu/~jensen/methods/net.pdf/netshp.pdf>, [2007].
- [9] Jesper Larsen & Jens Clausen. The Shortest Path Problem [Online]. Class slides Technical University of Denmark, Available from : <http://www.imada.sdu.dk/~jbj/DM85/lec6a.pdf>, [2007].
- [10] Somchai Prasitjutrakul. All-Pairs Shortest Path Problem [Online]. Class slides, Department of Computer Engineering, Faculty of Engineering, Chulalongkorn

University, Available from :

<http://www.cp.eng.chula.ac.th/~somchai/spj/slides/misc/AllPairsShortestPath.pdf>, [2007].

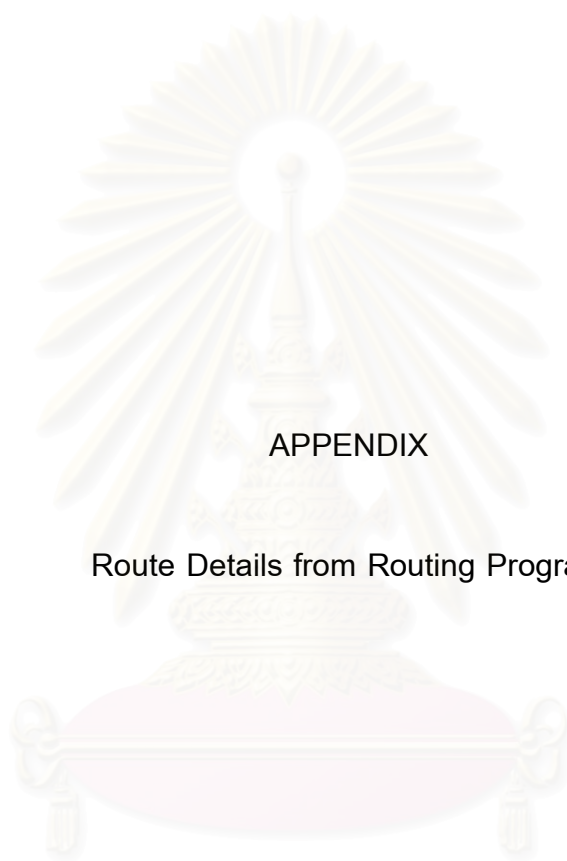
- [11] Wikipedia. Shortest path problem [Online]. Available from :
http://en.wikipedia.org/wiki/Shortest_path_problem, [2007].
- [12] Olli Bräysy. Genetic Algorithms for the Vehicle Routing Problem with Time Windows.
 Arpakannus 1/2001, Special issue on Bioinformatics and Genetic Algorithms,
 2001, pp. 33-38.
- [13] Wikipedia. Dijkstra's algorithm [Online]. Available from :
http://en.wikipedia.org/wiki/Dijkstra%27s_algorithm, [2007].
- [14] Boris V. Cherkassky, Andrew V. Goldberg, Tomas Radzik. Shortest path algorithms: Theory and Experimental Evaluation. Technical Report 93-1480, Computer Science Department, Stanford University, 1993.
- [15] F. Benjamin Zhan. Three Fastest Shortest Path Algorithms on Real Road Networks: Data Structures and Procedures. Journal of Geographic Information and Decision Analysis, 1, 1 : pp. 69-82.
- [16] F. Benjamin Zhan, Charles E. Noon. Shortest Path Algorithms: An Evaluation using Real Road Networks. Transportation Science, 32, 1(1998).
- [17] Wikipedia. Bellman-Ford algorithm [Online]. Available from :
http://en.wikipedia.org/wiki/Bellman-Ford_algorithm, [2007].
- [18] Wikipedia. A* search algorithm [Online]. Available from :
http://en.wikipedia.org/wiki/A%2A_search_algorithm, [2007].
- [19] Computer Science Department, Cornell University. Recitation 26: A* search [Online]. Available from :
<http://www.cs.cornell.edu/courses/cs312/2007sp/recitations/rec26.html>, [2007].
- [20] Martin Johnson. Informed Search Methods [Online]. Institute of Information and Mathematical Sciences, Massey University. Available from :
<http://www.massey.ac.nz/~mjohnso/notes/59302/I04.html>, [2007].
- [21] Wikipedia. Floyd–Warshall algorithm [Online]. Available from :
http://en.wikipedia.org/wiki/Floyd-Warshall_algorithm, [2007].

- [22] Marco Pellegrini. Algorithm and Data Structure APSP with negative weights [Online]. Class slides, Math. Department, Parma University. Spring, 2002. Available from : <http://www.cs.unipr.it/~zaffanella/Teaching/AlgoritmiStruttureDati/parma-03-21-six.pdf>, [2007].
- [23] Wikipedia. Johnson's algorithm [Online]. Available from : http://en.wikipedia.org/wiki/Johnson's_algorithm, [2007].
- [24] Travelling Salesman Problem: Insertion Algorithms [Online]. Online Logistics Tutorial, H. Milton Stewart School of Industrial & Systems Engineering, College of Engineering, Georgia Institute of Technology. May 19, 1999. Available from : http://www2.isye.gatech.edu/logisticstutorial/vehicle/tsp/tsp009_.htm, [2007].
- [25] Stefan Røpke. Polynomial time heuristics for the VRP [Online]. Department of Computer Science, University of Copenhagen. September, 2005. Available from : http://www.diku.dk/undervisning/2005e/426/slides2_4.pdf, [2007].
- [26] Marco Dorigo. Ant Colony Optimization [Online]. IRIDIA, Université Libre de Bruxelles, Belgium. Available from : <http://www.aco-metaheuristic.org>, [2007].
- [27] Alain Hertz, Eric Taillard, Dominique do Werra. A TUTORIAL ON TABU SEARCH [Online]. Available from : <http://www.cs.colostate.edu/~whitley/CS640/hertz92tutorial.pdf>, [2007].
- [28] Michel Gendreau. AN INTRODUCTION TO TABU SEARCH [Online]. Centre de recherche sur les transports and Département d'informatique et de recherche opérationnelle, Université de Montréal. July 2002. Available from : http://www.ifi.uio.no/infheur/Bakgrunn/Intro_to_TS_Gendreau.htm, [2007].
- [29] Ann Tighe and Finlay Smith. A Review of Artificial Intelligence Techniques in Fleet Logistics [Online]. Technical Report. Department of Information Technology, National University of Ireland, Galway, Ireland. 2002. Available from : <http://www.it.nuigalway.ie/publications/TR/abstracts/NUIG-IT-091002.pdf>, [2007].
- [30] K.C. Tan, L.H. Lee, Q.L. Zhu, K. Qu. Heuristic methods for vehicle routing problem with time windows. pp.281-295, Artificial Intelligence in Engineering, Vol.15, Issue 3, Jul 2001. Elsevier Science.

