**ชื่อเรื่องภาษาไทย**

นาย ชื่อ นามสกุล รหัสประจำตัวนิสิต ภาควิชา ........................

นาย ชื่อ นามสกุล รหัสประจำตัวนิสิต ภาควิชา ........................

นาย ชื่อ นามสกุล รหัสประจำตัวนิสิต ภาควิชา ........................

(โครงร่าง) ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาวิชาโครงงานทางวิศวกรรม (ข้ามสาขาวิชา)

ตามหลักสูตรวิศวกรรมศาสตรบัณฑิต

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2560

**ชื่อเรื่องภาษาอังกฤษ**

Mr. Name Surname Student ID Department …………………

Mr. Name Surname Student ID Department …………………

Mr. Name Surname Student ID Department …………………

A Senior Project Submitted in Partial Fulfillment of the Requirements

For the Degree of Bachelor of Engineering Program

Faculty of Engineering Chulalongkorn University

Academic Year 2017

**หัวข้อโครงงาน** ………………………………………………………………………………………………………

**โดย**  นาย/นางสาว ……………………….. …………...............

นาย/นางสาว ……………………….. ……………………….

**อาจารย์ที่ปรึกษา** ผู้ช่วยศาสตราจารย์ ดร.……………… ………………………..

# **บทคัดย่อ**

…………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………..……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………….

คำสำคัญ: ……………; …………………; ………………… (ใส่ key word ไม่เกิน 5 คำ)

**Title** ………………………………………………………………………………………………………

**Students** ……………………….. …………...............

……………………….. ……………………….

**Advisor**  Assistant Professor ……………… ……………………….., Ph.D.

# **Abstract**

…………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………..……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………….

Key words: ……………; …………………; ………………… (ใส่ key word ไม่เกิน 5 คำ)

# **กิตติกรรมประกาศ**

…………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………..……………………………………………………………………………………………………………………………………………………..……..…………………………………………………………………………………………………………………………………………………………..

สารบัญ **(เป็นเพียงตัวอย่าง ให้นิสิตปรับเนื้อหาตามสมควร)**

หน้า

บทคัดย่อ. ก

Abstract ข

กิตติกรรมประกาศ. ค

สารบัญ.. ง

สารบัญตาราง ช

สารบัญรูป. ฉ

บทที่ 1 บทนำ. 1

1.1 ที่มาและความสำคัญ.. 1

1.2 วัตถุประสงค์. 1

1.3 ขอบเขตการวิจัย. 1

1.4 ประโยชน์ที่คาดว่าจะได้รับ. 1

บทที่ 2 วรรณกรรมที่เกี่ยวข้อง 2

2.1 การจัดหน้ากระดาษ. 2

2.2 คำแนะนำการเขียน. 2

2.2.1 ขนาดตัวอักษร. 2

2.2.2 รูปภาพ. 2

2.2.3 ตาราง 3

2.2.4 สมการ. 3

บทที่ 3 ระเบียบวิธีวิจัย. 5

3.1 .......................... 5

3.2 .......................... 5

3.2.1 .......................... 5

บทที่ 4 ผลการดำเนินงานวิจัย. 6

4.1 ……………………………….. 6

บทที่ 5 สรุปผลการวิจัย อภิปรายผล และข้อเสนอแนะ. 7

เอกสารอ้างอิง 8

# **สารบัญตาราง**

|  |  |
| --- | --- |
| ตารางที่ 1.1 ……………………………………………… | 3 |
| ตารางที่ 2.1 ……………………………………………… | 9 |

# **สารบัญรูป**

|  |  |
| --- | --- |
| รูปที่ 2.1 ……………………………………………… | 4 |
| รูปที่ 2.2 ……………………………………………… | 5 |
| รูปที่ 2.3 ……………………………………………… | 7 |

# **บทที่ 1**

# **บทนำ**

## **1.1 ที่มาและความสำคัญ**

Bangkok is full of businesses and companies, and even more companies today due to the booming of startups. Therefore, the demands of the messenger services are high. Moreover, Bangkok is also full of bike taxis (Win-motorcycle, motorcycle depots) which is good for us. However, the existing messenger services have some downsides. For example, the traditional messenger services generate all requests separately (1 job = 1 request). However, in the system we are trying to develop, the requests will be combined to minimize total travel distances (1 job can be many requests from many users. We will separate requests into groups to generate jobs for deliverymen to take, and when all jobs are generated, the deliverymen can choose the nearest jobs to do. This is the advantage of our system because the bikes (deliverymen) can do many requests in one job with fewer distances than existing systems, assuming that the bikes have to return to their depots. For more understanding, please see the pictures below.

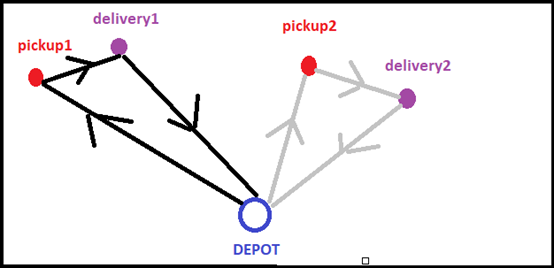


Fig. 1.1 – The existing system. 2 job 2 requests; Red nodes are pickup points; purple nodes are delivery points.

Assume that bikes have to travel to pickup points (red nodes), then to delivery points (blue nodes), then to their depots, which is somewhere near the pickup points (the lowest distances case is that the depots are at pickup points).

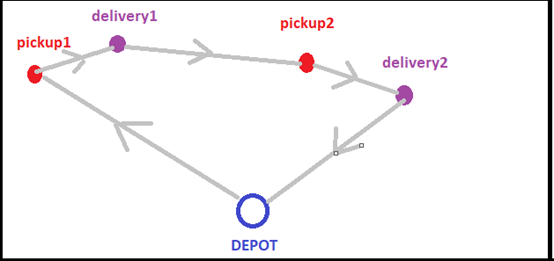


Fig. 1.2 – The new system. 1 job, 2 requests.

The requests are combined into 1 job, the total travel distances are decreased.

## **1.2 วัตถุประสงค์**

## The objective of this project is to find and analyze an algorithm for the messenger service problem. The messenger service is a service which users have requests to send documents, and service providers (deliverymen) pick the documents from users at the pickup points and deliver them to the delivery points. This problem can be classified as a pickup and delivery problem. Given, the requests from users (the pickup points, the delivery points, time windows, etc.), we want to generate jobs (the tours of vehicles) for the service providers (deliverymen) with the lowest costs (travel distances) and still satisfy the constraints. Our objective is to find a way to generate the best jobs.

## **1.3 ขอบเขตการวิจัย**

The scope of this project includes studying research papers, researching and developing our own algorithm for the problem, simulating and running test instances. Also, developing preliminary business model for messenger service including cost and revenue management.

About the problem, we are trying to find the best ways to generate jobs for service providers (deliverymen); each job consists of request(s) from user(s) who want to send documents, given coordinates of places ,traveling distances between places, time windows that users appoint to receive the documents, load capacities of vehicles and load demands of objects to be delivered. We want to generate the jobs with the lowest total travel distances . We also assume that the amount of bikes in Bangkok are high enough so that we do not have to think about constraints of the number of vehicles, and there will always be a bike (deliveryman) who accepts any jobs we created.

After researching, we have found that the messenger business can be categorized into 2 types. First, the business that own the asset. Second, the business that do not. Our group are focusing on the second one as we intend to provide the complete route for any driver (motorcycle) to participate in the business.

## **1.4 ประโยชน์ที่คาดว่าจะได้รับ**

- To develop the algorithms for solving the practical problem.

- To develop the business model for the problem.

- To learn more about algorithms.

- To work as a team.

# **บทที่ 2**

# **วรรณกรรรมที่เกี่ยวข้อง**

นิสิตสามารถเริ่มต้นการเขียนบทความโดยการแทนที่เนื้อหาในเอกสารต้นแบบฉบับนี้ โดยก่อนนิสิตจะเขียนปริญญานิพนธ์ขอให้นิสิตศึกษา “คู่มือการจัดทำปริญญานิพนธ์” ที่ได้จัดทำขึ้นอย่างละเอียด นอกเนื้อจากรายละเอียดใน “คู่มือการจัดทำปริญญานิพนธ์” แล้วรายละเอียดเพิ่มเติมในเรื่องรูปแบบการจัดทำปริญญานิพนธ์ มีดังต่อไปนี้

## **2.1 การจัดหน้ากระดาษ**

ขนาดของบทความจะอยู่ในพื้นที่ของกระดาษ A4 โดยเว้นระยะด้านบนและด้านซ้ายเป็นระยะ 3.81 ซม. และ 2.54 ซม. สำหรับด้านบนและด้านขวา การลำดับหัวข้อในส่วนของเนื้อเรื่อง ให้ใส่เลขกำกับ และหากมีการแบ่งหัวข้อย่อย ให้ใช้เลขระบบทศนิยมกำกับหัวข้อย่อย เช่น 2.1, 2.1.1 เป็นต้น

## **2.2 คำแนะนำการเขียน**

### **2.2.1 ขนาดตัวอักษร**

ตัวอักษรที่ใช้คือ “TH Sarabun New” รายละเอียดตัวอักษรแสดงในตารางที่ 2.1 เนื้อเรื่องในแต่ละบรรทัดให้จัดเรียงชิดซ้ายและขวาอย่างสวยงามโดยตั้งค่า Alignment แบบ Thai Distributed (“Justify”)

### **2.2.2 รูปภาพ**

รูปภาพจะต้องวางไว้ตำแหน่งหน้ากระดาษ โดยรูปภาพทุกรูปจะต้องมีหมายเลขแสดงลำดับและคำบรรยายได้ภาพ ตัวอย่างการจัดวางรูปดังแสดงในรูปที่ 2.1 คำบรรยายใต้ภาพ ห้ามใช้คำว่า “แสดง” เช่น ห้ามเขียนว่า” รูปที่ 1 แสดงความสัมพันธ์...” ที่ถูกต้องควรเป็น “รูปที่ 1 ความสัมพันธ์ระหว่าง...”

