Last-mile delivery model for E-commerce businesses using self-storage

COMP4801 - Final Year Project
Interim Report

<table>
<thead>
<tr>
<th>Name</th>
<th>UID</th>
</tr>
</thead>
<tbody>
<tr>
<td>TANG Wing Kan</td>
<td>3035468304</td>
</tr>
</tbody>
</table>

Supervised by Dr C. Huang

Mon Jan. 21 2023
ACKNOWLEDGEMENTS

I would like to thank my supervisor, Dr Chao Huang, for his encouragement, motivation and the considerable time he spent throughout the final year project.

I would also thank my progress report advisor, Dr Matthew Yeung, for his valuable suggestions and remarks.

I am grateful to all my schoolmates who have encouraged me during my study.
ABSTRACT

Last-mile delivery is an essential part of eCommerce logistics. Most eCommerce will use third-party logistics (3PL) to handle the final delivery. Nevertheless, the solution is not adaptive to a real-time environment, leading to frequently failed delivery. Therefore, a new delivery approach, the self-storage-as-parcel model (SSAP), has been proposed to facilitate last-mile delivery. The objective is to investigate the best-fit algorithm to optimize routes for SSAP. In this study, the SSAP will be treated as a Vehicle Routing Problem with Time Windows (VRPTW), an extension of the Vehicle Routing Problem (VRP). The objection function in this model is the minimization of total travel cost and the number of vehicle routes. Constraints on time window, vehicle capacity and vehicle route are assumed to be tight. A comparative study will be carried out on the existing VRPTW algorithms. Each algorithm will be tested on VRPTW data sets. The comparison results will show the algorithms' effectiveness and superiority over other algorithms. Currently, the team has identified the mathematical formation and complexity of VRPTW. The team also finalized three heuristic algorithms for the comparative experiment. The current progress is considered acceptable, and the team will focus on the experiment demo and prepare for the first presentation in this phase. Given the limitation that the routing model omits other delivery factors, such as multi-depot, it is suggested that more constraints can be added to the model in the future.
# TABLE OF CONTENTS

1. **BACKGROUND**
   
   1.1. **BACKGROUND OF "LAST MILE" DELIVERY**
   1.2. **THE CURRENT APPROACH TO "LAST MILE" DELIVERY**
   1.3. **CHALLENGES OF EXISTING "LAST MILE" SOLUTION**
   1.4. **AN ALTERNATIVE APPROACH FOR THE CURRENT LAST-MILE MODEL**
   1.5. **PROJECT SCOPE**
   1.6. **OUTLINE OF REPORT**

2. **METHODOLOGY**
   
   2.1. **LITERATURE REVIEW**
   2.2. **LANGUAGE AND ENVIRONMENT**
   2.3. **DATASET**

3. **CURRENT PROGRESS**
   
   3.1. **MATHEMATICAL FORMATION**
   3.2. **VRPTW COMPLEXITY**
   3.3. **ALGORITHMS FOR THE COMPARATIVE EXPERIMENT**
   3.4. **IMPLEMENTATION OF EXPERIMENTAL DEMO**
   3.4.1. **DATA PRE-PROCESSING**
   3.4.2. **IMPLEMENTATION OF GENETIC ALGORITHM**

4. **SCHEDULE**

5. **CONCLUSION**

6. **REFERENCE**

7. **APPENDICES**
   
   **APPENDIX A: LIST OF ABBREVIATIONS**
   **APPENDIX B: LIST OF SYMBOLS**
LIST OF FIGURE

FIGURE 1. JSON FILE FORMAT FOR DEFINING EACH PROBLEM INSTANCE (ASSUMING 100 SELF-STORAGES) 13

FIGURE 2. DESCRIPTION OF FITNESS VALUE IN EACH GENERATION FOR GENETIC SEARCH ON R101 14

FIGURE 3. TERMINATION OF GENETIC SEARCH ON R101 AND OPTIMIZED ROUTE AND TOTAL COST OUTPUT 14
LIST OF TABLE

TABLE 1. PROJECT SCHEDULE (MAY BE SUBJECT TO CHANGE)
1. BACKGROUND

1.1. Background of "last mile" delivery
Logistic management is the backbone of every eCommerce business. The logistic system's effectiveness directly affects the company's profit margin and the customer's purchasing experience. "Last mile" delivery is the most critical part of eCommerce logistics. It transfers goods from the last point of transit to the final destination. An effective last-mile delivery strategy ensures that the correct goods are delivered to customers' hands within the prearranged deadline [1]. Over half of the shipping costs are associated with last-mile delivery, equivalent to more than 40% of the total supply chain expense [2], making it the most crucial stage of the entire delivery process.