- [31] Vipada Supavita and Duangpan Krichcharnchai. Prototype of transportation management system [Online]. Conference paper. Operations Research Conference, 2006, Thailand. Available from : <http://as.nida.ac.th/~ornet/ORCRN2006/PAPERS/transport.pdf>, [2007].
- [32] Viroj Putvithee, et al. Development of Vehicle Assignment and Routing Software for Goods Distribution System. Water Pacific Part., Ltd. Funded by National Electronics and Computer Technology Center (NECTEC), member of National Science and Technology Development Agency (NSTDA), Thailand. 2004.



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย



APPENDIX

Route Details from Routing Program

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

Table A.1: Route details from a run of the routing program with initial conditions

(The first line lists the sequence of the customer names in the route. The following lines tell how to travel from one customer to the next one in the sequence)

| Route No. | Detail of route |
|-----------|--|
| 1 | เที่ยวที่1 == Motorcycle ; q ; r ; s ; bf ; be ; bb ; t ; bh ; bg ; ar ; bx ; by ; an ; ao |
| | Path 1 : คลังสินค้า --> ช.พระรามเก้า 57/1 --> ช.พระรามเก้า 57 --> ช.พระรามเก้า57 --> ช.รามคำแหง24 --> ทางเข้าด้านหลังสนามกีฬา --> สนามกีฬา --> สนามรัชมิ่งคลากีฬาสถาน --> ทางเข้าด้านหลังสนามรัชมิ่งคลากีฬาสถาน --> ช.รามคำแหง24 --> ถ.รามคำแหง --> ช.รามคำแหง18 --> q |
| | Path 2 : q --> ช.รามคำแหง18 --> r |
| | Path 3 : r --> ช.รามคำแหง18 --> road --> ช.รามคำแหง16 --> s |
| | Path 4 : s --> ช.รามคำแหง16 --> bf |
| | Path 5 : bf --> ช.รามคำแหง16 --> be |
| | Path 6 : be --> ช.รามคำแหง16 --> bb |
| | Path 7 : bb --> ช.รามคำแหง16 --> road --> ช.รามคำแหง14 --> t |
| | Path 8 : t --> ช.รามคำแหง14 --> bh |
| | Path 9 : bh --> ช.รามคำแหง14 --> bg |
| | Path 10 : bg --> ช.รามคำแหง14 --> ถ.รามคำแหง --> ช.รามคำแหง8 --> ar |
| | Path 11 : ar --> ช.รามคำแหง8 --> ช.พระรามเก้า31 --> bx |
| | Path 12 : bx --> ช.พระรามเก้า31 --> ช.พระรามเก้า31-33 --> by |
| | Path 13 : by --> ช.พระรามเก้า31-33 --> ช.พระรามเก้า33 --> an |
| | Path 14 : an --> ช.พระรามเก้า33 --> ช.รามคำแหง8 --> ช.พระรามเก้า29 --> ao |
| | Path 15 : ao --> ช.พระรามเก้า29 --> ถ.พระรามเก้า --> ช.พระรามเก้า 53 --> ช.พระรามเก้า 55 --> ช.พระรามเก้า 57 --> ช.พระรามเก้า 57/1 --> คลังสินค้า |
| 2 | เที่ยวที่2 == Motorcycle ; aa ; ac ; z ; x ; y ; bv ; bt ; bu ; bs ; au ; av ; br ; bp ; bo ; bq ; bn ; bm ; as ; at ; bd ; bc ; ba ; aq ; bz ; ai ; v ; w |
| | Path 1 : คลังสินค้า --> ช.พระรามเก้า 57/1 --> ช.พระรามเก้า 57 --> ช.พระรามเก้า 55 --> ถ.พระรามเก้า --> กลับรถ --> ถ.พระรามเก้า --> ช.พระรามเก้า62 --> aa |
| | Path 2 : aa --> ช.พระรามเก้า62 --> road --> ช.พระรามเก้า60 --> ac |