**รูปที่ 2.1** ภาพถ่ายขยายกำลังสูงของปูนซีเมนต์ปอร์ตแลนด์

## **2.2.3 ตาราง**

ตารางทุกตารางจะต้องมีหมายเลขและคำบรรยายกำกับเหนือตารางดังแสดงใน ตารางที่ 2.1 ในคำบรรยายเหนือตารางห้ามใช้คำว่า “แสดง” เช่นเดียวกับกรณีรูปภาพ ตารางควรจะอยู่หน้าเดียวทั้งตาราง ไม่ควรคร่อมระหว่างหน้า

**ตารางที่ 2.1** สรุปรายละเอียดรูปแบบตัวอักษร

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| หัวข้อ | ลักษณะ | ตัวอักษร | รูปแบบ | ขนาด (จุด) |
| ชื่อบท (1,2,..) | Heading1 | TH Sarabun New | หนา | 18 |
| หัวเรื่อง 2 (1.1,1.2,..) | Heading2 | TH Sarabun New | หนา | 16 |
| หัวเรื่อง 3 (1.1.1, …) | Heading3 | TH Sarabun New | หนา | 16 |
| เนื้อหา | Normal | TH Sarabun New | ธรรมดา | 16 |
| ตัวแปรในสมการ \*\* | N/A | Times New Roman | เอียง | 12 |
| เอกสารอ้างอิง | Reference\_new | TH Sarabun New | ธรรมดา | 16 |

\*\* จัดทำโดยการใช้ MathType/ Ms Equation Object.

### **2.2.4 สมการ**

สมการที่ใช้ในปริญญานิพนธ์ควรจะเป็นการสร้างจากโปรแกรม MathType หรือเป็นวัตถุของ Microsoft Equation มีขนาด 12 จุด และเป็นตัวอักษร“Times New Roman” ขนาด 12 จุด สมการทุกสมการจะต้องมีหมายเลขกำกับอยู่ภายในวงเล็บ และเรียงลำดับที่ถูกต้อง ตำแหน่งของหมายเลขสมการ (ใช้ตัวอักษร TH Sarabun New ธรรมดาขนาด 16 จุด) จะต้องอยู่ชิดขอบด้านขวาของหน้า ตำแหน่งของสมการให้จัดตามความสวยงาม ดังแสดงในสมการที่ (2.1)

(2.1)

โดยการอธิบายตัวแปรที่ระบุในสมการ ให้ใช้ตัวอักษร Times New Roman ตัวอักษรเอียง ขนาด 12 จุด ตัวอย่างเช่น *c* หมายถึงจำนวนช่างไฟฟ้า, *d* คือ จำนวนช่างฝ้า, *e* คือค่าคงที่เท่ากับ 2 และ *f* คือผลลัพธ์ที่ได้

**The vehicle routing problem (VRP) is the problem which aim to minimize the cost of a set of**

**routes for a fleet of vehicle which serves exactly once to a set of customers with known desire. The**

**pickup and delivery problem with time windows (PDPTW) is a further concerned problem which include the optimal route for the pickup and delivery service with a time constraint i.e. time**

**windowed. To solve this sophisticated problem, the special algorithm must be taken into account.**

**Researchers have been doing this problem since early 1900’s but, there are still no exact answer for**

**the problem. It is classified as a NP-hard problem; since the problem can be viewed as a Vehicle**

**Routing Problem (VRP) with precedence constraints (e.g. node i have to be visited before node j),**

**also the VRP’s general well-known case is the Traveling Salesman Problem (TSP. Therefore, many**

**researchers have developed heuristics and meta-heuristics to solve this kind of problems rather than**

**exact methods. There are many variations of the problem that are reviewed and listed below.**

**(note: all articles have the vehicle ‘load capacity’ in their constraints)**

**First, the single vehicle PDP. Xin-Lan Liao, et al. used the Genetic Algorithm (GA) to solve the**

**Minimum Latency PDP (MLPDP)[1] . The objective was to minimize the sum of transportation time**

**between demanders and the corresponding suppliers. The vehicle is capacitated and has LIFO**

**constraints. The study focuses on many-to-many PDP, where a delivery customer can have any**

**source of supply. They create their own techniques called reverse weighting to evaluate fitness, and**

**Edge Aggregation Crossover (EAC). The results show that their EAC give better solutions than the**

**other kinds of crossovers. GA is also used for Dynamic PDP problems (DPDP) (requests are dynamic,**

**optimal solutions can change by those requests). Yamming and his team used GA hybrid with Local**

**Sensitive Hashing (LSH) based local search called the multi-objective memetic algorithm based on**

**request prediction to solve DPDP[2] (Yamming Yan, 2017). They also used Google Maps API and test**

**the efficacy of their algorithm on the real map at Shenzen. Ant Colony Optimization (ACO) are also**

**used. In the paper An Ant System for the Selective Pickup and Delivery Problem[3] (Yu-Wei Chang,**

**2016) (SPDP, not all pickup nodes have to be visited), the researchers found that ACO outperformed**

**GA in terms of the solution quality (the route length).**

**Next, we will review more about the multi-vehicle PDP with one central depot. Margaretha**

**Gansterer et al. used two variations of general variable neighborhood search (GVNS) namely**

**sequantial GVNS (GVNSseq) and self-adaptive GVNS (GVNSsa) and compare them to the algorithm**

**based on Guided Local Search (GLS) to solve multi-vehicle profitable PDP. The objective of the**

**problem was to maximize the profit (revenue - travel cost). The results show that both variations of**

**GVNS outperform GLS regarding to solutions quality but used more computational time[4]**

**(Margaretha Gansterer, 2016). Next, Giselher Pankratz[???] used GA to solve the PDPTW and got the best known results on most of the Li and Lim’s test instances he tested on. Another interesting article is “Nature-inspired Heuristics for the**

**Multiple-Vehicle Selective Pickup and Delivery Problem under Maximum Profit and Incentive Fairness**

**Criteria” (Javier Del Ser e. a.)[5]. The objective of the problem was to maximize profits while giving a**

**fair share of net benefit among the company staffs based on their driving distance. They compare 4**

**meta-heuristics which are GA, Harmony Search, Firefly Algorithm, and ACO. The results showed that**

**ACO solutions outperform all other algorithms in both test instances and practical test instances**

**(Spain).**

**For the PDP with time windows (PDPTW) and handling operations (one depot), branch and**

**price and cut technique was also used[6], but computational time was rather expensive (96 requests**

**in up to 2 hours).**

**Next, for the Multi-Depot and Multi-Vehicle PDP (m-MDPDPTW), E. Ben Alaia and Imen**

**Harbaoui Dridi et al. use GA to solve this problem[7] (E. Ben Alaia, Optimization of the Multi-Depot &**

**Multi-Vehicle Pickup and Delivery Problem with Time Windows using Genetic Algorithm, 2013). They**

**later extend the problem to multi-objective (minimize travel distance, tardiness time, and vehicles**

**number) PDPTW[8] and solve it with GA hybrid with Pareto Dominance Optimization. Later, the biobjective**

**(minimize travel distance and tardiness time) dynamic PDPTW[9] solved with GA hybrid**

**with aggregation method.**

**For the VRP problems, there are articles[10] [11], that show the superiority of ACO over GA in**

**terms of solution qualities. Swarm Optimization was also used to solve PDP and VRP, there is a literature review for this topic.[12] A hybrid algorithm which primarily include the GA was also developed in order to encounter the problem.[13] In this paper, the author further develop a hybrid intelligent method for PDPTW. The hybrid intelligent algorithm transforms the multi-vehicle PDPTW problem into several single-vehicle PDPTW problems by using clustering algorithm, then using genetic algorithm to gain the best solutions of single-vehicle PDPTW problems, arrange them to construct initial solution for tabu search, and at last get the optimal solution by performing tabu search. In addition, a method to gain the minimal number of cluster is provided. So the algorithm guarantees the minimization of fleet. The objective function seeks to minimize total cost. Constraints ensure that each client is visited by exactly one vehicle, the equation of flow between clients and guarantee that the vehicles depart the depot at their starting time and return to the depot at the end of the planning period. Ant colony algorithm alone is not the best choice for solving this complex problem.**

**Elhassania MESSAOUD, Ahmed ELHILALI ALAOUI [14] use a hybridized ant colony to solve the VRP**

**with dynamic customer with concerned traffic factor. In this work, the author focus on the dynamic**

**version of the vehicle routing problem with the traffic factors. The main goal of this paper is to**

**minimize the total cost, by using a hybridized ant colony system algorithm, which takes consider**

**the reception of the new customers, and the alteration of the travel cost between two locations.**

**In order to simulate a form of dynamicity, the notion of a working day is simulated by T seconds.**

**Not all nodes are available at the start of the resolution. At the beginning of the day a tentative**

**tour is created with the available nodes. The working day is divided into many time slices. At each**

**time slice the solution is updated. This strategy allows the author to split up the dynamic problem**

**into many static problems, which can be solved. A different approach consists to restart the**

**algorithm when a node becomes available. This method could have a bad effect on the resolution**

**algorithm, because this later can be stopped before a good solution is found. There are many algorithm that can be useful in a VRPSDP (Vehicle Routing Problem with Simultaneous Delivery and Pick-up) optimization. One of the well-known is TS (Tabu search) algorithm. Jinhui Ge [15] use the improved TS algorithm in solving with several groups of instances. This paper solves the problem with simultaneous delivery and pick-up using the improved TS algorithm on the basis of the traditional TS algorithm. First, constructs the initial solution with the closest inserting algorithm. Second, to improve the initial solution, it uses2-opt to structure neighborhood Reference-set, which is divided into superior RefSetl and inferior RefSet2. Then, two solutions are selected respectively from RefSetl and RefSet2 to constitute a new solution, and form into Disperse and diverse Candidate-set. Finally, The Dynamic Tabu List is set up to make its length and construction change with search process and finally attains the purpose of improving the whole optimization. To test improved TS to solve the property of VRPSDP, the author use several groups of instances from the literature: The first instance used for testing is small-scale instances of 20 customers generated at random. The second instance is large-scale improved instances from Solomon's Benchmark Problems data set. The third instance is the real-life problem given by Min with the instance descriptor MIN 2. Nuo Zhu, Chunfu Shao [16] improved the most common algorithm, GA, in order to lower the convergence speed and improve the validity and feasibility in performing. In this paper, the improved VRPSDP mathematical model is set up on the basis of considering the impact of the amount of vehicles and the total cost of transportation. The model, based on basic genetic algorithm and modified algorithm, is worked out on Matlab program. In order to ensure the effectiveness of the chromosomes in the iterative process, chromosome encoding method of genetic algorithm and genetic operators are devised which are more suitable for solving vehicle routing problem. Simulation experiments are done by using emendatory Solomon R101 illustrative example. The optimal vehicle routing sequences and the optimal objective function values are processed with the two algorithms, and then the results are compared with each other according to the convergence process of the optimal solution. From the experiment, the fluctuation of the basic GA based on the traditional roulette is relatively large, the convergence speed is slow, and the transformation begins to become relatively stable at the 233rd generation. In addition, the**

