CS 2XB3 – Group 2 – Ridesharing Taxi Scheduler

Kuir Aguer, Brendan Duke, and Jaeden Guo

1. The Application Domain: Stakeholders

The stakeholders in the Ridesharing Taxi Scheduler application include taxicab passengers, taxi drivers, management of the taxicab companies, environmental organizations and public transit authorities.

- The goals of taxicab passengers in using the Ridesharing Taxi Scheduler software would be to have public transit that is firstly affordable. Secondly, passengers would want both a short time from playing their query to being picked up and also a short overall trip time from their origin to their destination. Passengers would also want the cost of sharing a ride among multiple people to be split fairly amongst all the people.
- Taxi drivers want to be able to minimize idle time so that they can be constantly earning money, hence drivers should be able to go from dropping off a passenger to picking up the next passenger with as little down-time as possible.
- Taxicab companies are concerned with the amount of profit that they can make. Therefore it is in the interest of taxicab companies to have a high ratio of number of satisfied queries to the number of drivers that the company has to employ (and therefore pay). The taxicab company also wants to be able to gather feedback so that they can further improve their service. The taxicab company also wants to be able to gather data about the volume of pickup requests on a per-region basis so that it can allocate the correct number of taxicabs to satisfy the number of requests in each region.
- Environmental organizations want to reduce the impact of running a taxicab service on the environment. Therefore it's in the interest of environmental organizations if Ridesharing Taxi Scheduler reduces the total distance travelled by all taxicabs using the software.
- The goals of public transit authorities are to be able to regulate public transit. Therefore public transit authorities would want Ridesharing Taxi Scheduler to be implemented in a way that makes it easy for them to ensure that regulations are being followed. If, for example, the Ridesharing Taxi Scheduler became available to be used by the public to have a hitch-hiking ridesharing system, that would conflict with the public transit authorities' interests because of the difficulty of regulating such a hitch-hiking system.

2. Functional Requirements

One important function in the Ridesharing Taxicab Scheduler takes in a pickup query and determines which taxicab will meet the query based on the requirements written below.
For a given query (request to be picked up by a taxicab and dropped off somewhere) Q:
function PICKUP(Q)
Input: Query time Q.t, query pickup point (origin) Q.o, query delivery point Q.d, delivery time window of query Q.wd.
Output: Taxicab ID, estimated pickup time t_pickup, estimated dropoff time t_dropoff, along with an updated taxicab schedule for that taxicab.
There are functional requirements on PICKUP's scheduling of a taxi to meet any given pickup request.

- Any pickup request should be met within a certain time limit given by the constant PICKUP_TIME_WINDOW, i.e. t_pickup <= Q.t + PICKUP_TIME_WINDOW. This value is set to 5 minutes as default.
- The dropoff time for the passenger should be within the delivery time window, i.e. t_dropoff <= Q.wd.
- The dropoff times in the new taxicab schedule of the other passengers already in the taxicab at pickup time of the new query Q should all be within their respective delivery time windows.

3. Non-Functional Requirements

Ridesharing Taxicab Scheduler will require three different interfaces: one for the passenger, one for the driver and one for a dispatcher.

- The passenger interface should allow customers to input a pickup location and time, along with a dropoff location. The interface should display for the customer the ID of the taxi that is going to pick them up, along with their estimated pickup and dropoff times.
- The driver interface should display his schedule of passengers: those that are already in the car as well as any new queries. The driver interface should also display for the driver his route on a map.
- The dispatcher interface should display all the current taxi locations. This interface should allow the dispatcher to select any given taxicab, displaying that taxicab's current schedule and route.
- With respect to portability, as much as possible of the Ridesharing Taxicab Scheduler should be platform-independent in the sense that it would not have to be re-written to implement the application on a different platform.

4. Requirements on the Development and Maintenance Process

Unit testing will be done to verify that each module implements the requirements corresponding to that module. For example, it should be verified that the PICKUP function picks each query up within PICKUP_TIME_LIMIT from the query time Q.t, and drops each query off within its respective dropoff time window Q.wd.

Furthermore, the effectiveness of the ridesharing algorithm will be compared against the same set of inputs (pickup queries) if no ridesharing were used. The amount of distance travelled, as well as time per pickup will be compared.

During development, a revision control system will be used: git. Files will be hosted at https://www.github.com/dukebw/, and tracked using git.

Parts of the Ridesharing Taxicab Scheduler software that might change would be the data structures used to store the geographical location and schedules of taxicabs. Furthermore, the algorithm used to select which taxicab will be given a given query could change, if for example a better algorithm were discovered. Therefore the data structure holding taxicab information and the pickup-query-satisfying algorithm should be modularized so that they can be easily changed during the development process.