1.2. The current approach to "last mile" delivery
Different companies have their last-mile logistic strategy. Many eCommerce businesses will use third-party logistics (3PL) to tackle the final delivery process [3]. The 3PL outsources the transportation processes of eCommerce to other transportation service providers [4]. eCommerce can then match a suitable delivery service provider to optimize its delivery route so that e-commerce provides customers with quick packet delivery while lowering their delivery costs [5].

1.3. Challenges of existing "last mile" solution
The 3PL approach did improve the last-mile delivery of eCommerce to some degree. However, there is a difference between a theoretical solution and a real-life situation. Most 3PL routes cannot adapt to the highly dynamic real-time environment. Fail, or late delivery prevails even though the logistic providers have a tracking system to manage the delivery progress throughout the entire process. Packages are not given to the customer within a fixed time window.

Delivery delays result in 2 negative impacts on the eCommerce company. Firstly, customers cannot have an enjoyable purchasing experience. Research has illustrated that 57% of consumers would not shop with the same retailer again after three late orders [6]. Fail delivery impedes business from forming a solid customer base. Secondly, eCommerce businesses may need to pay more for last-mile delivery. As some of the delivery is
unsuccessful under these approaches, eCommerce companies are forced to pay extra costs for rescheduling the order, reducing eCommerce's profit margin [7].

1.4. An alternative approach for the current last-mile model

Another approach is introduced to the last-mile delivery model to increase delivery efficiency. Instead of delivering directly from the depot to the customer's address, this project introduces the self-storage-as-parcel model (SSAP) to address the last-mile delivery. This model reduces some complex factors for the current depot-to-customer model to handle. The constraint of customer availability is diminished as the customer can now collect their packet from the pickup spot [8], [9]. Thus, time and memory for computing an optimized route are lessened.

1.5. Project scope

This project aims to investigate the best-suit algorithm for this alternative approach to solve the last-mile problem. The project treats the last mile problem as a Vehicle Routing Problem with Time Windows (VRPTW), an extension of the Vehicle Routing Problem (VRP). In the VRPTW model, the delivery must initiate and end within specific times of the day [10]. Restrictions such as Capacities on vehicles and Service time spent at each drop-off point will also be considered in the VRPTW model so that the model can be more adaptive to the real-life setting. The project's objective is to find an algorithm that can provide an optimized route with minimum travel time for last-mile delivery. The quality of the selected algorithms will be evaluated from the comparative study.

1.6. Outline of report

This report presents the research progress of the last-mile delivery model for E-commerce businesses using self-storage. Section 1 focuses on the background of last-mile delivery. The section also proposes a new model for the last mile problem and indicates the project's objectives. Section 2 details the project's methodology, including literature review, learning and environment, and dataset. In section 3, the project team explains the findings of the current progress and the algorithms finalized for the comparative study. Section 4 provides the schedule for future studies and deliverables. A conclusion of the study is provided in the section with problems encountered and the study's recommendation.
2. METHODOLOGY

This section focuses on the project's methodology, which comprises three parts: literature review, language and environment, and dataset.

2.1. Literature review

VRPTW methods used for the experiment and comparison will be primarily sourced from relevant research papers and articles. Google Scholar is the leading search engine this project utilized for exploring related research papers. If some of the publications are unavailable through google scholar, this project will access publications through HKUL E-resources.

2.2. Language and Environment

This project uses python to write the code and test because python provides extensive libraries for data processing and modelling, such as NumPy and Pandas [11]. For the development and testing stage, this project uses Google Colaboratory to host Jupyter Notebooks for writing the concept and comparative experiment of selected algorithms, as it provides free GPUs from Google Research for machine learning projects [12]. All experiment and coding instructions and assumptions will also be written in the notebook to guide users before running the code.

2.3. Dataset

The data sets of Solomon (1987) (http://web.cba.neu.edu/~msolomon/problems.htm) are used for conducting a comparative experiment on the VRPTW algorithm. This is because the data sets of Solomon are one of the most well-known benchmark points for VRP [13].

The six sets of data comprise 56 problems. Data set C presents a relatively clustered base, while data set R presents a randomly distributed base. 2 RC data sets present a mix of the relatively concentrated and randomly dispensed base. Problem sets R1, C1, and RC1 have a narrow time window, scheduling horizon, and low vehicle capacity. On the other hand, problem sets R2, C2 and RC2 have an open time window, a large scheduling horizon and a high vehicle capacity. Only data set R is used for the testing in this project as self-storage facilities are in a randomly distributed base [14].
3. CURRENT PROGRESS

This section will describe the current status and progress made thus far. The project team will first explain the finding on the mathematical formation and complexity of VRPTW. Then the project team will present the algorithms decided for the comparative experiment.