**value of the optimal objective function obtained by the improved genetic algorithm is more ideal and fit for the need of low transport costs. Therefore, the improved genetic algorithm which has validity and feasibility performs better than the basic genetic algorithm A. L. Jaimes et all [17] studied about relationship between objective which are Number Of Route (R), Travel Distance(D), Travel Time(T), Uncollected Profit(P), Length Of The Longest Route(d) and Duration of the route that ends the latest(t) to reduce cost by using Spearman’ s Correlation Coefficient and €MOEA (Multi Objective Evolution Algorithm) for evaluating the difficulty of MOPDP as more objectives are added.**

**[18] HUANG Wulan have developed their own techniques called Improved Genetic Algorithm**

**(IGALS) to solves Simultaneous Pickups and Deliveries (with time windows) which is more effective than genetic algorithm (GA) by improved the convergence speed by using elite reserved section crossover strategy and improved the local search strategy based on road delay time. they comparisons data between IGALS and Branch & Bound Algorithm by testing with 10, 20 and 30 customers the results show IGALS is better than B&B at computing time and the number of iteration and comparisons data between IGALS and GA testing with 29 customers the results show GA fall into the local optimal solution after iterative 19,400 times and the object value is 2350.48 and IGALS fall into the local optimal solution after iterative 13,500 times and the object value is 2332.12, thus convergence speed of IGALS is faster than GA. Though IGALS quickly jump out of the solution but it obtained a more optimal solution. [19] Studied An Ant System for the Selective Pickup and Delivery Problem (SPDP) The objective is to find the shortest route by relaxes the requirement visiting of all pickup nodes but impose load constrain to prevent impractical situation. To solve SPDP they used max-min ant system(MMAS is Ant Colony Optimization: ACO). The MMAS holds 2 features 1) the length of route constructed by ant is not fixed which corresponding to the number of selected pickup nodes and all delivery nodes. 2) this route can satisfy the constrain of vehicle load. Finally they comparison the route length that create by MMAS and GA, the result show the route length of MMAS is shorter than route length from GA.**

# **บทที่ 3**

# **ระเบียบวิธีวิจัย**

## **3.1 Algorithm Part**

**3.1.1 – Using Genetic Algorithm (GA) to solve the single depot Pickup and Delivery problems with time windows (PDPTW)**

## We chose Genetic Algorithm (GA) to solve this kind of problems. The reason we chose GA is that we want to solve problems in a limit amount of time, and in many papers (read the project proposal for more information) GA can give good solutions with not so much computational times.

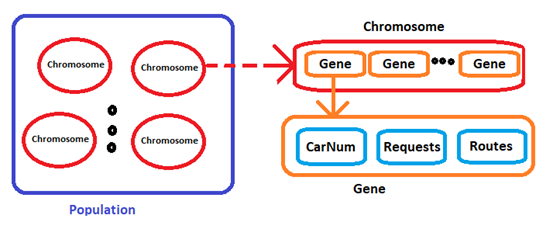
## We found an interesting research paper that we can use[1]. In the paper he used an algorithm called Grouping Genetic Algorithm to solve the PDPTW. The algorithm can be described below.

## Our group had reached the preliminary business model. Next, we aim to elaborate the realistic model.

## GGA for PDPTW (adapted from [1])

## The solutions of the problems are encoded into individuals (chromosomes) represented by an array of genes. A gene represents information of a vehicle’s route, as can bee seen below.

## request = a pair of an index pickup node and the corresponding index delivery node {Requests} = A set of indices of requests. [Route] = An array of a route (permutation of the indices of nodes) Gene = [num, {Requests} , [Route] ] Chromosome = [Gene1, Gene2, Gene3, ... ,Genev] ; v= the number of vehicle used (the number of routes) Populations = An array of chromosomes



**Procedures:**

First, we create populations of feasible chromosomes of size n. Each chromosome has feasible routes for all nodes. After that, we apply GA operations to the populations as can be seen from the pseudocode below.

Generation = 1

While not(termination criteria is met):

evaluate\_fitness(populations)

sort\_by\_fitness(populations)

remove 2 chromosomes with the worst fitness

select the best 2 chromosomes, name them elite1, elite2. These 2 chromosomes will go to the next generation unchanged

randomly select a pair of chromosomes, name them parent1 and parent2, remove them from the poplutions.

child1,child2 <= crossOver(parent1,parent2,prob =1.0)

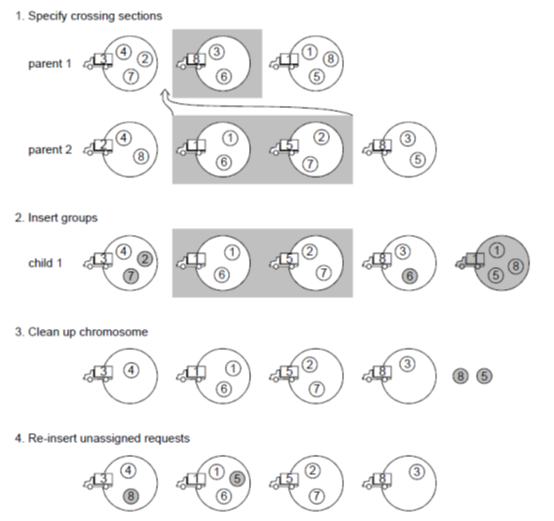
child1,child2<=mutate(child1,prob=0.5),mutate(child2,prob=0.5)

put child1,child2 back into the populations

Generation <= Generation + 1

**GA operations**

Crossover:



(Image from [1])

1.First, we have the chromosomes parent1 and parent2. We randomly specify a crossing section on parent2.

2. Remove the duplicate vehicles on parent1 that also have on part from parent2. Insert the section into parent1, now parent1 became child1.

3.Remove the duplicate requests that already have on child1 and also have on the part from parent2 before the insertion.

4.Insert the remaining requests (see descriptions below) to ensure that all of the requests are served.

5. Repeat the same processes on child2.

**Mutate**:

Randomly select one gene (vehicle) in the chromosome. Remove all requests and their corresponding route. Insert the requests again to the chromosome.

**Inserting requests**:

First, we shuffle the requests to insert so that requests are processed in a random order. Then we insert every request into the chromosome.

To insert a request (p,d) into a tour [a,b,c,d,e,...]; (a,b,c,.. are nodes), we check if the tour is empty, if it is, the request can be inserted without any further calculation.

If the tour is not empty, we insert p into the tour and check whether the new route violate constraints (time windows, load capacities) or not. If the new route is invalid, we just move on to the new position to insert p, else do the similar processes to d. The d has to be inserted after p. If the final tour is invalid, we discard it. Else, we calculate the new distance and subtract them by the old tour distance and call it a “cost”. We do this to every genes (vehicles) in the chromosome and choose the one with minimum cost. If we cannot find a point to insert the request that produce a feasible solution at all, we have to add a new vehicle (gene) into the chromosome to handle the request.

**Test Results**

We adapted and implemented the algorithm and tested it on some of the Li and Lim’s test instances to how it performs. We compared our results to the best known solutions (world records). The results are shown below.

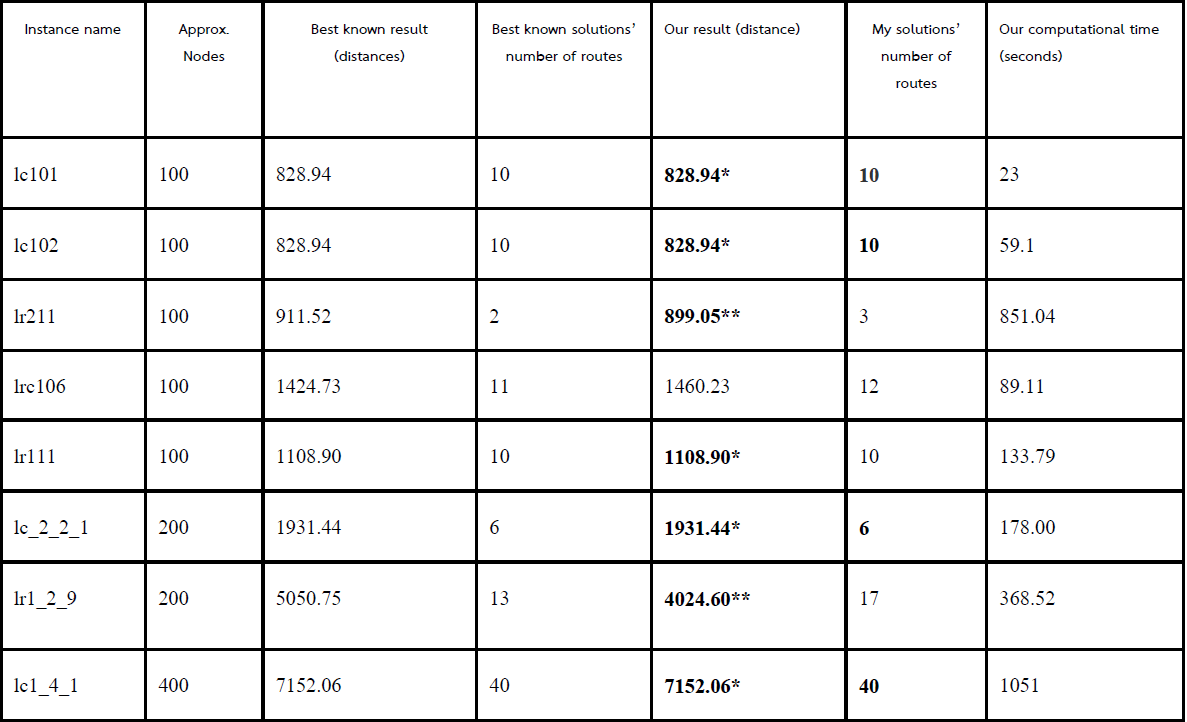
**Parameters for GA**

Generations : 3000, break if qualities of the best solutions in the populations are the same for 500 generations

Crossover rate = 1.0

Mutation rate = 0.5

Elitism (select the best 2 chromosomes and copy them unchanged to the next generation)



**(\*) Results that equal to the the best known solutions**

(\*\*) Results that have less distances than best known solutions. Best known solutions try to minimize number of vehicles first, then try to minimize distances, that’s why my solution has less distances.

