1 Introduction

The feature of sharing trips among riders has been available in ride-sharing apps like lyft and uber. Matching riders with similar or part of similar routes can allow drivers to deliver more riders given the same timeframe, which boosts their financial gains. The problem is to match riders to form combined trips so that the total distance travelled by all riders is minimized. We consider the case where maximum number of riders that are allowed to be matched is two. We first formulate it as a minimum weight matching problem and transform it into a minimum weight perfect matching problem. Then we use Edmond's blossom algorithm to solve it. We apply this approach to New York taxi data set and provide insights into properties of the minimum-weight matching in the numerical study.

2 Problem Formulation

Given a set of single routes. Each route is characterized by its start location and end location that are represented by pairs of (latitude, longitude) coordinates. There is no time window limit on the waiting time and travel delay experienced by riders. The distance is the only consideration in this problem. If two riders are matched, an order of pickup and drop-off locations is feasible if the pickup location of a rider precedes the corresponding drop-off location of that rider. We choose the pickup and drop-off order that minimizes the total distances traveled as the route of the combined trip. The total distance traveled in this combined trip is the sum of the distances between each location. If a rider is un-paired, then the travel distance is the distance of his single trip. The aim is to find a matching assignment for all riders such that the total distance is minimized.

3 Method

We can build a graph $G = (S, E, w)$ for the matching problem of riders. The idea follows from [4]. The set of nodes S is the set of rider-id's. For any pair of riders i, j, depending on the order in which the are visited, there are four possible routes of a combined trip shared by rider i and rider j. Denoting the start and end locations as o and d, the four possible routes are $o_i - o_j - d_i - d_j$, $o_i - o_j - d_j - d_i$ and $o_j - o_i - d_i - d_j$ and $o_j - o_i - d_j - d_i$. If there exists a route the distance of which is smaller than the sum of separate routes of rider i and j, the minimum distance of such routes is denoted as c_{ij}. The distance traveled in route of the single trip for rider i is denoted as c_i. The set of edges $E = \{(i, j) \in S \times S | c_{ij} < c_i + c_j\}$ is a set of unordered pair (i, j) of riders such that the total distance of their combined trip is less than their separate trips. The weight $w_{(i,j)}$ of edge (i, j) is $w_{(i,j)} = c_{ij} - c_i - c_j$.

Let T denote the set of feasible trips. A set of trips is feasible if for every rider $i \in S$, there exists a unique trip in set T, which can be single or combined with some other rider. Let $T_2 \subseteq S \times S$ be such that for any $(i, j) \in T_2$ and $(i, h) \in T_2$, $h = j$. That means for any two edges in set T_2, they are not incident to the same node. In other words, for every node $i \in S$, there exists at most one edge $e \in E$ in set T_2. This implies that T_2 is a matching in the graph $G = (S, E, w)$. Let $T_1 = \{i \in S | \nexists j \in S \text{ such that } (i, j) \in T_2\}$. Then set $T = T_1 \cup T_2$ is such that for any $i \in S$, either $i \in T_1$ or there exists a unique $j \in S$ such that $(i, j) \in T_2$.

The total Manhattan distance C traveled in all trips in a feasible set of trips T is given by

$$C = \sum_{i \in T_1} c_i + \sum_{(i,j) \in T_2} c_{ij} = \sum_{i \in T_1} c_i + \sum_{(i,j) \in T_2} (c_i + c_j) + \sum_{(i,j) \in T_2} (c_{ij} - c_i - c_j). \qquad (1)$$

For any $i \notin T_1$, there exists a unique $j \in S$ such that $(i,j) \in T_2$. For any $i, j \in S$ such that $(i,j) \in T_2$, $i \notin T_1$ and $j \notin T_1$. Therefore

$$\sum_{(i,j) \in T_2} (c_i + c_j) = \sum_{i \notin T_1} c_i \qquad (2)$$

Since $T_1 \cup T_1^c = S$,

$$C = \sum_{i \in S} c_i + \sum_{(i,j) \in T_2} (c_{ij} - c_i - c_j) \qquad (3)$$

