# Mode Fair Home Assessment
# Vehicle Fleet Optimisation
Version 1

# Contents

# 1 Background

The company has a fleet of delivery vehicles which are responsible for meeting the demand of the business' customers. To ensure that the business is allocating resources efficiently, business has requested for the development of an algorithm or program that will find the best route for the fleet of vehicles.

# 2 Overview

## 2.1 Objective

To develop an algorithm to compute the best route for the fleet of vehicles.

## 2.2 Benefit

To ensure efficient routing of vehicles and reduce the cost of delivery.

## 2.3 Scope

### 2.3.1 Depot

There is only one Depot, located at (Latitude = 4.4184, Longitude = 114.0932)

### 2.3.2 Types of Vehicles

There are 2 types of Vehicles, each with different capacity and cost.

| Vehicle | Capacity | Cost |
|---|---|---|
| Type A | 25 | RM 1.2 per km |
| Type B | 30 | RM 1.5 per km |

### 2.3.3 Customers

There are 10 customers with different demands, situated in different locations.

| Customer | Latitude | Longitude | Demand |
|---|---|---|---|
| 1 | 4.3555 | 113.9777 | 5 |
| 2 | 4.3976 | 114.0049 | 8 |
| 3 | 4.3163 | 114.0764 | 3 |
| 4 | 4.3184 | 113.9932 | 6 |
| 5 | 4.4024 | 113.9896 | 5 |
| 6 | 4.4142 | 114.0127 | 8 |
| 7 | 4.4804 | 114.0734 | 3 |
| 8 | 4.3818 | 114.2034 | 6 |
| 9 | 4.4935 | 114.1828 | 5 |

| | | | |
|---|---|---|---|
| 10 | 4.4932 | 114.1322 | 8 |

### 2.3.4 Requirements

Hard Constraint:

- Each delivery location must be visited exactly once.
- The total demand of each vehicle route must not exceed its maximum capacity.

Soft Constraint:

- Minimize cost required to meet all demands.

### 2.3.5 Key Assumption

- The vehicles start and end their routes at the same depot location.
- Each vehicle only travels one round trip. (depart from depot and back to the depot)
- There is no limit on the number of vehicles.
- Travel times between any two locations are the same in both directions.
- Deliveries can be made at any time, there are no time windows for deliveries.
- Vehicle travel distance is calculated using Euclidean distance formula.

## 2.4 Approach

### 2.4.1 Distance Formula

Euclidean distance formula:

$$distance\ (km) = 100 * \sqrt{(\text{Longitude2} - \text{Longitude1})^2\ +\ (\text{Latitude2} - \text{Latitude1})^2}$$

### 2.4.2 Tabu Search Algorithm

Tabu Search Algorithm is used to determine the most efficient route as well as the combination fleet of transport vehicles. As this approach leverages on an initial solution to acquire the optimised solution, multiple initial solutions shall be used.

#### *2.4.2.1 Generating Neighbours*

To generate neighbours, two swapping methods shall be implemented.

a) Swapping among own routes

Before:

Vehicle 1


Vehicle 2


After:

Vehicle 1


Vehicle 2

b) Swapping with other routes

Before:

Vehicle 1


Vehicle 2


After:

Vehicle 1


Vehicle 2

# 3 Results and Findings

The algorithm was run for number of vehicles between 2 to 6 and 20 sets of solutions were generated for each group. The full results can be found in the Appendix.