The results above are just one time results. For better approximation of algorithm efficiency the results should be recalculated many times and use the averages of them later. As can be seen above, some of the results are equal to the best known solutions. By the way, the qualities of solutions maybe increase if we do more generations of GA’s process, but it will take more time, of course.

Now, we had an algorithm to solve the single-depot problems which worked pretty well. We decided to move on to the multi-depot problems which are more realistic.

**3.1.2 – Finding the best way to assign requests to depots to solve the multi-depot pickup and delivery problems**

The Li and Lim’s test instances are single-depot instances. In order to make them multi-depot, we inserted another 4 depots to each of the instances. We used Li and Lim’s 100 instances which have approximately 100 nodes, all of them have the depot at the coordinate (40,50). We insert the other 4 depots at coordinates (25,25),(25,75),(75,25), and (75,75) respectively. For more understanding, see the pictures below.

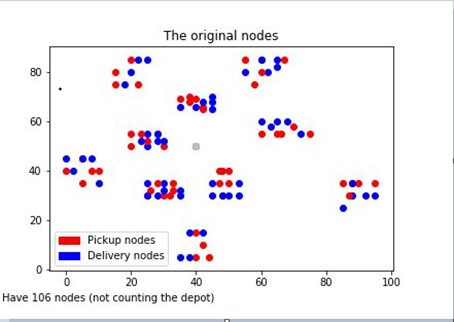


Fig. Original instances (the depot is at the gray dot)

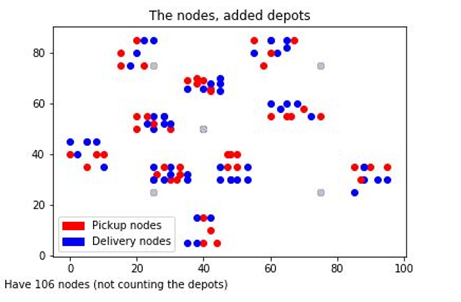


Fig. Added depots instance (the depots are at the gray dots)

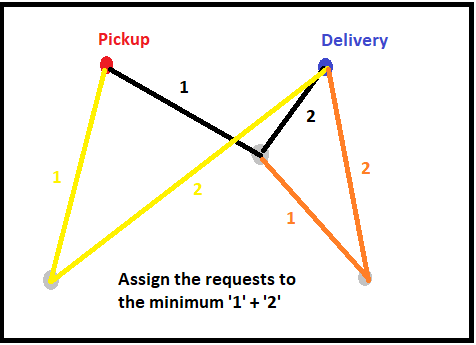
Now, we want to find the best we to assign requests (pickup,delivery) to the depots so that after they are assigned we can treat them as 5 single-depot PDPTWs and we can use the GA above to solve them.

We came up with 4 different ways for assigning requests to the depots.

Given the requests consist of the request (pi,di) where pi is the pickup node, di is the delivery node of the request.

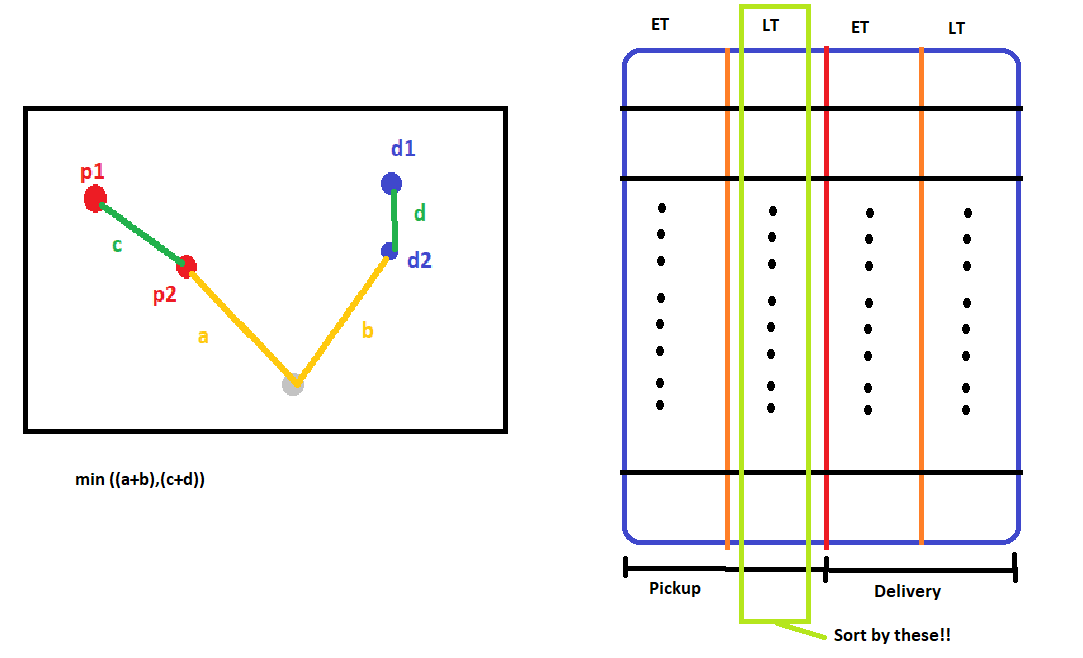
3.1.2.1 – Simple assign

This is the simplest way of assigning depots. Calculate the distances from all depots to pi and from di to all depots, sum them up. Assign the request to the depot that give the minimum distance.



3.1.2.2 – Nearest Routable

First, calculate the simple assigned depots of every requests as described in 3.1.2.1 and store them in a table. Then, sort the requests by the latest time to be visited of the pickup node. This table is the ‘default’ depots for the requests.



After that, loop through the sorted requests (pi,di),set the minimum distance to the distance of pi to the depot +the depot todi ,and check for every other requests (pj,dj) if they can route to each other or not. If yes, calculate the distance (pi ->pj) + (di ->dj) and check if it is less than the minimum, if yes update the depot.

3.1.2.3 - Nearest Routable 3-vote

Do things similar to 3.1.2.2, but instead of finding just one nearest candidate depot, this time find the nearest 3 depots from 3 requests and let them vote. If they cannot vote, use the depot from the request which give the nearest distance.

3.1.2.4 - Nearest Routable vote with condition

Check first if the nearest depot from pickup node and delivery node of the request are the same. If they are, assign the request to that depot. Else, mark the request as “problematic”. After that do things similar to 3.1.2.3 to the problematic requests.

**3.1.3 – Testing the algorithm on the real locations**

After we tested the algorithm on the Li and Lim’s instances, we moved on to the more realistic testings. We created our own 12 test instances from real places in Bangkok using GoogleMap. We call them “Goo” instances with number 1-12.

About the Goo instances, each instances has 80 places to visit, so there is 40 requests per instances. All distances between places were calculated using GoogleMap API. All vehicles are the same and have load capacities of 90. Load demands of requests are randomly generated from {10,20,30,40}. Time windows are randomly generated, and latest time that can be visited of the pickup nodes are at least 1 hour earlier to latest time of the delivery nodes to ensure that the instances can be solved. Service times of places are randomly generated from 1 second to 15 minutes.

About the depots, we use 5 real motorcycle depots in Bangkok which are BTS ChatuChak, Sanam-Pao , CentralRama9 MRT, Soi-SongPrha, and BTS-Siam depots. The distances from depot to places and from places to depots are also calculated using GoogleMap API.

## 

## 

## 

## 

## 

## **3.2 Buisiness Model**

Bangkok, a big city of South East Asia. A great destination of investment. Bangkok has grown rapidly in the last decade. Many foreign companies has branched in Bangkok, willing to expand their business. In this moment, more than 200,000 companies existing in Bangkok (Department of Business Development,Ministry of Commerce of Thailand,2013). The most common thing for running the business is the document, a contact between businesses or within the business. In order to deliver the document, many companies use their own messengers or the external messenger service companies. The moto-taxi is the other choice of choosing, due to the easy accessibility and lower cost. Our model is focusing on document delivery on the motorcycle which is easily found throughout Bangkok. The deliverymen can be anyone not just moto-taxi.