Given the first term in (3) is constant, finding optimal feasible set T that minimizes the total distance C is equivalent to finding a minimum-weight matching T_2 in the graph $G = (S, E, w)$. There have been some very efficient algorithms, e.g., Edmond's blossom algorithm to solve maximum-matching problems. We use the algorithm available at [2] to solve the minimum-weight matching in the graph G. Before this, we first define another graph G' as the algorithm available at [2] is applicable to minimum-weight perfect matching problems. Let $\tilde{G} = (\tilde{S}, \tilde{E}, \tilde{w})$ be a copy of graph G, that is, $\tilde{S} = S$, $\tilde{E} = E$ and $\tilde{w} = w$. Then consider the graph $G' = (S', E', w')$, where the set of nodes $S' = S \cup \tilde{S}$, the set of edges $E' = E \cup \tilde{E} \cup \{(i, \tilde{i}) \in S \times \tilde{S}\}$. The weight function w' of G' is defined as $w'_{(i,j)} = w_{(i,j)}$ for $(i,j) \in E$, $w'_{(i,j)} = \tilde{w}_{(i,j)}$ for $(i,j) \in \tilde{E}$ and $w_{(i,\tilde{i})} = 0$ for $i \in S$ and $\tilde{i} \in \tilde{S}$. The basic idea is to solve the minimum-weight perfect matching M' for the graph G' and show that M' restricted to the graph G is a minimum weight matching for the graph G. Such procedure can be found at [1]. Next we show two claims.

Claim. *For every matching in G, there exists a perfect matching in $\tilde{G}$.*

Proof. Given a matching M in the graph G, let $\tilde{M}$ denote the corresponding matching in G'. Consider matching $M' = M \cup \tilde{M} \cup M^n$, where M^n is the set of edges $e_{(s,\tilde{s})}$ where vertex $s \in S$ is unmatched under M in G. Matching $\tilde{M}$ is perfect for $\tilde{G}$. $\qquad \square$

Claim. *For every minimum-weight perfect matching in $\tilde{G}$, the corresponding matching in the graph G is a minimum-weight matching for the graph G.*

Proof. Suppose M' is a minimum-weight perfect matching in G'. $M' = M \cup \tilde{M} \cup \{(i, \tilde{i}) \in M' | i \in S, \tilde{i} \in \tilde{S}\}$. The matching M in the graph G and the matching $\tilde{M}$ in the graph $\tilde{G}$ has the same total weight . If not, a contradiction to the minimum-weight perfect matching M'. Then the matching M in G is a minimum-weight matching for the graph G. If not, there exists a matching $\hat{M}$ that achieves smaller total weight than the matching M does in the graph G. Copy $\hat{M}$ from G to $\tilde{G}$ and let the corresponding matching in $\tilde{G}$ denoted by $\tilde{\hat{M}}$. $\tilde{\hat{M}}$ achieves smaller weight than the matching $\tilde{M}$ in the graph G. For the remaining node s that is unmatched under $\hat{M}$, match s with the corresponding node s' in $\tilde{S}$ and let M^x denote the set of them. Then the matching $M^t = \hat{M} \cup \tilde{\hat{M}} \cup M^x$ is perfect and achieves a smaller total weight than the matching M', a contradiction. $\qquad \square$

Following the above steps, we can find a minimum-weight matching T_2 for the graph G. Let set T_1 be the set of nodes that do not have an incident edge in the matching T_2, i.e, set T_1 is the set of unmatched riders. Then $T_2 \cup T_1$ is a feasible set of trips that minimizes the total distance of routes.

