uParcel

Project 40 | ISTD | Ryan Goh, Dily Lee, Wu Zheyu **| ESD |** Kan Yan Zhe, Lim Chee Chin, Jordan Sim

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

Background

uParcel is a **crowdsourced delivery platform** which connects customers who need items delivered with delivery agents, on an ad-hoc basis. uParcel aims to solve the last-mile delivery problem which has been known to be disproportionately costly.

Problem Statement

Scrolling through the uParcel app to select jobs is quite inefficient, especially during peak season when number of jobs increase drastically. During the Lunar New Year period, the number of unfulfilled jobs exceeded 114. As a result of this inefficiency, not all jobs in the system are cleared daily.

The main objective of the project is to allow delivery agents to **fulfil more jobs** and **reduce costs** at the same time.

Final Deliverables

Our team aims to develop an app designed with effective front-end and back-end systems. The **front-end** will be a user-friendly app which makes it easier for delivery agents to quickly accept the jobs most relevant to them by including features such as **advanced filters** and **map view of jobs**. The **back-end** system will be an integrated solution running in the cloud which has a **recommendation system** that suggests new jobs to agents, allowing them to earn more. All-in-all, we expect the systems to help delivery agents do their jobs more efficiently.

State-of-the-art solution

Travelling Salesman Problem

$$\min \sum_{i \in V} \sum_{j \in V \setminus \{i\}} x_{ij} c_{ij}$$

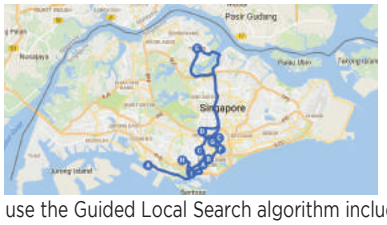
$$\begin{aligned}
\text{s.t.} \quad & \sum_{j \in V \setminus \{i\}} x_{ij} = 1, \quad \forall i \in V \\
& \sum_{i \in V \setminus \{j\}} x_{ij} = 1, \quad \forall j \in V \\
& y_{ij} + y_{ji} = 1, \quad \forall i \in V, \forall j \in V \setminus \{1\}, i \neq j \\
& x_{ij} \leq y_{ij}, \quad \forall i \in V, \forall j \in V \setminus \{1\}, i \neq j \\
& x_{ji} \leq y_{ji}, \quad \forall i \in V, \forall j \in V \setminus \{1\}, i \neq j \\
& x_{ik} \leq y_{ij}, \quad \forall i \in V, \forall j \in V \setminus \{1\}, i \neq j \\
& -(1 - x_{ik}) \leq y_{ij} - y_{kj} \leq 1 - x_{ik}, \quad \forall i \in V \setminus \{1\}, \forall j \in V \setminus \{1\}, \forall k \in V \setminus \{1\}, i \neq j \neq k \\
& x_{ij} \in \{0, 1\} \quad \forall i \in V, \forall j \in V, i \neq j \\
& y_{ij} \geq 0, \quad \forall i \in V, \forall j \in V \setminus \{1\}, i \neq j \\
& V = \{1, \ldots, N\}
\end{aligned}$$

Where x_{ij} denotes if node i immediately precedes node j, and y_{ij} denotes if node i precedes node j, not necessarily immediately.

Traveling Salesman Problem with Precedence Constraints or TSP-PC, is the **route optimization problem** that agents face when they have a set of jobs. The additional constraints is because agents are required to pick up items of each job before visiting the respective delivery locations.

Solving the TSP-PC means that we can suggest the ordering for the agents to take for them to **minimize their cost** (distance travelled).

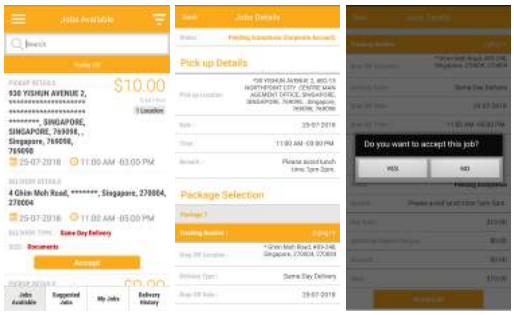
Google OR-Tools



We use the Guided Local Search algorithm included in OR-Tools which is able to find solutions to the TSP-PC with **sub-second computation times**, as compared to Genetic algorithm which we also experimented with.