### 3.2.1 Roles in the business

#### 3.2.1.1 Users

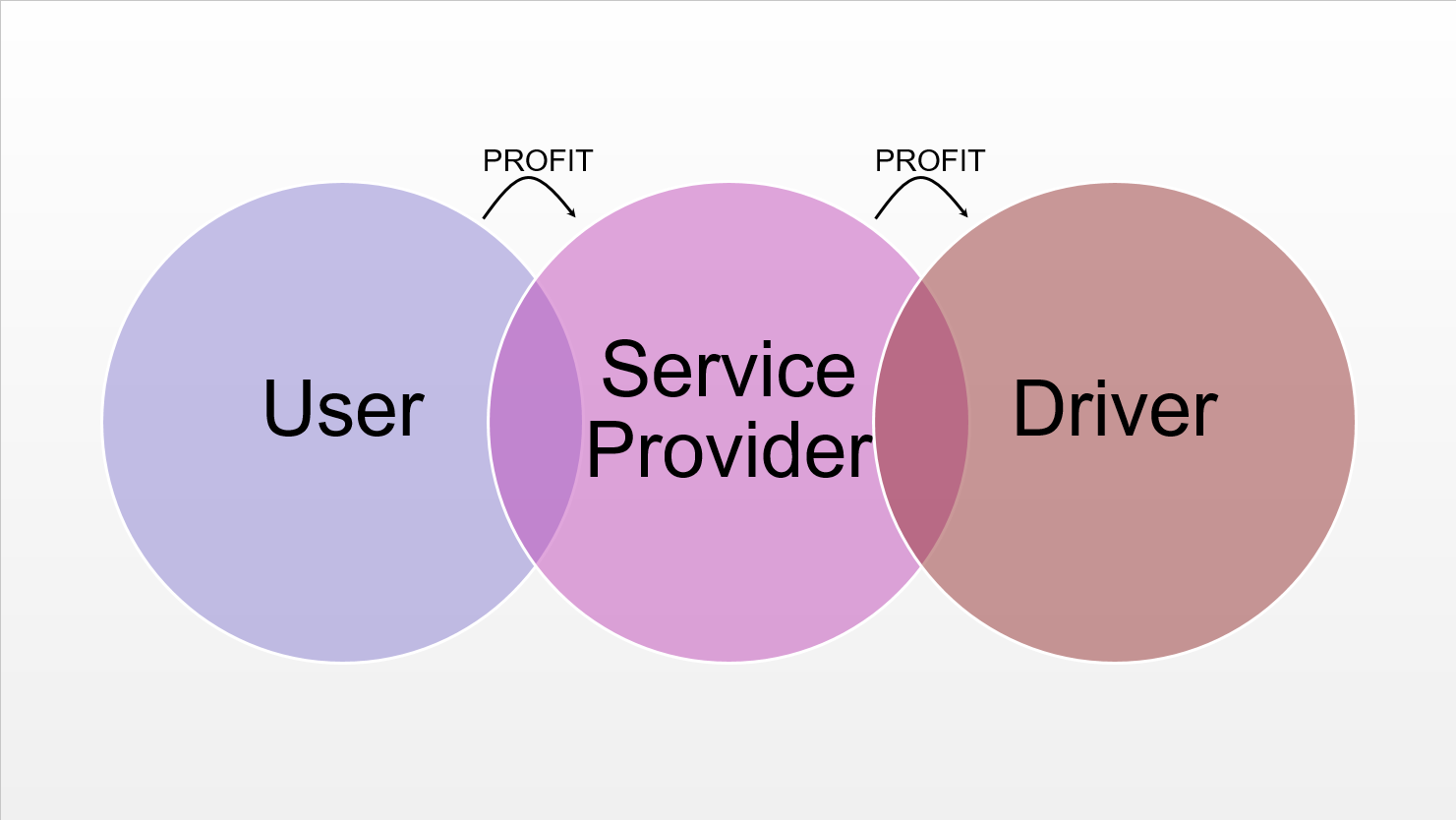
### Users are the person who needs to use the delivery service. Users are informed at first, the service fee, and are able to independently pick the delivery company as their own interest. Users are the source of income of the business, with over priced delivery fee, the service will not be chosen.

#### 3.2.1.2 Service Provider

Service providers are the medium between users and deliverymen, using the service application (website) as a media. Service providers are responsible in providing a complete optimized route and paying the wage to the deliverymen. For users, service providers decide price which is not excessive for users and profitable to the business.

#### 3.2.1.3 Deliverymen

Focusing on distributing documents, deliverymen tend to be anyone who is interested in extra income other than the normal monthly salary. Deliverymen are responsible in the traveling cost which include the gas and the depreciation and maintenance expense. Service provider will pay the wage reasonably to this expense.

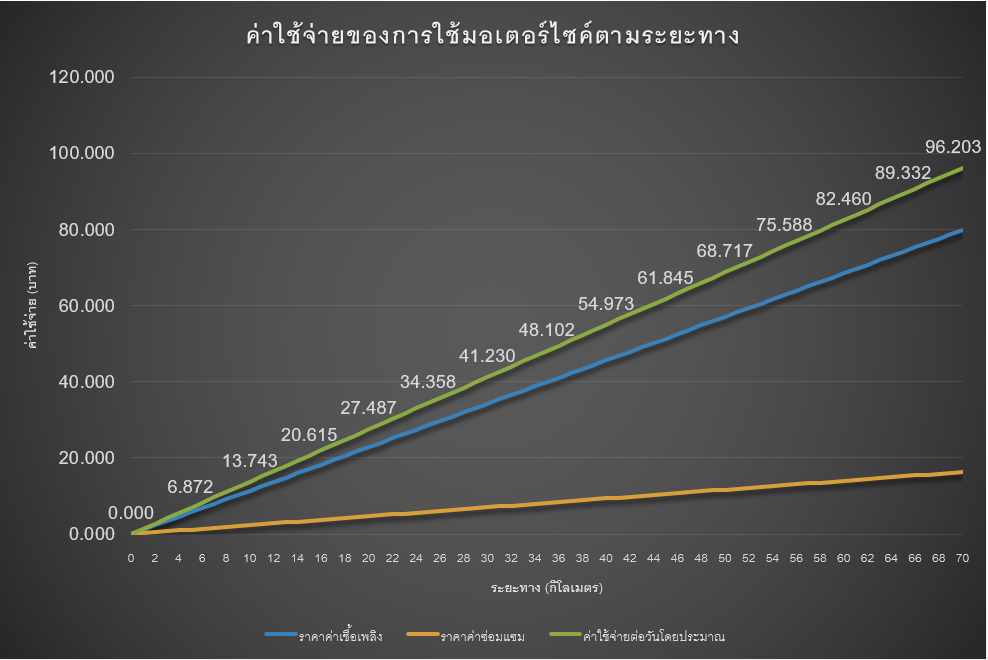


### 3.2.2 Revenue Management

#### 3.2.2.1 Cost

##### 3.2.2.1.1 Deliverymen cost

The delivermen costs include the gas and depreciation expense. Motorcycle which has been used for longer time tends to have more cost than the new one. Motorcycle, which has been used for five years, has a gas consumption rate at 18 Kilometre/Litre (Pollution Control Department of Thailand, 2005) which can be calculated to 1.141 Baht/Kilometre, Based on Thailand gas price 2018. On the other hand, the new motorcycle has a gas consumption rate at 50 Kilometre/Litre (Pollution Control Department of Thailand, 2005) which can be calculated to 0.507 Baht/Kilometre, Based on Thailand gas price 2018. The the depreciation and maintenance rate of the motorcycle is 200 Baht/Month approximated (Pollution Control Department of Thailand, 2005).

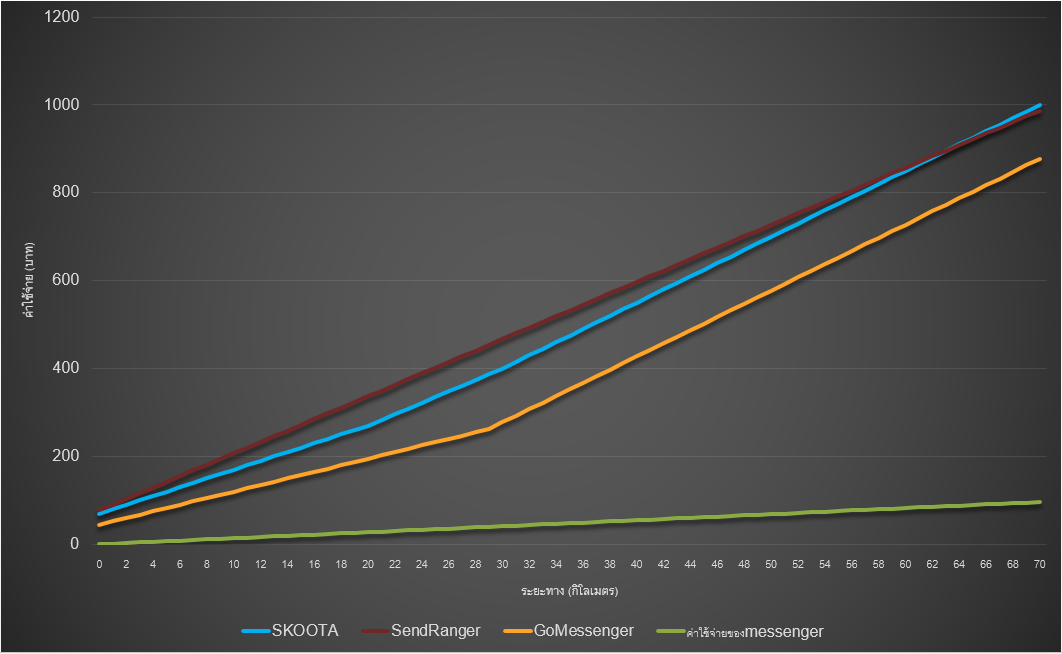


#### 3.2.2.2 Pricing

The pricing is a difficult decision. In order to design the appropriate price, our group have surveyed the delivery market pricing which contain couple of delivery businesses.