| Route No. | Detail of route |
|-----------|--|
| | Path 3 : ac --> ซ.พระรามเก้า60 --> road --> ซ.พระรามเก้า58 --> road --> z |
| | Path 4 : z --> road --> ซ.พระรามเก้า54 --> x |
| | Path 5 : x --> ซ.พระรามเก้า54 --> y |
| | Path 6 : y --> ซ.พระรามเก้า54 --> ถ.พระรามเก้า --> เลี้ยวซ้ายเข้า ถ.รามคำแหง --> ถ.รามคำแหง --> รามคำแหง4 --> bv |
| | Path 7 : bv --> รามคำแหง4 --> รามคำแหง4 ซอย7 --> bt |
| | Path 8 : bt --> รามคำแหง4 ซอย7 --> ทางระหว่างซอย7 กับซอย8 --> รามคำแหง4 ซอย8 --> bu |
| | Path 9 : bu --> รามคำแหง4 ซอย8 --> ทางระหว่างซอย7 กับซอย8 --> ทางระหว่างซอย6 กับซอย7 --> bs |
| | Path 10 : bs --> ทางระหว่างซอย6 กับซอย7 --> ทางระหว่างซอย5 กับซอย6 --> รามคำแหง4 ซอย5 --> au |
| | Path 11 : au --> รามคำแหง4 ซอย5 --> รามคำแหง4 --> รามคำแหง4-รามคำแหง2 --> ซ.รามคำแหง2 --> av |
| | Path 12 : av --> ซ.รามคำแหง2 --> br |
| | Path 13 : br --> ซ.รามคำแหง2 --> ถ.รามคำแหง --> จุดกลับรถหน้าการไฟฟ้า --> ถ.รามคำแหง --> ซ.รามคำแหง1 --> bp |
| | Path 14 : bp --> ซ.รามคำแหง1 --> bo |
| | Path 15 : bo --> ซ.รามคำแหง1 --> ถ.รามคำแหง --> ซ.รามคำแหง3/1 --> bq |
| | Path 16 : bq --> ซ.รามคำแหง3/1 --> road --> ถ.รามคำแหง --> ซ.รามคำแหง5 --> ซ.รามคำแหง5 แยก1 --> bn |
| | Path 17 : bn --> ซ.รามคำแหง5 แยก1 --> bm |
| | Path 18 : bm --> ซ.รามคำแหง5 แยก1 --> ซ.รามคำแหง5 --> as |
| | Path 19 : as --> ซ.รามคำแหง5 --> ถ.รามคำแหง --> ซ.รามคำแหง7 --> at |
| | Path 20 : at --> ซ.รามคำแหง7 --> ถ.รามคำแหง --> กลับรถหน้า ซ.รามคำแหง25 --> ถ.รามคำแหง --> ซ.รามคำแหง18 --> bd |
| | Path 21 : bd --> ซ.รามคำแหง18 --> bc |

| Route No. | Detail of route |
|-----------|--|
| | Path 22 : bc --> ซ.รามคำแหง18 --> ba |
| | Path 23 : ba --> ซ.รามคำแหง18 --> ถ.รามคำแหง --> ซ.รามคำแหง10 --> aq |
| | Path 24 : aq --> ซ.รามคำแหง10 --> ถ.รามคำแหง --> ซ.รามคำแหง8 --> ซ.พระรามเก้า 35 --> bz |
| | Path 25 : bz --> ซ.พระรามเก้า35 --> ซ.พระรามเก้า35-37 --> ai |
| | Path 26 : ai --> ซ.พระรามเก้า35-37 --> ซ.พระรามเก้า37 --> v |
| | Path 27 : v --> ซ.พระรามเก้า37 --> w |
| | Path 28 : w --> ซ.พระรามเก้า37 --> ถ.พระรามเก้า --> ซ.พระรามเก้า 53 --> ซ.พระราม เก้า 55 --> ซ.พระรามเก้า 57 --> ซ.พระรามเก้า 57/1 --> คลังสินค้า |
| 3 | เที่ยวที่3 == Motorcycle ; ag ; ae ; af ; p ; o ; bl ; ap ; ah ; am ; al ; ak ; bk ; bj ; bi ; aj ; u ; bw ; ab |
| | Path 1 : คลังสินค้า --> ซ.พระรามเก้า 57/1 --> ซ.พระรามเก้า 57 --> ซ.พระรามเก้า57 --> ซ.รามคำแหง24 --> กลับรถหน้า ABAC --> ซ.รามคำแหง24 --> ซ.รามคำแหง24/1 --> ag |
| | Path 2 : ag --> ซ.รามคำแหง24/1 --> ae |
| | Path 3 : ae --> ซ.รามคำแหง24/1 --> af |
| | Path 4 : af --> ซ.รามคำแหง24/1 --> ถ.รามคำแหง --> ซ.รามคำแหง22 --> p |
| | Path 5 : p --> ซ.รามคำแหง22 --> o |
| | Path 6 : o --> ซ.รามคำแหง22 --> ถ.รามคำแหง --> ซ.รามคำแหง8 --> ซ.พระรามเก้า29 --> bl |
| | Path 7 : bl --> ซ.พระรามเก้า29 --> ap |
| | Path 8 : ap --> ซ.พระรามเก้า29 --> ซ.รามคำแหง8 --> ซ.พระรามเก้า35 --> ah |
| | Path 9 : ah --> ซ.พระรามเก้า35 --> ซ.พระรามเก้า35-37 --> am |
| | Path 10 : am --> ซ.พระรามเก้า35-37 --> ซ.พระรามเก้า37 --> ซ.พระรามเก้า37-39 --> ซ.พระรามเก้า39 --> ซ.รามคำแหง12 --> ซ.อุดมยศ4 --> al |
| | Path 11 : al --> ซ.อุดมยศ4 --> ak |
| | Path 12 : ak --> ซ.อุดมยศ4 --> ซ.รามคำแหง12 --> bk |