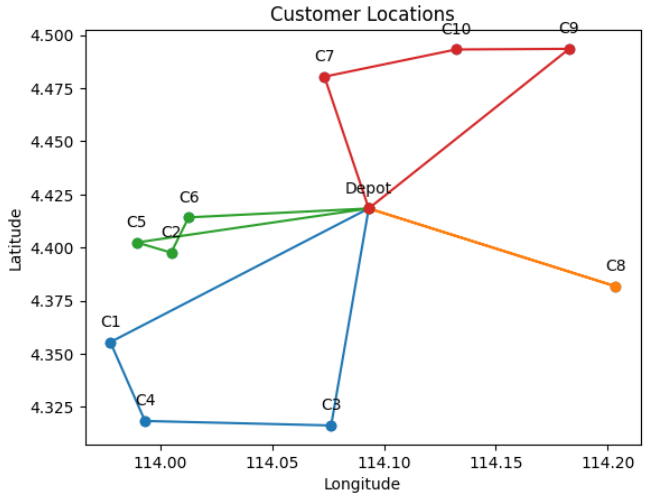
| Max Vehicles | Mean | | | |
|---|---|---|---|---|
| | Total Distance (before) | Total Cost (before) | Total Distance (after) | Total Cost (after) |
| **2** | 131.24781 | 196.871716 | 79.41051 | 104.92603 |
| **3** | 130.34389 | 163.500522 | 76.203356 | 94.026466 |
| **4** | 123.08585 | 151.486536 | 75.025339 | 90.580835 |
| **5** | 124.10248 | 150.545009 | 74.690698 | 90.06583 |
| **6** | 122.18255 | 152.374434 | 74.854034 | 90.675882 |

| Max Vehicles | Minimum | | | |
|---|---|---|---|---|
| | Total Distance (before) | Total Cost (before) | Total Distance (after) | Total Cost (after) |
| **2** | 103.2601 | 154.8902 | 69.52929 | 91.53985 |
| **3** | 108.467 | 142.6447 | 65.43126 | 78.51751 |
| **4** | 86.67678 | 104.0121 | 64.20533 | 80.79574 |
| **5** | 106.9144 | 128.2972 | 63.38749 | 76.06498 |
| **6** | 98.26213 | 117.9146 | 66.28039 | 79.53647 |

The summary above illustrates how the optimum solution lies within the band with the limit of 5 Max Vehicles, with their mean total cost averaging around RM91.62, and with the solution giving a Total Cost of RM 76.06.

It is interesting to note that although there is the availability of both Type A and Type B vehicles, only 21 out of the 80 optimised solutions (excluding 2 Max Vehicles generated use Vehicle Type A. As such, it may also be worth taking into consideration to decommission that vehicle type.

## Best Optimum Solution




*Total Distance = 63.3875 km*

*Total Cost = RM 76.06*

*Vehicle 1 (Type A):*

*Round Trip Distance:22.691 km, Cost: RM 27.23, Demand:14*

*Depot -> C3 (10.347 km) -> C4 (8.323 km) -> C1 (4.021 km)*

*Vehicle 2 (Type A):*

*Round Trip Distance:11.612 km, Cost: RM 13.93, Demand:6*

*Depot -> C8 (11.612 km)*

*Vehicle 3 (Type A):*

*Round Trip Distance:11.499 km, Cost: RM 13.8, Demand:21*

*Depot -> C6 (8.061 km) -> C2 (1.834 km) -> C5 (1.604 km)*

*Vehicle 4 (Type A):*

*Round Trip Distance:17.586 km, Cost: RM 21.1, Demand:16*

*Depot -> C7 (6.508 km) -> C10 (6.018 km) -> C9 (5.06 km)*

# 4 Appendix

## 4.1 Inputs

| File Name | Description | File |
|---|---|---|
| customers.csv | Sample Data for customers | customers.csv |
| depot.csv | Sample Data for Depot | depot.csv |
| vehicles.csv | Sample Data for Vehicle | vehicles.csv |

## 4.2 Figures for Solutions

| File Name | Description | File |
|---|---|---|
| fig.zip | All figures for the generated initial solution and their optimised solutions.<br><br>*Naming Convention:*<br>*<max_vehicle>_<iteration>_<ind>*<br>*where ind = i for initial solution*<br>*ind = f for final optimised solution* | fig.zip |

## 4.3 Jupyter Notebook

| File Name | Description | File |
|---|---|---|
| solution.ipynb | Jupyter Notebook for the code for the entire project | solution.ipynb |

# 5 Version History

| Ver | Date | Change Description | Author(s) |
|---|---|---|---|
| **1.0** | 18/05/2024 | First Draft | Tyler Ho |
| | | | |
| | | | |
| | | | |
| | | | |