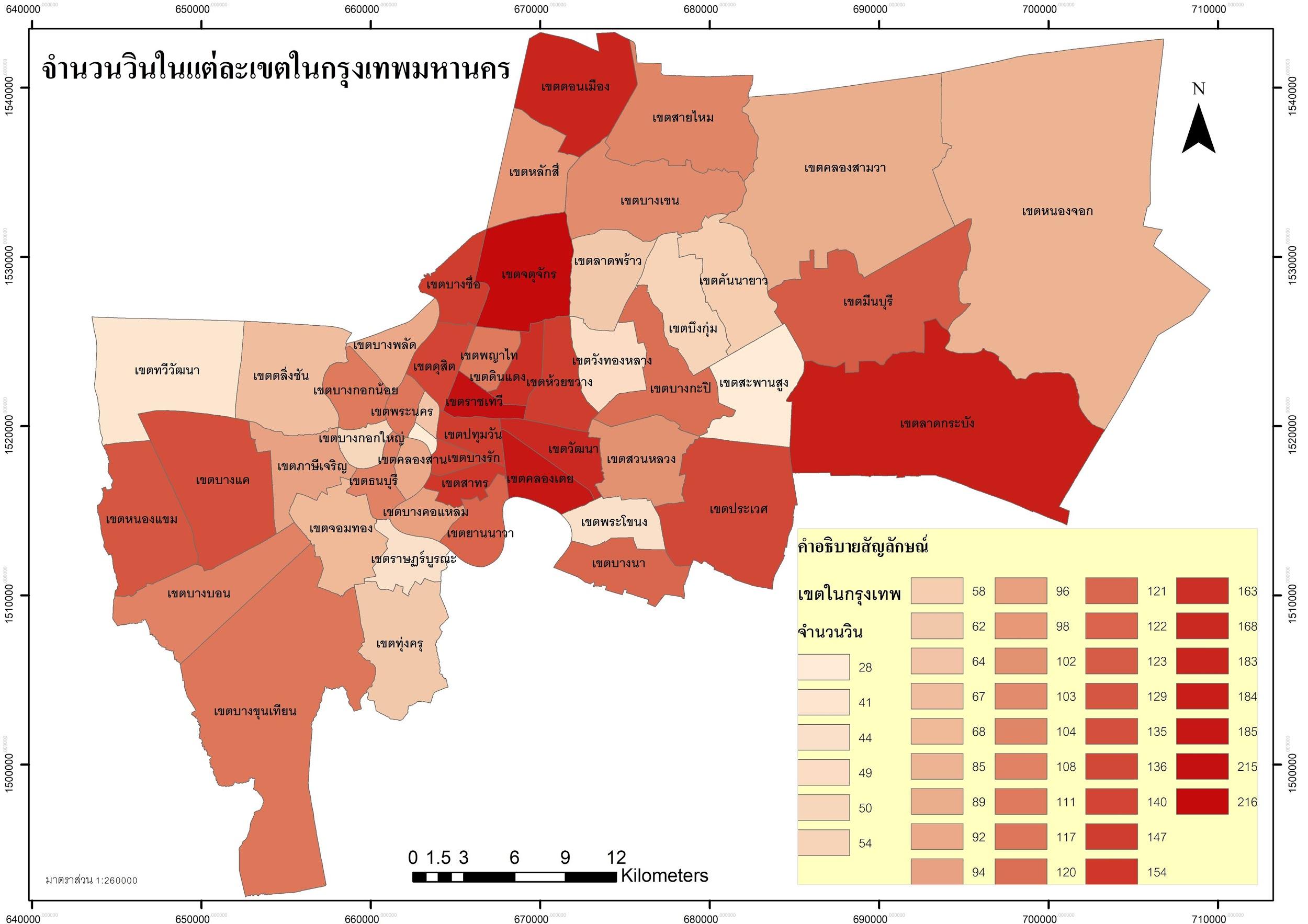
After analysing and plotted against the deliverymen cost. We can see the gap between the income of the business and the partial expense. The missing part was the service provider cost.



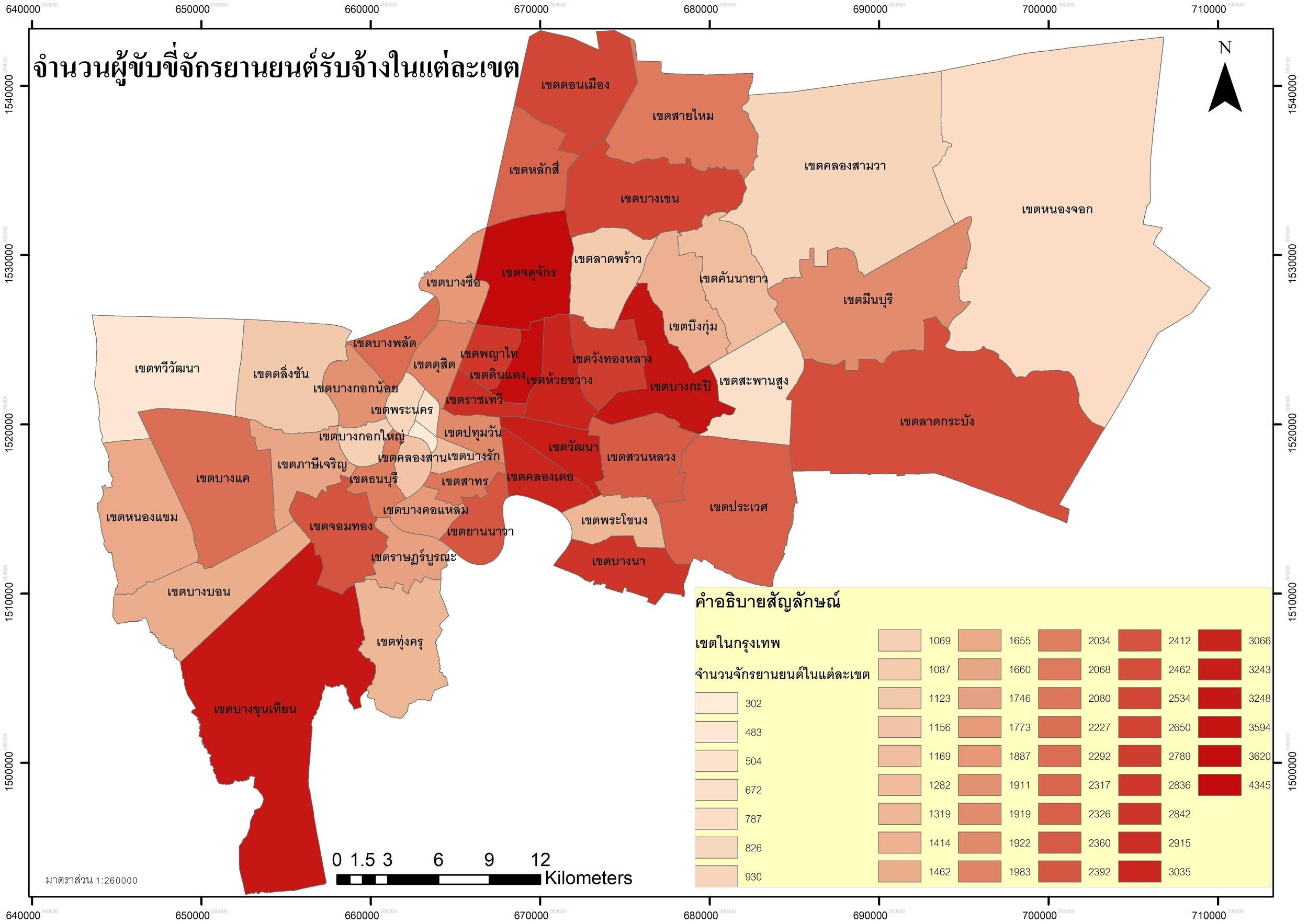
We can see that the SendRanger price is the highest in all of the pricing model at first which means more flexible gap between the income and the partial cost. The model of SKOOTA will meet the SendRanger at 64 kilometers which means after 64 kilometers, SKOOTA’s model will exceed SendRanger’s.

### 3.2.3 Moto-taxi in Bangkok

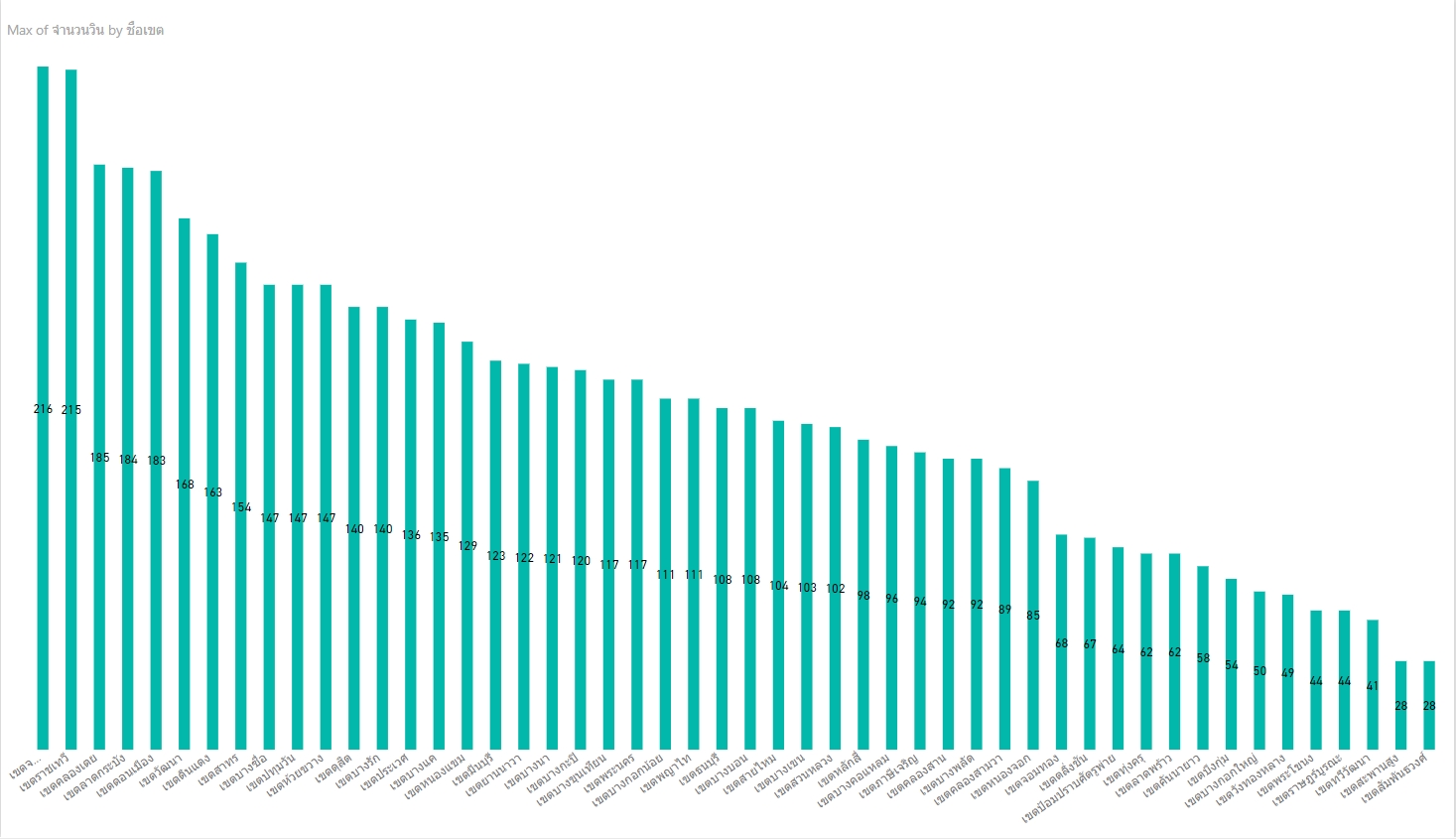
Over 100,000 moto-taxi were operated throughout Bangkok (Ministry of Social Development and Human Security, 2017). 83% of moto-taxi were used to deliver the documents on the behalf of the companies (ณัฐนัย โล้พิรุณ และ ทรรศ์ อนุรักษ์วงศ์ศรี, 2016). 86% are willing to do the delivery job if the income is reasonable (ณัฐนัย โล้พิรุณ และ ทรรศ์ อนุรักษ์วงศ์ศรี, 2016).