| Route No. | Detail of route |
|-----------|---|
| | Path 13 : bk --> ซ.รามคำแหง12 --> bj |
| | Path 14 : bj --> ซ.รามคำแหง12 --> bi |
| | Path 15 : bi --> ซ.รามคำแหง12 --> ซ.พระรามเก้า39 --> aj |
| | Path 16 : aj --> ซ.พระรามเก้า39 --> u |
| | Path 17 : u --> ซ.พระรามเก้า39 --> bw |
| | Path 18 : bw --> ซ.พระรามเก้า39 --> ถ.พระรามเก้า --> ซ.พระรามเก้า 53 --> ab |
| | Path 19 : ab --> ซ.พระรามเก้า 53 --> ซ.พระรามเก้า 55 --> ซ.พระรามเก้า 57 --> ซ.พระรามเก้า 57/1 --> คลังสินค้า |
| 4 | เที่ยวที่4 == Motorcycle ; az ; ay ; ad ; n ; aw ; ax ; k ; j ; l ; m ; a ; d ; b ; c ; h ; e ; f ; g ; i |
| | Path 1 : คลังสินค้า --> ซ.พระรามเก้า 57/1 --> ซ.พระรามเก้า 57 --> ซ.พระรามเก้า57 --> ซ.รามคำแหง24 --> กลับรถ --> ซ.รามคำแหง26 --> เลี้ยวขวาเข้าถ.รามคำแหง --> ถ |
| | : .รามคำแหง --> กลับรถ --> ถ.รามคำแหง --> ซ.รามคำแหง30/1 --> az |
| | Path 2 : az --> ซ.รามคำแหง30/1 --> ay |
| | Path 3 : ay --> ซ.รามคำแหง30/1 --> ถ.รามคำแหง --> ซ.รามคำแหง30 --> ad |
| | Path 4 : ad --> ซ.รามคำแหง30 --> ถ.รามคำแหง --> ซ.รามคำแหง26/2 --> road --> ซ.รามคำแหง26/1 --> n |
| | Path 5 : n --> ซ.รามคำแหง26/1 --> road --> ซ.รามคำแหง26/2 --> aw |
| | Path 6 : aw --> ซ.รามคำแหง26/2 --> ax |
| | Path 7 : ax --> ซ.รามคำแหง26/2 --> k |
| | Path 8 : k --> ซ.รามคำแหง26/2 --> j |
| | Path 9 : j --> ซ.รามคำแหง26/2 --> road --> ซ.รามคำแหง26/1 --> l |
| | Path 10 : l --> ซ.รามคำแหง26/1 --> m |
| | Path 11 : m --> ซ.รามคำแหง26/1 --> ถ.รามคำแหง --> ซ.รามคำแหง 24/2 --> a |
| | Path 12 : a --> ซ.รามคำแหง 24/2 --> d |
| | Path 13 : d --> ซ.รามคำแหง 24/2 --> b |

| Route No. | Detail of route |
|-----------|---|
| | Path 14 : b --> ซ.รามคำแหง 24/2 --> c |
| | Path 15 : c --> ซ.รามคำแหง 24/2 --> h |
| | Path 16 : h --> ซ.รามคำแหง 24/2 --> e |
| | Path 17 : e --> ซ.รามคำแหง 24/2 --> f |
| | Path 18 : f --> ซ.รามคำแหง 24/2 --> g |
| | Path 19 : g --> ซ.รามคำแหง 24/2 --> i |
| | Path 20 : i --> ซ.รามคำแหง 24/2 --> ซ.รามคำแหง 24 --> กลับรถ --> ซ.พระรามเก้า 57 --> ซ.พระรามเก้า 57 --> ซ.พระรามเก้า 57/1 --> คลังสินค้า |

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

Table A.2: Route details from a run of the routing program with relaxation of time frame
or drop-off time of 1:30 minute

(The first line lists the sequence of the customer names in the route. The following lines
tell how to travel from one customer to the next one in the sequence)

| Route No. | Detail of route |
|-----------|--|
| 1 | เที่ยวที่1 == Motorcycle ; q ; r ; s ; bf ; be ; bb ; t ; bh ; bg ; ar ; bx ; by ; an ; ao ; p ; o ; bl ; ap ; ah ; am ; ab |
| | Path 1 : คลังสินค้า --> ช.พระรามเก้า 57/1 --> ช.พระรามเก้า 57 --> ช.พระรามเก้า57 --> ช.รามคำแหง24 --> ทางเข้าด้านหลังสนามกีฬา --> สนามกีฬา --> สนามรัชม้งคลากีฬาสถาน --> ทางเข้าด้านหลังสนามรัชม้งคลากีฬาสถาน --> ช.รามคำแหง24 --> อ.รามคำแหง --> ช.รามคำแหง18 --> q |
| | Path 2 : q --> ช.รามคำแหง18 --> r |
| | Path 3 : r --> ช.รามคำแหง18 --> road --> ช.รามคำแหง16 --> s |
| | Path 4 : s --> ช.รามคำแหง16 --> bf |
| | Path 5 : bf --> ช.รามคำแหง16 --> be |
| | Path 6 : be --> ช.รามคำแหง16 --> bb |
| | Path 7 : bb --> ช.รามคำแหง16 --> road --> ช.รามคำแหง14 --> t |
| | Path 8 : t --> ช.รามคำแหง14 --> bh |
| | Path 9 : bh --> ช.รามคำแหง14 --> bg |
| | Path 10 : bg --> ช.รามคำแหง14 --> อ.รามคำแหง --> ช.รามคำแหง8 --> ar |
| | Path 11 : ar --> ช.รามคำแหง8 --> ช.พระรามเก้า31 --> bx |
| | Path 12 : bx --> ช.พระรามเก้า31 --> ช.พระรามเก้า31-33 --> by |
| | Path 13 : by --> ช.พระรามเก้า31-33 --> ช.พระรามเก้า33 --> an |
| | Path 14 : an --> ช.พระรามเก้า33 --> ช.รามคำแหง8 --> ช.พระรามเก้า29 --> ao |
| | Path 15 : ao --> ช.พระรามเก้า29 --> อ.พระรามเก้า --> เลี้ยวซ้ายเข้า ช.รามคำแหงแยก2 --> ช.รามคำแหง24แยก2 --> ช.รามคำแหง24 --> อ.รามคำแหง --> ช.รามคำแหง22 --> p |
| | Path 16 : p --> ช.รามคำแหง22 --> o |
| | Path 17 : o --> ช.รามคำแหง22 --> อ.รามคำแหง --> ช.รามคำแหง8 --> ช.พระรามเก้า |