3.1. Mathematical formation

The project team has deduced the mathematical formation for the VRPTW model, which is explained as follows with description [15]–[18].

Parameter:

\[
G = (N, A) \\
A = \{(i, j) : i, j \in N, i \neq j\} \\
N = \{0, 1, ..., n\} \\
N' = \{1, ..., n\} \\
V = \{1, ..., v\}
\]

\(v\): number of vehicle route
\(c_{ij}\): travel cost (time) of going from vertex \(i\) to vertex \(j\)

\(s_i\): Service time at vertex \(i\)

\(e_0\): Earliest start time of depot

\(l_0\): Latest return time of depot

\(e_i\): Earliest arrival time at vertex \(i\)

\(l_i\): Latest arrival time at vertex \(i\)

\(Q\): Vehicle capacity

Decision Variables:

\[
x_{ij}^k = \begin{cases} 
1 & \text{if vehicle } k \text{ travels directly from } i \text{ to } j \\
0 & \text{otherwise}
\end{cases}
\]

\[
y_j^k = \begin{cases} 
1 & \text{if storage } i \text{ is served by vehicle } k \\
0 & \text{otherwise}
\end{cases}
\]

\(w_i\): waiting time at vertex \(i\)

\(t_i\): arrival time at vertex \(i\)

\(z_i\): vehicle loading going to vertex \(i\)
The VRPTW model is given by vehicles, a set of customers and a directed graph $G$. The graph comprises $(n + 1)$ vertices denoted as $N$. Vertex 0 refers to the depot while vertices 1, 2, ..., $n$ represent all the self-storage facilities. $N'$ is the set of vertices excluding the depot. $V$ is the set of vehicle routes 1, 2, ..., $v$. The set of arcs $(A)$ represents the path between vertices. All arc starts from vertex 0 (depot) and terminates in the depot. A travel cost ($c_{ij}$) is assigned to each arc (from vertex $i$ to vertex $j$), where $i \neq j$, Each Vehicle has a capacity $Q$. Every vertex $i$ has a time window $[e_i, l_i]$. A vehicle must wait until the earliest arrival time at vertex $i$ ($e_i$) to drop off the load if the vehicle arrives at vertex $i$ earlier than $e_i$. Conversely, a vehicle cannot drop off at vertex $i$ if it arrives after its latest arrival time ($l_i$).

**Objective Function:**

\[
\text{Min } \sum_{i \in N} \sum_{j \in N} \sum_{k \in V} c_{ij} x_{ij}^k \tag{1}
\]

**subject to:**

\[
\sum_{i \in N} x_{ij}^k = y_j^k \quad \forall k \in V, \ \forall j \in N', \tag{2}
\]

\[
\sum_{j \in N} x_{ij}^k = y_i^k \quad \forall k \in V, \ \forall i \in N', \tag{3}
\]

\[
z_j \leq Q \quad \forall i \in N, \tag{4}
\]

\[
\sum_{k \in V} y_j^k = 1 \quad \forall i \in N', \tag{5}
\]

\[
\sum_{k \in V} y_i^k = v
\]

\[
l_i \geq t_i \geq e_i \quad \forall i \in N, \tag{7}
\]

\[
t_i + s_i + w_i + t_{ij} = t_j \quad \forall i \in N, \ \forall j \in N, \ i \neq j \tag{8}
\]

\[
w_i = \max \{e_i - t_i, 0\} \quad \forall i \in N, \tag{9}
\]

The objective function (1) minimizes the distance travelled with the same number of routes. Constraints (2,3) ensure that each route only allows one vehicle to pass. Constraint (4) is the capacity constraint for each vehicle. Constraint (5) ensures that each customer is assigned precisely one vehicle route. Constraint (6) represents that all the routes start from the depot. Constraints (7,8,9) explain the time window constraint.
3.2. **VRPTW Complexity**

The VRP is considered one of the most complex problems to tackle. The VRP shares many similarities with the Travelling Salesman Problem (TSP). TSP is an NP-hard problem investigating the shortest possible route for one person or car to reach every customer [19]. However, VRP is more complicated than TSP, even with a small number of destinations and a moderate number of travel requests [20]. Thus, VRPTW becomes a more challenging version of VSP as a time window is also included. Even if the number of vehicles is constant, an optimized solution for VRPTW is already an NP-Complete problem [21].

3.3. **Algorithms for the comparative experiment**

The project team decides to experiment with four classic heuristic algorithms: savings algorithm, insertion algorithm and sweep algorithm for testing. Exact algorithms cannot compute optimal solutions on NP-complete type problems within a limited time window [22]. Heuristic algorithms are more suitable as they only provide "nearly optimal solutions" [23].