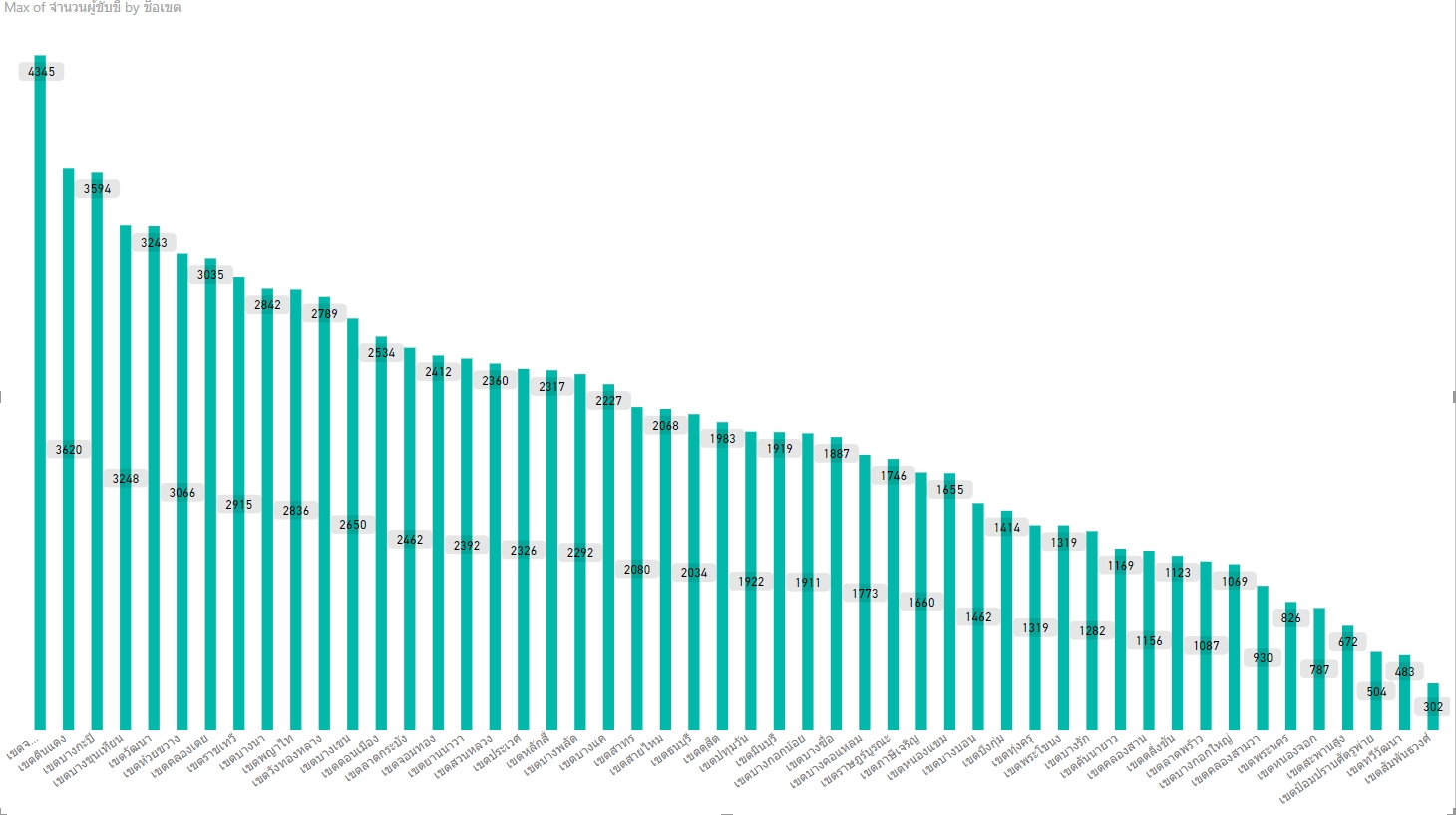
From the picture, we can see the saturation of the number of moto-taxi depot through all district in Bangkok.



From the picture, we can see the saturation of the number of moto-taxi through all district in Bangkok.



From the picture, we can see the bar chart of the number of moto-taxi depot through all district in Bangkok.



From the picture, we can see the bar chart of the number of moto-taxi through all district in Bangkok.

# บทที่ 4

# **The result**

The optimization technique of this model is based on the genetics algorithm and applied with different methods of arrangement.

## 4.1 Algorithm

### 4.1.1 Modified Li and Lim’s instances

First, the modified Li and Lim’s instances. We modified Li and Lim’s instances namely lc101-lc109,lc201-208,lr101-106,lrc101-105; total 28 instances, from 1 depot to 5 depots, then we assigned the requests to the depots as described in 3.1.2. After that, we solved the problems depot by depot using GA.

**Parameters for GA**:

Population size = 100

Generations = 2000; break if the result is non-improving for 500 generations

Crossover rate = 1.0

Mutation rate = 0.5

Because of the random nature of GA, the tests were run 10 times and the results were calculated for the means(averages).

Note: CT = Total Computational time

### 4.1.1.1 Simple Assigned

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Instance names | AVG - simple distances | | | | | |  |
|  | DEPOT-0 | DEPOT-1 | DEPOT-2 | DEPOT-3 | DEPOT-4 | Total | CT |
|  |  |  |  |  |  |  |  |
| lc101.txt | 149.2499 | 95.79996 | 148.5184 | 201.1459 | 52.5273 | 647.2414 | 11.47188 |
| lc102.txt | 147.6996 | 103.6621 | 148.5184 | 201.1459 | 52.5273 | 653.5533 | 12.91153 |
| lc103.txt | 151.1196 | 93.36045 | 147.74 | 200.3889 | 109.7609 | 702.3699 | 16.20411 |
| lc104.txt | 188.7658 | 106.7806 | 130.8621 | 194.7995 | 109.7609 | 730.9689 | 27.59068 |
| lc105.txt | 148.7564 | 98.78233 | 148.5184 | 197.9585 | 52.5273 | 646.5428 | 11.66214 |
| lc106.txt | 210.3113 | 97.91315 | 139.4501 | 201.1459 | 52.5273 | 701.3478 | 12.68033 |
| lc107.txt | 147.6996 | 103.7609 | 148.5184 | 201.1459 | 52.5273 | 653.6521 | 11.84812 |
| lc108.txt | 151.9818 | 96.58973 | 148.5184 | 201.2129 | 52.5273 | 650.8301 | 14.9553 |
| lc109.txt | 150.8102 | 91.38035 | 148.5184 | 201.1459 | 97.64936 | 689.5042 | 20.36613 |
| lc201.txt | 247.2168 | 152.7656 | 203.8168 | 259.4879 | 191.4071 | 1054.694 | 11.96596 |
| lc202.txt | 249.3299 | 174.6948 | 206.8661 | 267.0632 | 165.4433 | 1063.397 | 23.05458 |
| lc203.txt | 232.8206 | 186.341 | 200.6914 | 253.9903 | 165.4433 | 1039.286 | 35.02466 |
| lc204.txt | 236.2056 | 151.7586 | 186.0305 | 252.159 | 186.9513 | 1013.105 | 80.30202 |
| lc205.txt | 289.6272 | 151.6655 | 188.7633 | 245.7608 | 152.9611 | 1028.778 | 17.97586 |
| lc206.txt | 274.0689 | 161.7423 | 219.3085 | 290.7121 | 129.9753 | 1075.807 | 28.04533 |
| lc207.txt | 289.6272 | 151.4589 | 188.7633 | 245.6129 | 152.9611 | 1028.423 | 27.13088 |
| lc208.txt | 289.3896 | 151.4964 | 188.7633 | 243.1005 | 152.9611 | 1025.711 | 25.2196 |
| lr101.txt | 396.4272 | 423.5846 | 180.3528 | 508.8487 | 191.7622 | 1700.975 | 13.49554 |
| lr102.txt | 486.0616 | 346.6974 | 189.2025 | 418.0515 | 200.2603 | 1640.273 | 19.04473 |
| lr103.txt | 498.3646 | 214.0585 | 132.0188 | 452.6566 | 159.8818 | 1456.98 | 18.26399 |
| lr104.txt | 230.8062 | 207.7172 | 135.1735 | 384.5955 | 141.0829 | 1099.375 | 27.14254 |
| lr105.txt | 430.8509 | 234.4518 | 153.8744 | 400.0975 | 120.9389 | 1340.214 | 14.72767 |
| lr106.txt | 305.9395 | 300.4429 | 79.57826 | 411.5049 | 199.0404 | 1296.506 | 17.81936 |
| lrc101.txt | 324.7367 | 248.997 | 159.8159 | 453.8169 | 321.3387 | 1508.705 | 18.80513 |
| lrc102.txt | 347.9504 | 229.3172 | 100.9853 | 488.8267 | 259.036 | 1426.116 | 21.20696 |
| lrc103.txt | 263.5936 | 170.3745 | 130.7625 | 456.2756 | 202.5294 | 1223.536 | 26.22574 |
| lrc104.txt | 190.2777 | 190.6143 | 100.2531 | 371.9012 | 237.2097 | 1090.256 | 28.57814 |
| lrc105.txt | 338.3259 | 269.0521 | 127.6637 | 497.7443 | 276.0301 | 1508.816 | 21.04318 |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| **Mean** | **263.1434** | **178.7593** | **156.4945** | **310.7963** | **151.4125** | **1060.606** | **21.95579** |

### 4.1.1.2 Nearest Routable

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Instance names | AVG Nearest Routable Distances | | | | | |  |
|  | DEPOT-0 | DEPOT-1 | DEPOT-2 | DEPOT-3 | DEPOT-4 | Total | CT |
|  |  |  |  |  |  |  |  |
| lc101.txt | 149.2499 | 95.79996 | 148.5184 | 201.1459 | 52.5273 | 647.2414 | 10.83327 |
| lc102.txt | 110.2067 | 110.4073 | 148.5184 | 201.1459 | 52.5273 | 622.8055 | 11.79633 |
| lc103.txt | 151.1196 | 93.36045 | 147.74 | 200.3889 | 109.7609 | 702.3699 | 15.29504 |
| lc104.txt | 159.1492 | 108.9697 | 144.4736 | 194.9321 | 109.7609 | 717.2854 | 24.36398 |
| lc105.txt | 110.2067 | 110.7577 | 148.5184 | 197.9585 | 52.5273 | 619.9685 | 10.4885 |
| lc106.txt | 110.2067 | 110.7577 | 148.5184 | 201.1459 | 52.5273 | 623.156 | 11.12329 |
| lc107.txt | 156.2131 | 106.4665 | 148.5184 | 201.1459 | 52.5273 | 664.8712 | 11.14795 |
| lc108.txt | 151.9818 | 96.58973 | 148.5184 | 201.1459 | 52.5273 | 650.7631 | 13.39045 |
| lc109.txt | 166.2218 | 71.824 | 148.5184 | 201.1459 | 52.5273 | 640.2373 | 21.88619 |
| lc201.txt | 251.6853 | 152.7656 | 238.6353 | 255.5187 | 109.1542 | 1007.759 | 12.45302 |
| lc202.txt | 355.2567 | 169.5529 | 175.027 | 271.5837 | 170.6773 | 1142.098 | 17.78973 |
| lc203.txt | 248.3867 | 179.9877 | 169.8363 | 260.2301 | 168.4907 | 1026.931 | 33.00888 |
| lc204.txt | 227.5533 | 151.7586 | 170.9032 | 255.3379 | 157.0016 | 962.5546 | 68.69705 |
| lc205.txt | 313.1322 | 153.2748 | 241.6715 | 186.9052 | 115.5669 | 1010.551 | 18.78839 |
| lc206.txt | 133.5926 | 151.6928 | 255.5105 | 280.9676 | 71.05408 | 892.8176 | 32.70072 |
| lc207.txt | 238.5925 | 153.0683 | 241.7235 | 217.0051 | 115.5669 | 965.9563 | 29.10972 |
| lc208.txt | 228.9993 | 153.1058 | 245.9326 | 197.6421 | 107.9849 | 933.6647 | 29.49411 |
| lr101.txt | 521.486 | 424.1287 | 0 | 503.2854 | 138.8182 | 1587.718 | 22.08667 |
| lr102.txt | 375.3949 | 309.719 | 162.6991 | 412.1281 | 267.7143 | 1527.655 | 24.56394 |
| lr103.txt | 378.9213 | 324.4335 | 107.0365 | 405.6595 | 149.5323 | 1365.583 | 28.71268 |
| lr104.txt | 156.9358 | 256.3231 | 110.2158 | 395.5352 | 135.3268 | 1054.337 | 34.63675 |
| lr105.txt | 457.5445 | 288.7077 | 77.89621 | 402.9959 | 97.2112 | 1324.356 | 25.31384 |
| lr106.txt | 254.2653 | 447.5419 | 57.71723 | 377.029 | 192.739 | 1329.292 | 22.82423 |
| lrc101.txt | 336.7795 | 259.7795 | 159.8159 | 482.2595 | 262.7261 | 1501.36 | 45.06618 |
| lrc102.txt | 229.2696 | 281.8787 | 80.4541 | 630.9962 | 259.036 | 1481.635 | 59.27429 |
| lrc103.txt | 290.6056 | 210.552 | 133.5904 | 360.5795 | 197.1662 | 1192.494 | 62.08121 |
| lrc104.txt | 168.2038 | 287.791 | 70.10277 | 320.3038 | 172.4077 | 1018.809 | 75.3196 |
| lrc105.txt | 312.4433 | 277.2998 | 60.45047 | 485.8129 | 276.0301 | 1412.037 | 48.31497 |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| **Mean** | **240.843** | **197.7962** | **146.1093** | **303.6404** | **133.9792** | **1022.368** | **29.30575** |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

### 

### 4.1.1.3 Nearest Routable 3-vote

### 4.1.1.4 Nearest Routable 3-vote with condition

## 

### 4.1.2. Results on the GOO instances (real places)

### 4.1.2.1 Simple Assigned

### 4.1.2.3 Nearest Routable

### 4.1.2.3 Nearest Routable 3-vote

### 4.1.2.4 Nearest Routable 3-vote with condition

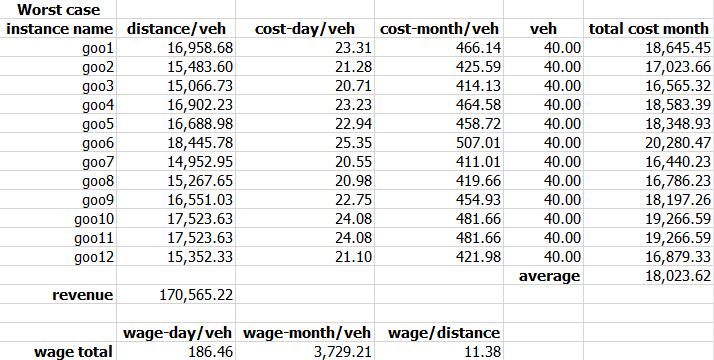
## 

## 

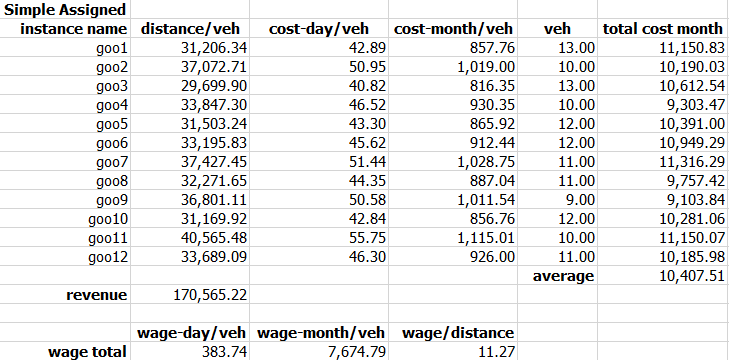
## 4.2 Business Model

### 4.2.1 Worst Case

In order to point out the comparable result, the worst case must be compiled. The worst case is where the individual deliverymen does only one request then back to the depot. This method causes an individual vehicle to run a short distance but a large amount in total.



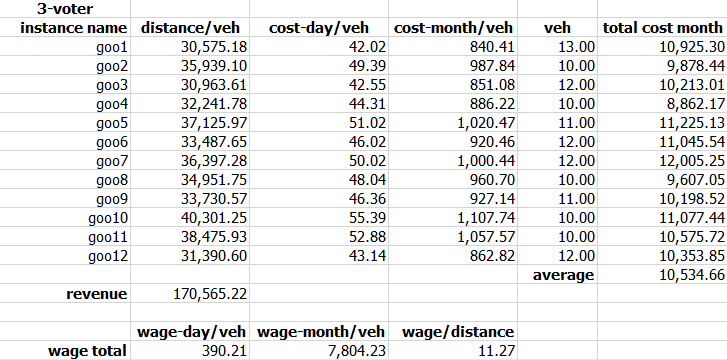
### 4.2.2 Simple assigned



### 4.2.3 Nearest Routable

table!

### 4.2.4 Nearest Routable 3-voter



### 4.2.5 Nearest Routable 3-voter with condition

Table!

# 

# 

# 

# 

# **บทที่ 5**

Conclusion

Table!

# **สรุปผลการวิจัย อภิปรายผล และข้อเสนอแนะ**

..................................................................................................................................................................................................................................................................................................................................................................................................................................

**For the future suggestions, there are many things that can be improved described below.**

**First, in this project the requests are assigned deterministically in the beginning. We can improve the quality of the solutions of every instances further by create chromosomes that also represent the depots. Next, the computational times are expensive because implemented this project in Python, the algorithms may run faster if we implement it in faster languages (C++). Moreover, the more realistic constraints can be considered such as dynamic requests, sacrificing some requests, multi-criteria etc.**

# **เอกสารอ้างอิง**

(นิสิตต้องอ้างอิงให้ถูกต้องตามหลักการ สามารถศึกษาวิธีการเขียนอ้างอิงได้จากเอกสาร “คู่มือการจัดทำปริญญานิพนธ์”)