| Route No. | Detail of route |
|-----------|---|
| | 29 --> bl |
| | Path 18 : bl --> ซ.พระรามเก้า29 --> ap |
| | Path 19 : ap --> ซ.พระรามเก้า29 --> ซ.รามคำแหง8 --> ซ.พระรามเก้า35 --> ah |
| | Path 20 : ah --> ซ.พระรามเก้า35 --> ซ.พระรามเก้า35-37 --> am |
| | Path 21 : am --> ซ.พระรามเก้า35-37 --> ซ.พระรามเก้า37 --> ถ.พระรามเก้า --> ซ.พระรามเก้า 53 --> ab |
| | Path 22 : ab --> ซ.พระรามเก้า 53 --> ซ.พระรามเก้า 55 --> ซ.พระรามเก้า 57 --> ซ.พระรามเก้า 57/1 --> คลังสินค้า |
| 2 | เที่ยวที่2 == Motorcycle ; aa ; ac ; z ; x ; y ; bv ; bt ; bu ; bs ; au ; av ; br ; bp ; bo ; bq ; bn ; bm ; as ; at ; bd ; bc ; ba ; aq ; bz ; ai ; v ; w ; al ; ak ; bk ; bj ; bi ; aj ; u ; bw |
| | Path 1 : คลังสินค้า --> ซ.พระรามเก้า 57/1 --> ซ.พระรามเก้า 57 --> ซ.พระรามเก้า 55 --> ถ.พระรามเก้า --> กลับรถ --> ถ.พระรามเก้า --> ซ.พระรามเก้า62 --> aa |
| | Path 2 : aa --> ซ.พระรามเก้า62 --> road --> ซ.พระรามเก้า60 --> ac |
| | Path 3 : ac --> ซ.พระรามเก้า60 --> road --> ซ.พระรามเก้า58 --> road --> z |
| | Path 4 : z --> road --> ซ.พระรามเก้า54 --> x |
| | Path 5 : x --> ซ.พระรามเก้า54 --> y |
| | Path 6 : y --> ซ.พระรามเก้า54 --> ถ.พระรามเก้า --> เลี้ยวซ้ายเข้า ถ.รามคำแหง --> ถ.รามคำแหง --> รามคำแหง4 --> bv |
| | Path 7 : bv --> รามคำแหง4 --> รามคำแหง4 ซอย7 --> bt |
| | Path 8 : bt --> รามคำแหง4 ซอย7 --> ทางระหว่างซอย7 กับซอย8 --> รามคำแหง4 ซอย8 --> bu |
| | Path 9 : bu --> รามคำแหง4 ซอย8 --> ทางระหว่างซอย7 กับซอย8 --> ทางระหว่างซอย6 กับซอย7 --> bs |
| | Path 10 : bs --> ทางระหว่างซอย6 กับซอย7 --> ทางระหว่างซอย5 กับซอย6 --> รามคำแหง4 ซอย5 --> au |
| | Path 11 : au --> รามคำแหง4 ซอย5 --> รามคำแหง4 --> รามคำแหง4-รามคำแหง2 --> |

| Route No. | Detail of route |
|-----------|--|
| | ซ.รามคำแหง2 --> av |
| | Path 12 : av --> ซ.รามคำแหง2 --> br |
| | Path 13 : br --> ซ.รามคำแหง2 --> ถ.รามคำแหง --> จุดกลับรถหน้าการไฟฟ้า --> ถ.รามคำแหง --> ซ.รามคำแหง1 --> bp |
| | Path 14 : bp --> ซ.รามคำแหง1 --> bo |
| | Path 15 : bo --> ซ.รามคำแหง1 --> ถ.รามคำแหง --> ซ.รามคำแหง3/1 --> bq |
| | Path 16 : bq --> ซ.รามคำแหง3/1 --> road --> ถ.รามคำแหง --> ซ.รามคำแหง5 --> ซ.รามคำแหง5 แยก1 --> bn |
| | Path 17 : bn --> ซ.รามคำแหง5 แยก1 --> bm |
| | Path 18 : bm --> ซ.รามคำแหง5 แยก1 --> ซ.รามคำแหง5 --> as |
| | Path 19 : as --> ซ.รามคำแหง5 --> ถ.รามคำแหง --> ซ.รามคำแหง7 --> at |
| | Path 20 : at --> ซ.รามคำแหง7 --> ถ.รามคำแหง --> กลับรถหน้า ซ.รามคำแหง25 --> ถ.รามคำแหง --> ซ.รามคำแหง18 --> bd |
| | Path 21 : bd --> ซ.รามคำแหง18 --> bc |
| | Path 22 : bc --> ซ.รามคำแหง18 --> ba |
| | Path 23 : ba --> ซ.รามคำแหง18 --> ถ.รามคำแหง --> ซ.รามคำแหง10 --> aq |
| | Path 24 : aq --> ซ.รามคำแหง10 --> ถ.รามคำแหง --> ซ.รามคำแหง8 --> ซ.พระรามเกล้า 35 --> bz |
| | Path 25 : bz --> ซ.พระรามเกล้า35 --> ซ.พระรามเกล้า35-37 --> ai |
| | Path 26 : ai --> ซ.พระรามเกล้า35-37 --> ซ.พระรามเกล้า37 --> v |
| | Path 27 : v --> ซ.พระรามเกล้า37 --> w |
| | Path 28 : w --> ซ.พระรามเกล้า37 --> ซ.พระรามเกล้า37-39 --> ซ.พระรามเกล้า39 --> ซ.รามคำแหง12 --> ซ.อุดมยศ4 --> al |
| | Path 29 : al --> ซ.อุดมยศ4 --> ak |
| | Path 30 : ak --> ซ.อุดมยศ4 --> ซ.รามคำแหง12 --> bk |
| | Path 31 : bk --> ซ.รามคำแหง12 --> bj |