The genetic algorithm is a randomized search technique for a group of individuals. Each individual presents a fitness value used for the genetic search, comprising Representation (encoding features of every individual to a chromosome), reproduction (choose two parent chromosomes from the current group), recombination (create two offspring from the parent chromosomes) and mutation (conduct mutation randomly on each offspring). Reproduction, recombination and mutation are repeated, and the search terminates when the population does not produce offspring significantly different from the previous generation [24].

Savings algorithm merges two separate loops of the routing issue into one circuit with the shortest total distance. The optimization can divide into serial and parallel manner [25].

Insertion algorithm integrates approach method into savings method: The farthest spot of the terminal is initially set as the seed point. Under the concept of adjacent points, the minimum insert value spot is set as the next spot. After that, the insert location is determined following the mechanism of the saving method. The selection and insertion will be repeated until the limit of the time window or vehicle capacity is met [26].
Polar coordinates are applied in the sweep algorithm to represent demand points. A random demand point is set as the start point with an angle equal to zero. The service zone is then split following the limited car capacity. Finally, the demand points are ranked for scheduling delivery routes [27].

3.4. Implementation of Experimental Demo

The project team has built an experimental demo for the Genetic algorithm, which is explained in the description.

3.4.1. Data Pre-processing

The required data set problem instance has been converted to JSON format for the experiment as follows (see Figure 1).

```json
[ ] {
    "instance_name" : "<Instance name>",
    "max_vehicle_number" : K,
    "vehicle_capacity" : 0,
    "depart" : {
        "coordinates" : {
            "x" : x0,
            "y" : y0
        },
        "demand" : q0,
        "ready_time" : e0,
        "due_time" : d0,
        "service_time" : s0
    },
    "customer_1" : {
        "coordinates" : {
            "x" : x1,
            "y" : y1
        },
        "demand" : q1,
        "ready_time" : e1,
        "due_time" : d1,
        "service_time" : s1
    },
    ...
    "customer_100" : {
        "coordinates" : {
            "x" : x100,
            "y" : y100
        },
        "demand" : q100,
        "ready_time" : e100,
        "due_time" : d100,
        "service_time" : s100
    },
    "distance_matrix" : [
        [dist_0, dist_1, ..., dist_100],
        [dist_1, dist_2, ..., dist_100],
        ...
        [dist_100, dist_100, ..., dist_0]
    ]
}
```

Figure 1. JSON file format for defining each problem instance (assuming 100 self-storages)
Each value in the "distance_matrix" represents the distance between 2 respective self-storages, which value should be 0 if the self-storages are the same.

### 3.4.2. Implementation of Genetic Algorithm

The following content illustrates the genetic search on the R101 data from the dataset of Solomon. In this project, the number of generations for this experiment is set to 100, and 5 vehicles are provided for the routing experiment. The total cost computed consists of the sub-routing time and transportation costs for every vehicle. The minimum value, maximum value, average value and standard deviation of the fitness value are calculated for each generation (Figure 2). After 100 generations, the program computed the optimized routes for each vehicle as well as the total cost of the route (Figure 3).

![Figure 2. Description of fitness value in each generation for genetic search on R101](image1)

![Figure 3. Termination of genetic search on R101 and optimized route and total cost output](image2)

Further modifications might be made, such as distance cost calculation and finalizing the number of vehicles and the number of generations, to test for every algorithm.
4. SCHEDULE

Table 1 provides details of the project schedule. All milestones for the inception and elaboration phases are complete, indicating that the project is on schedule till Jan. 22 2023. As the project enters the construction phase, the project team will start implementing code on other selected algorithms. The built genetic algorithm will also be reviewed and examined for further improvement. The performance of all the selected algorithms will then be evaluated and analyzed. Research studies regarding the topic will also continue to explore further research if possible. The project team will then organize all the findings and prepare for the final presentation before April 2, 2023.