4 Numerical Study

4.1 Data Set

The dataset we used is the New York taxi dataset for January, 2015, available at `https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page`. We used the first 10000 rows for this project. First we check whether there are requests with missing data or requests that start and end at the same locations. After this has been done, there are 9751 number of riders left. We visualize the requests data by plotting the start location and end locations in Figure 1. Most of routes are cluttered in the narrow middle area.
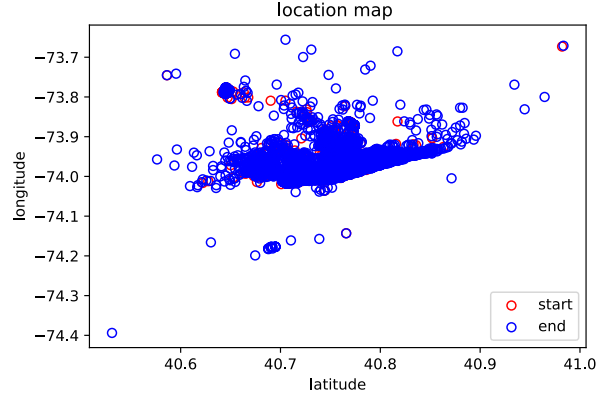

Figure 1: map of start locations and end locations

Figure 2 shows the histogram of distance traveled in routes for all riders. Most riders have travel distance less than 10. A further look at the statistics at Table 1 shows that 0.75% of routes have travel distance less than 4.67. This means that most routes are short-distanced.

Min	0.25 Quantile	Median	Mean	0.75 Quantile	Max
0.000321	1.555883	2.625406	4.251945	4.667862	69.001952

Table 1: Summary statistics of distance of routes

4.2 Results

Using the approach described in Section 3, we construct a graph with 9751 nodes and 6760284 edges. Edges with weight whose absolute value less than 10^{-6} were not added to the graph. In order to solve the minimum-weight matching for the graph, we create an auxiliary graph
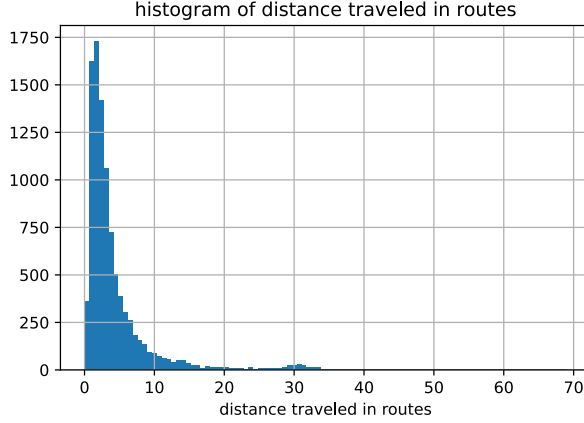
Figure 2: histogram of distance traveled in routes

	Min	0.25 Quantile	Median	Mean	0.75 Quantile	0.9 Quantile	Max
pickup	0.000000	0.139507	0.287112	0.398746	0.479957	0.798925	12.548913
drop-off	0.000000	0.161482	0.297184	0.450719	0.499501	0.832800	33.685735

Table 2: the statistics on the distance of pickup locations and drop-off locations between a matched pair

and use it to solve the minimum-weight perfect matching using code in [2]. The edges and weights of the auxiliary graph is stored in "input.txt" and the matching result is stored in "mat.txt". The total distance traveled in all routes is 18415.81. There are 4804 pairs of riders and 143 riders unmatched. The routes is attached in "matches.csv" file.

We look at the statistics of matched rides, which can be found at Table 2. We can see that most paired rides have close proximity regarding the pickup locations and drop-off locations. Therefore, it is unlikely to match riders whose distance between the start locations or between end locations is larger than the distance of their single trips together. This actually indicates when searching for combined trips to save distance for a rider, one can limit the search space to riders who are within a close distance to the rider's start location or the rider's end location.

Next we evaluate the impact of distance traveled in routes where riders travel alone in matching decisions. For this purpose, we present the distance traveled in separate routes for paired riders in Figure 3. There seems to be strong positive correlation between distance of separate routes if paired riders travel alone. The correlation coefficient r of the distance traveled in route of rider 1 and the distance traveled in route of rider 2 is 0.975. If we assume that the travel distance of riders follow a normal distribution, the t-statistic of the correlation r between distance in two routes is $t = \frac{r\sqrt{n-2}}{\sqrt{1-r^2}} = 295.54$ given $n = 4804$. which indicates that r is highly significant. It can be concluded that the matching algorithm tends to match riders whose routes are of the same length together.