Front End Development

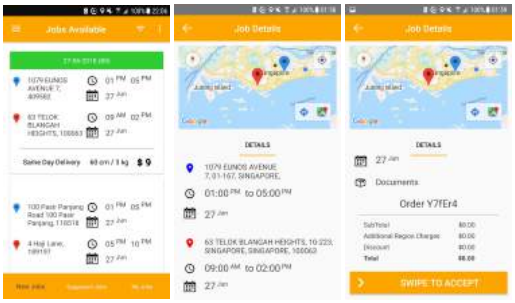
Before



Changes made in UI:

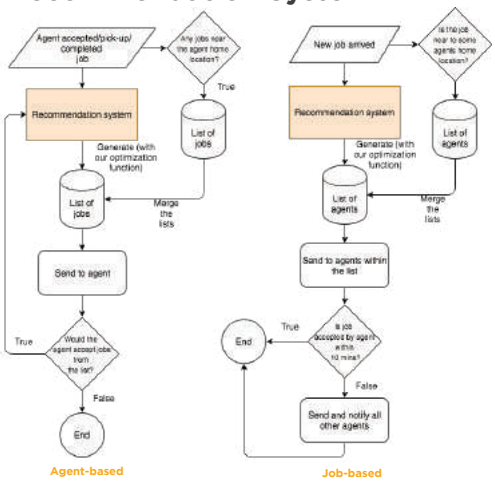
1. Removed unnecessary masked details of address
2. Rearranged job information. Include colored pins.
3. Added map showing pickup/delivery/current location
4. Swipe to accept job or change job status

After



Back End Development

Recommendation System



Optimization strategy

$$y = \frac{Revenue}{Cost^{\alpha}}$$

The objective function of the optimization problem has metric (y) - **earning efficiency** for the agents. The metric is simply a ratio of the **total revenue** gained from a set of jobs, and the **total cost** (distance travelled) which is the result from the TSP. We do this for the agent's current set of jobs, as well as a job that is not taken and see if the ratio improves. If it improves, we will recommend it to the agent. α is a parameter which determines the **relative importance** between revenue and distance travelled in a set of jobs.

Survey

To understand the delivery agents' perspective and to tweak our solutions to **address their pain points**, we got reponses from 49 delivery agents regarding their needs and what changes they would like to see. Being main stakeholders, the **49 agents account for 40% of current daily demand**. Our consolidated results are:

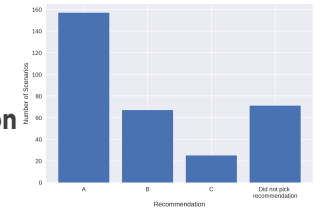
Relevant jobs	• Want jobs on the way and in the same area (home location)
Smarter app	• Want job recommendation system and bundle formation system
Responsive app	• Want to get information quickly for better decision making
Better UI/UX	• Want to look through job list with ease
More jobs	• With a better app, more customers will choose uParcel

Our main takeaway from the survey is to give the **same priority to UI and recommendation system** in our project.

Alpha Test - Phase I

We validated different aspects of our project. For **UI/UX testing**, the agents interacted with the app and gave written feedback. To **test our algorithm**, we gave them various job recommendations, asked for comments and recorded results - whether they choose to accept or not.

Validation of algorithm & recommendation system



From graph, **75% of all scenarios**, our recommended jobs are **accepted by agents**. 50% of the time, the most suitable job (according to objective function) recommended by the engine is accepted by agents. This indicates that our recommendation system algorithm is **applicable in reality**.

Alpha Test - Phase II

With the feedback from Phase I, we **improved the UI/UX** and **tweaked the algorithm**. To validate the functionality of the app, we conducted a focused group testing. Aspects of the test includes **integration testing** and **user acceptance testing.**

"The app is good, it is user-friendly. The layout of address shown is clean; blue and red pins are good. I like the map feature in the Job Details page!"

Delivery agent A

"The recommended jobs are appropriate. 4 out of 5 stars. Perhaps a filter for proximity and price will be good."

Delivery agent B

Possible Improvement

Collect data to **identify threshold** for y^* to recommend a job.

The parameter α of the objective function can be trained using **machine learning techniques**.

By varying α and y^* and measuring corresponding recommended job acceptance (hit/miss) rate, the performance of the algorithm can be modelled using linear regression as a function of α or y^*.