| Route No. | Detail of route |
|-----------|--|
| | Path 32 : bj --> ซ.รามคำแหง12 --> bi |
| | Path 33 : bi --> ซ.รามคำแหง12 --> ซ.พระรามเก้า39 --> aj |
| | Path 34 : aj --> ซ.พระรามเก้า39 --> u |
| | Path 35 : u --> ซ.พระรามเก้า39 --> bw |
| | Path 36 : bw --> ซ.พระรามเก้า39 --> ถ.พระรามเก้า --> ซ.พระรามเก้า 53 --> ซ.พระรามเก้า 55 --> ซ.พระรามเก้า 57 --> ซ.พระรามเก้า 57/1 --> คลังสินค้า |
| 3 | เที่ยวที่3 == Motorcycle ; ag ; ae ; af ; az ; ay ; ad ; n ; aw ; ax ; k ; j ; l ; m ; a ; d ; b ; c ; h ; e ; f ; g ; i |
| | Path 1 : คลังสินค้า --> ซ.พระรามเก้า 57/1 --> ซ.พระรามเก้า 57 --> ซ.พระรามเก้า57 --> ซ.รามคำแหง24 --> กลับรถหน้า ABAC --> ซ.รามคำแหง24 --> ซ.รามคำแหง24/1 --> ag |
| | Path 2 : ag --> ซ.รามคำแหง24/1 --> ae |
| | Path 3 : ae --> ซ.รามคำแหง24/1 --> af |
| | Path 4 : af --> ซ.รามคำแหง24/1 --> ถ.รามคำแหง --> กลับรถหน้า สน.หัวหมาก --> ถ.รามคำแหง --> กลับรถ --> ถ.รามคำแหง --> ซ.รามคำแหง30/1 --> az |
| | Path 5 : az --> ซ.รามคำแหง30/1 --> ay |
| | Path 6 : ay --> ซ.รามคำแหง30/1 --> ถ.รามคำแหง --> ซ.รามคำแหง30 --> ad |
| | Path 7 : ad --> ซ.รามคำแหง30 --> ถ.รามคำแหง --> ซ.รามคำแหง26/2 --> road --> ซ.รามคำแหง26/1 --> n |
| | Path 8 : n --> ซ.รามคำแหง26/1 --> road --> ซ.รามคำแหง26/2 --> aw |
| | Path 9 : aw --> ซ.รามคำแหง26/2 --> ax |
| | Path 10 : ax --> ซ.รามคำแหง26/2 --> k |
| | Path 11 : k --> ซ.รามคำแหง26/2 --> j |
| | Path 12 : j --> ซ.รามคำแหง26/2 --> road --> ซ.รามคำแหง26/1 --> l |
| | Path 13 : l --> ซ.รามคำแหง26/1 --> m |
| | Path 14 : m --> ซ.รามคำแหง26/1 --> ถ.รามคำแหง --> ซ.รามคำแหง 24/2 --> a |

| Route No. | Detail of route |
|-----------|---|
| | Path 15 : a --> ซ.รามคำแหง 24/2 --> d |
| | Path 16 : d --> ซ.รามคำแหง 24/2 --> b |
| | Path 17 : b --> ซ.รามคำแหง 24/2 --> c |
| | Path 18 : c --> ซ.รามคำแหง 24/2 --> h |
| | Path 19 : h --> ซ.รามคำแหง 24/2 --> e |
| | Path 20 : e --> ซ.รามคำแหง 24/2 --> f |
| | Path 21 : f --> ซ.รามคำแหง 24/2 --> g |
| | Path 22 : g --> ซ.รามคำแหง 24/2 --> i |
| | Path 23 : i --> ซ.รามคำแหง 24/2 --> ซ.รามคำแหง 24 --> กลับริด --> ซ.พระรามเกล้า 57 - -> ซ.พระรามเกล้า 57 --> ซ.พระรามเกล้า 57/1 --> คลังสินค้า |

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

Table A.3: Route details from a run of the routing program with drop-off time of 1:45 minute

(The first line lists the sequence of the customer names in the route. The following lines tell how to travel from one customer to the next one in the sequence)

| Route No. | Detail of route |
|-----------|--|
| 1 | เที่ยวที่1 == Motorcycle ; q ; r ; s ; bf ; be ; bb ; t ; bh ; bg ; ar ; bx ; by ; an ; ao ; p ; o ; bl ; ap ; ah ; am ; aa ; ac ; z ; x ; y ; ab |
| | Path 1 : คลังสินค้า --> ช.พระรามเก้า 57/1 --> ช.พระรามเก้า 57 --> ช.พระรามเก้า57 --> ช.รามคำแหง24 --> ทางเข้าด้านหลังสนามกีฬา --> สนามกีฬา --> สนามรัชมิ่งคลาภิฬาสถาน --> ทางเข้าด้านหลังสนามรัชมิ่งคลาภิฬาสถาน --> ช.รามคำแหง24 --> ถ.รามคำแหง --> ช.รามคำแหง18 --> q |
| | Path 2 : q --> ช.รามคำแหง18 --> r |
| | Path 3 : r --> ช.รามคำแหง18 --> road --> ช.รามคำแหง16 --> s |
| | Path 4 : s --> ช.รามคำแหง16 --> bf |
| | Path 5 : bf --> ช.รามคำแหง16 --> be |
| | Path 6 : be --> ช.รามคำแหง16 --> bb |
| | Path 7 : bb --> ช.รามคำแหง16 --> road --> ช.รามคำแหง14 --> t |
| | Path 8 : t --> ช.รามคำแหง14 --> bh |
| | Path 9 : bh --> ช.รามคำแหง14 --> bg |
| | Path 10 : bg --> ช.รามคำแหง14 --> ถ.รามคำแหง --> ช.รามคำแหง8 --> ar |
| | Path 11 : ar --> ช.รามคำแหง8 --> ช.พระรามเก้า31 --> bx |
| | Path 12 : bx --> ช.พระรามเก้า31 --> ช.พระรามเก้า31-33 --> by |
| | Path 13 : by --> ช.พระรามเก้า31-33 --> ช.พระรามเก้า33 --> an |
| | Path 14 : an --> ช.พระรามเก้า33 --> ช.รามคำแหง8 --> ช.พระรามเก้า29 --> ao |
| | Path 15 : ao --> ช.พระรามเก้า29 --> ถ.พระรามเก้า --> เลี้ยวซ้ายเข้า ช.รามคำแหงแยก2 --> ช.รามคำแหง24แยก2 --> ช.รามคำแหง24 --> ถ.รามคำแหง --> ช.รามคำแหง22 --> p |
| | Path 16 : p --> ช.รามคำแหง22 --> o |
| | Path 17 : o --> ช.รามคำแหง22 --> ถ.รามคำแหง --> ช.รามคำแหง8 --> ช.พระรามเก้า |