Lastly, we use the difference between the sum of distance of separate routes and distance of the combined route divided by the sum of distance of separate routes to measure the effect of matching on the distance traveled. For example, for riders i, j in a pair, the above metric C_p is $C_p = (c_i + c_j - c_{ij})/(c_i + c_j)$. Out of 4804 pairs of riders, the maximum saved distance

4

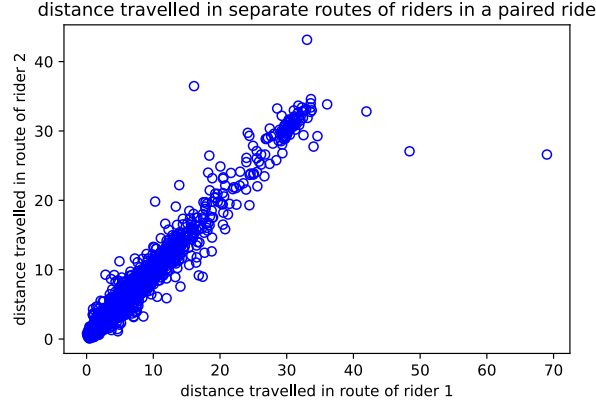travelled in separate routes of riders in a paired ride 2.pdf

Figure 3: distance traveled in separate routes of riders in a paired ride

$C_p = 0.5$, the mean saved distance $C_p = 0.42$ and the percent of routes with saved distance C_p higher than 0.4 is 74%. Therefore we can conclude that matching helps save distance traveled in routes efficiently.

5 Discussion and Conclusion

We solved a matching problem where riders are paired such that the total travel distance of routes is minimized. We formulate this problem as a minimum-weight matching problem and solve it with the blossom algorithm. We apply the method to the New York taxi dataset. The numerical study shows that the paired riders have close proximity in pickup and drop-off locations and the routes of paired routes are of the same length.

There are several limitations to this project. The number of riders to be matched is 9751 in this data set. The algorithm presented in this report fully explore all possible routes in building the graph and optimizing matching decisions, which may not be possible for data sets of larger size. When searching for possible combined route for a rider, one can limit the search space to its neighbors as concluded from table 2. One way to achieve this is to sort riders based on their start and end locations, and then find possible routes among the close neighbors. Another approach is to develop a graph partitioning model that divides a graph into smaller components so that only riders within the same component are considered for matching. Then one can solve the optimal matching for each component. Some machine learning techniques like K-nearest neighbors could be used to divide the graph into clusters of locations.

This work can be extended in following ways. In real applications, pickup and delivery are done within a pre-specified time window to ensure quality of service. Adding this constraint into the matching problem would reduce the number of possible routes that could be explored. Second, real-time requests arrive sequentially, so an online matching algorithm is more appropriate to handle incoming requests as well as temporarily unmatched riders on the fly. When a new rider arrives, it may create better matching choice than pre-matched pairs in terms of minimizing distance. Therefore the algorithm has to reevaluate routes

previously assigned and take the service quality constraint for riders who have been picked up into account. The detailed extension to the dynamic matching algorithm that iteratively matches riders in batch can be found at [3].

References

[1] `https://homepages.cwi.nl/~schaefer/ftp/pdf/masters-thesis.pdf`, Accessed: 2024-01-21.

[2] *Blossom algorithm code*, `https://pub.ista.ac.at/~vnk/software.html#BLOSSOM5`, Accessed: 2024-01-21.

[3] Javier Alonso-Mora, Samitha Samaranayake, Alex Wallar, Emilio Frazzoli, and Daniela Rus, *On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment*, Proceedings of the National Academy of Sciences **114** (2017), no. 3, 462–467.

[4] Paolo Santi, Giovanni Resta, Michael Szell, Stanislav Sobolevsky, Steven H Strogatz, and Carlo Ratti, *Quantifying the benefits of vehicle pooling with shareability networks*, Proceedings of the National Academy of Sciences **111** (2014), no. 37, 13290–13294.