รายงานความก้าวหน้าครั้งที่ 2 โครงงานทางวิศวกรรม (ข้ามสาขาวิชา)

(Progress Report 2: Senior Project XD)

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

รหัสโครงงาน SP 3

ชื่อโครงงาน

(ภาษาไทย)	อัลกอริทึมสำหรับการแก้ปัญหาการส่งเอกสาร
(ภาษาอังกฤษ)	Algorithm for solving messenger service problem

รายชื่อสมาชิกกลุ่ม :

ชื่อ – สกุล	**รหัสประจำตัว**	**ลายมือชื่อ**
1. นาย ปรัชญา เอี่ยมทรงศักดิ์	5730327221	ปรัชญา
2. นาย ภคภูมิ ปธานราษฎร์	5730430121	ภคภูมิ
3. นาย วสุวัชร สถิตธรรมจิตร	5731095821	วสุวัชร

รายชื่ออาจารย์ที่ปรึกษาโครงงาน :

1. ผศ.ดร.มาโนช โลหเตปานนท์
2. ผศ.ดร.อรรถสิทธิ์ สุรฤกษ์

วัตถุประสงค์ของโครงงาน

The objective of this project is to find and analyze an algorithm for the messenger service problem. The messenger service is a service that users have requests to send documents. The service providers assign the delivery men to 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, which is a variation of the vehicle routing problems. Given, the requests and constraints from users (the pickup points, the delivery points, distances and traveling times, time windows, load capacities, service times), we want to assign the tours (routes) of vehicles for the delivery men with the lowest costs (travel distances) that still satisfy all of the constraints.

ขอบเขตการดำเนินงาน

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 traveling distances between places, time windows that users appoint to receive the documents, coordinates of places. We want to generate the jobs with the lowest total travel distances (and maybe also total waiting time of all vehicles). 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.

แผนการดำเนินงานโครงงาน

First, we have studied research papers about the subject in the first semester. Currently, we are trying to get clearer objectives and constraints. For example, apart from traveling distances, should we also consider other factors as costs? Also, we are currently trying to extend our algorithm for multi-depot and dynamic versions of the problem. In addition, we are formulating our problems into mathematical models. We are also testing both on the benchmark test instances and creating real test instances using real places to test the algorithm.

Second, apart from formulation. We are developing our own business model that suits the project. To emphasize the idea, none of the business available today are provide the optimized route for driver, they only provide a job, single back and forth service, from a request. So, in order to develop the business model we need to know the raw cost (gas, driver salary and maintaining cost) and the price that satisfied. Furthermore, not only the raw cost as mentioned, we as a service provider needs to provide a price that satisfies all aspects e.g. driver, customer and service provider itself.

รายละเอียดผลการดำเนินงานจนถึงปัจจุบัน

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.

Populations = An array of chromosomes

Chromosome = [Gene$_1$, Gene$_2$, Gene$_3$, ... ,Gene$_v$] ; v= the number of vehicle used (the number of routes)

Gene = [num, {Requests} , [Route]]

{Requests} = A set of indices of requests.

request = a pair of an index pickup node and the corresponding index delivery node

[Route] = An array of a route (permutation of the indices of nodes)

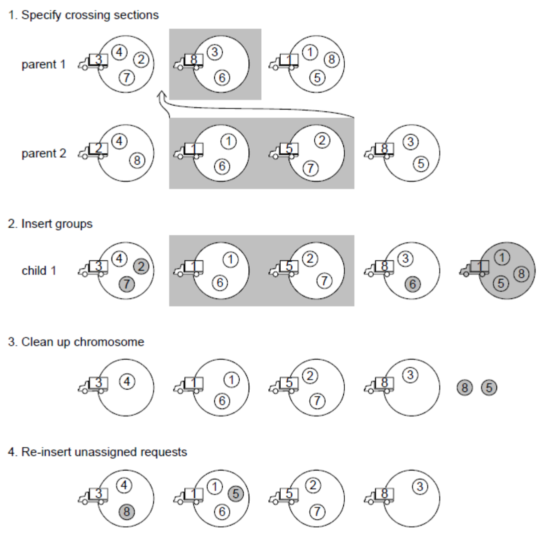
Procedures:

____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)

Instance name	Approx. Nodes	Best known result (distances)	Best known solutions' number of routes	Our result (distance)	My solutions' number of routes	Our computational time (seconds)
lc101	100	828.94	10	**828.94***	**10**	23
lc102	100	828.94	10	**828.94***	**10**	59.1
lr211	100	911.52	2	**899.05****	3	851.04
lrc106	100	1424.73	11	1460.23	12	89.11
lr111	100	1108.90	10	**1108.90***	10	133.79
lc_2_2_1	200	1931.44	6	**1931.44***	**6**	178.00
lr1_2_9	200	5050.75	13	**4024.60****	17	368.52
lc1_4_1	400	7152.06	40	**7152.06***	**40**	1051

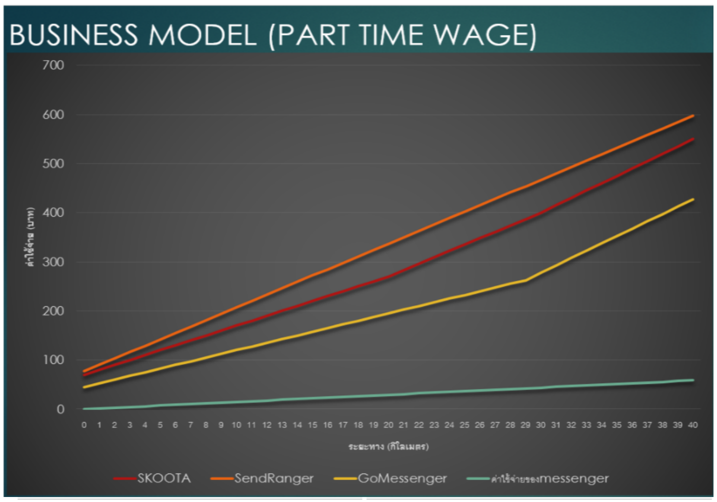
(*) Results that equal to the the best known solutions

(**) 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.

Business Model

The messenger service business consists of 3 major compositions the user, the service provider and the driver. To clarify, the users are the one whom provide request(s). The service provider generate the optimized route then, drivers accommodate the job(s). To make a business model we need to know the cost of all aspects such as the motorcycle fuel consumption, the driver minimum wage and the competitor service price. After researching, we are able to plot the cost and supply graph seeking for the optimum point. This model is only a preliminary model, further developments are carried out.



Reference:

[1] 2005, Giselher Pankratz, A Grouping Genetic Algorithm for the Pickup and Delivery Problem with Time Windows.