| Route No. | Detail of route |
|-----------|--|
| | 29 --> bl |
| | Path 18 : bl --> ซ.พระรามเก้า29 --> ap |
| | Path 19 : ap --> ซ.พระรามเก้า29 --> ซ.รามคำแหง8 --> ซ.พระรามเก้า35 --> ah |
| | Path 20 : ah --> ซ.พระรามเก้า35 --> ซ.พระรามเก้า35-37 --> am |
| | Path 21 : am --> ซ.พระรามเก้า35-37 --> ซ.พระรามเก้า37 --> ถ.พระรามเก้า --> กลับรถ --> ถ.พระรามเก้า --> ซ.พระรามเก้า62 --> aa |
| | Path 22 : aa --> ซ.พระรามเก้า62 --> road --> ซ.พระรามเก้า60 --> ac |
| | Path 23 : ac --> ซ.พระรามเก้า60 --> road --> ซ.พระรามเก้า58 --> road --> z |
| | Path 24 : z --> road --> ซ.พระรามเก้า54 --> x |
| | Path 25 : x --> ซ.พระรามเก้า54 --> y |
| | Path 26 : y --> ซ.พระรามเก้า54 --> ถ.พระรามเก้า --> กลับรถ --> ถ.พระรามเก้า --> ซ.พระรามเก้า 53 --> ab |
| | Path 27 : ab --> ซ.พระรามเก้า 53 --> ซ.พระรามเก้า 55 --> ซ.พระรามเก้า 57 --> ซ.พระรามเก้า 57/1 --> คลังสินค้า |
| 2 | เที่ยวที่2 == Motorcycle ; ag ; ae ; af ; az ; ay ; ad ; n ; aw ; ax ; k ; j ; l ; m ; a ; d ; b ; c ; h ; e ; f ; g ; i |
| | Path 1 : คลังสินค้า --> ซ.พระรามเก้า 57/1 --> ซ.พระรามเก้า 57 --> ซ.พระรามเก้า57 --> ซ.รามคำแหง24 --> กลับรถหน้า ABAC --> ซ.รามคำแหง24 --> ซ.รามคำแหง24/1 --> ag |
| | Path 2 : ag --> ซ.รามคำแหง24/1 --> ae |
| | Path 3 : ae --> ซ.รามคำแหง24/1 --> af |
| | Path 4 : af --> ซ.รามคำแหง24/1 --> ถ.รามคำแหง --> กลับรถหน้า สน.หัวหมาก --> ถ.รามคำแหง --> กลับรถ --> ถ.รามคำแหง --> ซ.รามคำแหง30/1 --> az |
| | Path 5 : az --> ซ.รามคำแหง30/1 --> ay |
| | Path 6 : ay --> ซ.รามคำแหง30/1 --> ถ.รามคำแหง --> ซ.รามคำแหง30 --> ad |
| | Path 7 : ad --> ซ.รามคำแหง30 --> ถ.รามคำแหง --> ซ.รามคำแหง26/2 --> road --> ซ.รามคำแหง26/1 --> n |

| Route No. | Detail of route |
|-----------|---|
| | Path 8 : n --> ซ.รามคำแหง26/1 --> road --> ซ.รามคำแหง26/2 --> aw |
| | Path 9 : aw --> ซ.รามคำแหง26/2 --> ax |
| | Path 10 : ax --> ซ.รามคำแหง26/2 --> k |
| | Path 11 : k --> ซ.รามคำแหง26/2 --> j |
| | Path 12 : j --> ซ.รามคำแหง26/2 --> road --> ซ.รามคำแหง26/1 --> l |
| | Path 13 : l --> ซ.รามคำแหง26/1 --> m |
| | Path 14 : m --> ซ.รามคำแหง26/1 --> ถ.รามคำแหง --> ซ.รามคำแหง 24/2 --> a |
| | Path 15 : a --> ซ.รามคำแหง 24/2 --> d |
| | Path 16 : d --> ซ.รามคำแหง 24/2 --> b |
| | Path 17 : b --> ซ.รามคำแหง 24/2 --> c |
| | Path 18 : c --> ซ.รามคำแหง 24/2 --> h |
| | Path 19 : h --> ซ.รามคำแหง 24/2 --> e |
| | Path 20 : e --> ซ.รามคำแหง 24/2 --> f |
| | Path 21 : f --> ซ.รามคำแหง 24/2 --> g |
| | Path 22 : g --> ซ.รามคำแหง 24/2 --> i |
| | Path 23 : i --> ซ.รามคำแหง 24/2 --> ซ.รามคำแหง24 --> กลับรถ --> ซ.พระรามเกล้า57 --> ซ.พระรามเกล้า 57 --> ซ.พระรามเกล้า 57/1 --> คลังสินค้า |
| 3 | เที่ยวที่3 == Motorcycle ; bv ; bt ; bu ; bs ; au ; av ; br ; bp ; bo ; bq ; bn ; bm ; as ; at ; bd ; bc ; ba ; aq ; bz ; ai ; v ; w ; al ; ak ; bk ; bj ; bi ; aj ; u ; bw |
| | Path 1 : คลังสินค้า --> ซ.พระรามเกล้า 57/1 --> ซ.พระรามเกล้า 57 --> ซ.พระรามเกล้า 55 --> ถ.พระรามเกล้า --> กลับรถ --> ถ.พระรามเกล้า --> เลี้ยวซ้ายเข้า ถ.รามคำแหง --> ถ.รามคำแหง --> รามคำแหง4 --> bv |
| | Path 2 : bv --> รามคำแหง4 --> รามคำแหง4 ซอย7 --> bt |
| | Path 3 : bt --> รามคำแหง4 ซอย7 --> ทางระหว่างซอย7 กับซอย8 --> รามคำแหง4 ซอย8 --> bu |
| | Path 4 : bu --> รามคำแหง4 ซอย8 --> ทางระหว่างซอย7 กับซอย8 --> ทางระหว่างซอย |