In the rest of the construction phase, from early April to early May, the final report will be conducted, and test implementation will be finalized. All the results and implementation will be presented in the final presentation. The project team will also review the feedback from the supervisor and advisors and make further improvements before the project exhibition in early May.
<table>
<thead>
<tr>
<th>Date</th>
<th>Milestones</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oct. 3 2022</td>
<td>Conduct a further study on the relevant topic</td>
<td></td>
</tr>
<tr>
<td>– Jan. 13 2023</td>
<td>Build experimental demo</td>
<td></td>
</tr>
<tr>
<td>Jan. 14 2023</td>
<td>Prepare for the first presentation</td>
<td>Due: 9-13 Jan 2023</td>
</tr>
<tr>
<td>Jan. 14 2023</td>
<td>Write Interim Report and finish Preliminary implementation (Deliverable 2)</td>
<td>Due: Jan. 22 2023</td>
</tr>
<tr>
<td>Jan. 23 2023</td>
<td>Test and compare the algorithm proposed from the relevant publication</td>
<td></td>
</tr>
<tr>
<td>– Apr. 2 2023</td>
<td>Analyze testing results</td>
<td></td>
</tr>
<tr>
<td>– Apr. 18 2023</td>
<td>Explore further research (if possible)</td>
<td></td>
</tr>
<tr>
<td>Apr. 3 2023</td>
<td>Prepare for the Final Presentation</td>
<td>Due: 17-21 Apr 2023</td>
</tr>
<tr>
<td>– Apr. 19 2023</td>
<td>Write Final Report and finalize test implementation (Deliverable 3)</td>
<td>Due: Apr. 18 2023</td>
</tr>
<tr>
<td>Apr. 19 2023</td>
<td>Prepare for Project Exhibition</td>
<td>Due: May 3 2023</td>
</tr>
</tbody>
</table>

| May 2 2023 |                                            |                                            |

Table 1. Project schedule (may be subject to change)
5. CONCLUSION

As last-mile delivery is one of the critical points, there is a demand for a more optimized solution that reduces eCommerce owners' logistic cost and improve customer experience. The project's overall progress is acceptable and on schedule as the milestone, "Conduct a further study on the relevant topic", is completed. Other selected algorithms will be tested with the data set of Solomon before Apr. 2 2023. The algorithm's effectiveness will be evaluated by the total travel cost required for the delivery. Additional data set sources will be required if the project team cannot deduce the most optimized algorithm from the comparative results.

As this project mainly focuses on finding a route with the shortest distance and the minimum number of vehicles, the limitation of this model is that other delivery factors, such as multi-depots and penalties, are not included in this model. Therefore, an important direction for future investigation is to improve the constraints of the mathematical formation of VRPTW. More delivery constraints, such as the type of vehicle and goods, can be considered so that the model can be more adaptive to real-life scenarios. Also, it might be valuable to consider green emissions when evaluating the solution's efficiency, which can be significant for reducing pollution in logistics [28].
6. REFERENCE


7. APPENDICES

Appendix A: List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviations</th>
<th>Meaning</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>VRPTW</td>
<td>Vehicle Routing Problem with Time Windows</td>
<td>8</td>
</tr>
<tr>
<td>HKUL</td>
<td>The University of Hong Kong Libraries</td>
<td>9</td>
</tr>
<tr>
<td>JSON</td>
<td>JavaScript Object Notation</td>
<td>13</td>
</tr>
<tr>
<td>NP-complete</td>
<td>Nondeterministic polynomial-time complete</td>
<td>12</td>
</tr>
<tr>
<td>NP-hard</td>
<td>Nondeterministic polynomial-time hard</td>
<td>12</td>
</tr>
<tr>
<td>SSAP</td>
<td>self-storage-as-parcel model</td>
<td>8</td>
</tr>
<tr>
<td>TSP</td>
<td>Travelling Salesman Problem</td>
<td>12</td>
</tr>
<tr>
<td>VRP</td>
<td>Vehicle Routing Problem</td>
<td>8</td>
</tr>
<tr>
<td>3PL</td>
<td>third-party logistics</td>
<td>7</td>
</tr>
</tbody>
</table>

Appendix B: List of Symbols

<table>
<thead>
<tr>
<th>Symbols</th>
<th>Meaning</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>v</td>
<td>Number of vehicle route</td>
<td>10</td>
</tr>
<tr>
<td>$c_{ij}$</td>
<td>Travel cost (time) of going from vertex $i$ to vertex $j$</td>
<td>10</td>
</tr>
<tr>
<td>$s_i$</td>
<td>Service time at vertex $i$</td>
<td>10</td>
</tr>
<tr>
<td>$e_0$</td>
<td>Earliest start time of the depot</td>
<td>10</td>
</tr>
<tr>
<td>$l_0$</td>
<td>Latest return time of the depot</td>
<td>10</td>
</tr>
<tr>
<td>$e_i$</td>
<td>Earliest arrival time at vertex $i$</td>
<td>10</td>
</tr>
<tr>
<td>$l_i$</td>
<td>Latest arrival time at vertex $i$</td>
<td>10</td>
</tr>
<tr>
<td>$Q$</td>
<td>Vehicle capacity</td>
<td>10</td>
</tr>
</tbody>
</table>