| Route No. | Detail of route |
|-----------|---|
| | 6 กีบซอย7 --> bs |
| | Path 5 : bs --> ทางระหว่างซอย6 กีบซอย7 --> ทางระหว่างซอย5 กีบซอย6 --> รามคำแหง4 ซอย5 --> au |
| | Path 6 : au --> รามคำแหง4 ซอย5 --> รามคำแหง4 --> รามคำแหง4-รามคำแหง2 --> ซ. รามคำแหง2 --> av |
| | Path 7 : av --> ซ.รามคำแหง2 --> br |
| | Path 8 : br --> ซ.รามคำแหง2 --> ถ.รามคำแหง --> จุดกลับรถหน้าการไฟฟ้า --> ถ. รามคำแหง --> ซ.รามคำแหง1 --> bp |
| | Path 9 : bp --> ซ.รามคำแหง1 --> bo |
| | Path 10 : bo --> ซ.รามคำแหง1 --> ถ.รามคำแหง --> ซ.รามคำแหง3/1 --> bq |
| | Path 11 : bq --> ซ.รามคำแหง3/1 --> road --> ถ.รามคำแหง --> ซ.รามคำแหง5 --> ซ. รามคำแหง5 แยก1 --> bn |
| | Path 12 : bn --> ซ.รมคำแหง5 แยก1 --> bm |
| | Path 13 : bm --> ซ.รมคำแหง5 แยก1 --> ซ.รามคำแหง5 --> as |
| | Path 14 : as --> ซ.รามคำแหง5 --> ถ.รามคำแหง --> ซ.รามคำแหง7 --> at |
| | Path 15 : at --> ซ.รามคำแหง7 --> ถ.รามคำแหง --> กลับรถหน้า ซ.รามคำแหง25 --> ถ. รามคำแหง --> ซ.รามคำแหง18 --> bd |
| | Path 16 : bd --> ซ.รามคำแหง18 --> bc |
| | Path 17 : bc --> ซ.รามคำแหง18 --> ba |
| | Path 18 : ba --> ซ.รามคำแหง18 --> ถ.รามคำแหง --> ซ.รามคำแหง10 --> aq |
| | Path 19 : aq --> ซ.รามคำแหง10 --> ถ.รามคำแหง --> ซ.รามคำแหง8 --> ซ.พระรามเกล้า 35 --> bz |
| | Path 20 : bz --> ซ.พระรามเกล้า35 --> ซ.พระรามเกล้า35-37 --> ai |
| | Path 21 : ai --> ซ.พระรามเกล้า35-37 --> ซ.พระรามเกล้า37 --> v |
| | Path 22 : v --> ซ.พระรามเกล้า37 --> w |
| | Path 23 : w --> ซ.พระรามเกล้า37 --> ซ.พระรามเกล้า37-39 --> ซ.พระรามเกล้า39 --> ซ. รามคำแหง12 --> ซ.อุดมยศ4 --> al |

| Route No. | Detail of route |
|-----------|---|
| | Path 24 : al --> ซ.อุดมยศ4 --> ak |
| | Path 25 : ak --> ซ.อุดมยศ4 --> ซ.รามคำแหง12 --> bk |
| | Path 26 : bk --> ซ.รามคำแหง12 --> bj |
| | Path 27 : bj --> ซ.รามคำแหง12 --> bi |
| | Path 28 : bi --> ซ.รามคำแหง12 --> ซ.พระรามเก้า39 --> aj |
| | Path 29 : aj --> ซ.พระรามเก้า39 --> u |
| | Path 30 : u --> ซ.พระรามเก้า39 --> bw |
| | Path 31 : bw --> ซ.พระรามเก้า39 --> ถ.พระรามเก้า --> ซ.พระรามเก้า 53 --> ซ.พระรามเก้า 55 --> ซ.พระรามเก้า 57 --> ซ.พระรามเก้า 57/1 --> คลังสินค้า |

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

BIOGRAPHY

Mr. Thanat Suensilpong was born on 27th of March 1979 in Bangkok, Thailand. He obtained a Bachelor degree of Engineering in Computer Engineering from Chulalongkorn University, Thailand in 2000. He has been working at Pen K Intertrading Co., Ltd. in the position of engineer since 2001. In 2004, he enrolled as a part-time student working toward a Master's degree in engineering management at the Regional Centre for Manufacturing systems Engineering, Chulalongkorn University.



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย