MASTER SEMESTER PROJECT

Comparing Rebalancing Strategies in Car Sharing Systems

Xinling Li

May 31st 2022

Lead Teacher
Prof. Michel Bierlaire
Supervisors
Mrs. Selin Atac, Mr. Nicola Marco Ortelli

TRANSP-OR - Transport and Mobility Laboratory

Contents

1 Introduction

Car sharing has gained more and more popularity in recent years, especially in large cities, where the congestion and parking cost make it less attractive to own a private car. In big cities, people rely on public transport to commute and only need cars for special purposes, which creates a context for shared cars to thrive. By using shared cars, people do not need to take care of the maintenance and parking fees to have the choice of moving by car. Generally, there are two types of car sharing services with respect to the trip types:

- **Round-trip car sharing**: people need to return the car back to the same station where they picked it up.
- **One-way car sharing**: people can return the car back to any station that is the most convenient for them.

There is another more complicated system named as free-floating car sharing system, in which there are no fixed stations and the users can park the car wherever they want. Since this type of system can be regarded as a one-way car sharing system with a large amount of stations, the rebalancing problem in free-floating system can be solved by the ideas from one-way system.

Round-trip car sharing is very similar to car rental, while one-way car sharing ensures higher flexibility for the users. Since the flexibility of the car sharing system is one of the most important attributes that makes it stand out, we are interested in the operation and management of this type of system.

The flexibility of return stations makes one-way car sharing more appealing, but also makes it harder to operate and maintain. Due to the asymmetry of pick-ups and drop-offs across different stations, the vehicle imbalance problem constantly arises. Therefore, operations need to be carefully planned and carried out to assure the system's long-term viability, where the term "viability" refers to both the users' satisfaction rate and the operator's profitability. Generally, there are three levels on which we can make decisions to solve the vehicle imbalance problem: strategic, tactical, and operational. The first two decision levels are mainly concerned with the decisions on system construction. For example, on the strategic level, the number of stations and their location are to be decided, and on the tactical level, fleet size is to be decided. The operational level is mainly concerned with real-time decisions. At this level, the operators frequently make use of rebalancing to solve the distribution imbalance problem. When doing the rebalancing, operators attempt to decide the optimal rebalancing scheme with a known system configuration to make the vehicle distribution more consistent with users' trip demand pattern.

There are two types of rebalancing methods, distinguished by operating hours, named static and dynamic rebalancing. Strategies assuming that rebalancing is carried out when the vehicle usage rate is negligible (e.g. at night) are called static, while if a significant amount of usage is taken into consideration, it is called dynamic. The ideas behind the car sharing problem is very similar to that of bike sharing rebalancing. The difference is that in car sharing rebalancing, except for car rebalancing, we also need to consider staff relocation. Another difference is that in car sharing systems, staff and cars are always a one-to-one match (sometimes multiple staff to one vehicle, see in Zhao et al. (2018)). The assignment of staff to specific vehicles in a cost-effective way makes the modeling of the rebalancing problem in car sharing more complex than in bike sharing systems.

The rebalancing problem in car sharing systems has already been a research hotspot in recent years. The basic ideas behind all approaches can be classified into two categories: optimization-based and simulation-based. The problem can also be classified as user-based and operator-based. A user-based method intends to make the users do the rebalancing. The operators provide an appropriate amount of bonus for those users who are willing to pick up or drop off the vehicles in the recommended area. For an operator-based method, the operators make the rebalancing schemes and hire staff to carry them out. In this report, we concentrate on operator-based strategies in one-way station-based car sharing rebalancing problem. In Section 2, we first do a literature review on this topic. We then talk about the data processing in Section 3. Then in Section 4, we select and present two strategies and the two models from the reviewed papers, implement them and compare their performance. We then propose a new strategy based on the first strategy. We analyze the results of the numerical experiments in Section 5 and finally conclude in Section 6.

2 Literature review

As mentioned in Section 1, we only talk about operator-based rebalancing models in car sharing systems in this report. Readers who are interested in simulation-based or user-based models can refer to the comprehensive review papers by Ataç et al. (2021), Illgen and Höck (2019), and Brandstätter et al. (2016). In this section, we organize the review according to the operating hours the rebalancing problem considers: dynamic rebalancing or static rebalancing.

We first discuss papers on dynamic rebalancing. Bruglieri et al. (2019) propose a network built on request pair. In this paper, it is assumed that the delivery requests have been known in advance, which means that we already know the number of vehicles needed to be moved or delivered to a station by staff. In the request-based network, each node is characterized by the station number, residual charge of the battery and a time window when it is allowed to carry out this delivery. The staff travel between these nodes to maximize the requests served respecting the time window and battery charge level constraints. The authors then present three algorithms to solve this electric vehicle relocation problem.

To make the above request-based model work, another model that can predict the imbalance state of the system to predict the delivery request is required. However, the imbalance state information is not always available, so this model may fail for some systems. A more general way to model the system is by using time-space network. A time-space network only requires stations' location and time horizon to build.

Ait-Ouahmed et al. (2018) consider a multi-objective routing problem based on the time-space network. The authors take the number of used vehicles and staff, sum of traveling distance and number of unsatisfied trip demand into consideration when defining the objective function. The authors then model the rebalancing problem as a vehicle routing problem without staff relocation. The structure of the model is very straightforward, making it easy to understand and implement, but without staff relocation, the validity of the rebalancing scheme cannot be guaranteed. The vehicle rebalancing scheme always needs to be carried out by staff, so it is only valid if we can make sure that there are valid schedules which can be assigned to the staff. What's more, without staff moving schedules, the rebalancing scheme is useless for car sharing operators.

To provide a more comprehensive solution, some researchers start to incorporate both vehicle rebalancing and staff relocation into the models. In Nourinejad and Roorda (2014), the authors only concentrate on vehicle rebalancing and parking inventory optimization without staff relocation. But in their more recent work, Nourinejad et al. (2015), the authors model the rebalancing problem as a multiple traveling salesman problem(TSP). Two TSP network are proposed, one for vehicles and the other for the staff.

Gambella et al. (2018) propose a comprehensive model framework that can output reasonable rebalancing and relocation schemes and initial distribution. This model intends to maximize profit with rebalancing. Based on the target distribution from the dynamic rebalancing model, a static rebalancing model is also proposed to compute the rebalancing needed to be carried out during non-operating hours at night to achieve this target distribution. Then a heuristic algorithm is developed to solve these two models. It is compared with that built-in in CPLEX and demonstrated to have a better performance.

Zhao et al. (2018) also propose a modeling framework that specifically takes staff relocation into account. In this paper, the authors use a time-space network to represent two sets of paths for both vehicle rebalancing and staff relocation in one model. The framework is essentially formulated as a multi-vehicle routing problem. This model is built based on the assumption that user reservations and station locations are known in advance and station capacities are sufficient to serve all the reservations (which is not usually the case). The objective is to minimize the cost to serve all reservations. The model is easy to extend and interpret, but this framework has a high number of variables and constraints which makes it hard to solve even a median size system with 10 stations, 30 trips and 30 vehicles by CPLEX.

The problem of long computation time is brought by the essence of time-space network. The number of time-space pairs grows rapidly with the problem size, but these time-space pairs are also indispensable in this kind of network. To resolve this dilemma, Boyacı et al. (2017) build a time-space network on the clustered stations. The aim of the clustering is to reduce the number of variables. The workflow first creates station clusters, then solves the operations optimization model and finally uses the staff flow model to create personnel assignments. Because of the reduced number of stations, the model achieve a shorter computation time compared to the original size.

While dynamic rebalancing is more flexible and can better adapt to the uncertainty of the system demand than static rebalancing, static rebalancing allows the rebalancing fleet to travel swiftly in the city without contributing to traffic congestion and parking problems. Static rebalancing is a major research subject in bike rebalancing problem. There is already a systematic way of modeling static rebalancing problem in bike sharing systems, with the major ideas adopted from one commodity pickup and delivery, traveling salesman problem and vehicle routing problem (Chemla et al. (2013), Raviv et al. (2013)). In bike sharing static rebalancing modeling, the authors always assume prior knowledge of the target distribution, which needs to be modeled.

There is not much research that concentrates on static rebalancing in station-based car sharing compared to bike sharing. The difference between the two types of system is that a big number of bikes can be moved using a single (or a few) little truck(s), while each car needs to be moved by staff in a car-sharing service. Therefore, it is not straightforward to adapt the ideas behind static bike sharing rebalancing to car sharing system. In some of the few papers regarding static rebalancing in car sharing systems, the authors concentrate on developing models to decide target distribution, and the strategies are always based on charging process and cost for electric cars (Caggiani et al., 2021), but the charging constraints are out of the scope of this report.

3 Data processing and analysis

3.1 Data description and preprocessing

The data we use is an open-source anonymized trip dataset of 2 months (from September to November 2017) in the city of Turin, Italy (Luca et al., 2019). Each trip contains: origin and destination geographic coordinates and start and end timestamps. There are 122,016 trips in total, with a daily average of more than 2000.

While the longest distance of one trip is less than 20 km, there are trips that last several hours, which may exist because of several trips in one reservation. For example, a user first drives to the supermarket, shops for some hours and drives back. There are also trips that last slightly longer than 1 minute, which may be due to temporary schedule changes or data noise. Since we only concentrate on demand between stations without internal stop, and the longest distance between two points in the dataset is about 20 km, we first filter the trip data by only keeping trips with a duration between 5 minutes and 2 hours. After filtering, there are 120,854 trip data left.

3.2 Data processing

The trip data is collected from a free-floating car sharing system. As mentioned above, we focus on the rebalancing problem in a station-based system. To use the data, we need to determine the location of the stations first. In the following part, we discuss three distinct clustering methods: DBSCAN, K-means and K-medoids. Since the whole dataset is too large and we assume that the demand pattern does not differ much between weeks, we base the clustering on the data from the first Monday to Friday in September. The distribution of the data points is shown in Figure 1.

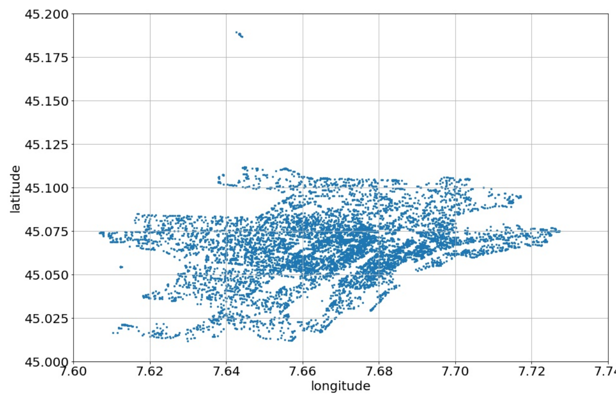

Figure 1: Distribution of data points

3.2.1 DBSCAN

DBSCAN is a density based clustering algorithm. It views clusters as areas of high density separated by areas of low density. The advantages of DBSCAN are that the number of clusters does not need to be pre-defined and that it is insensitive to outliers.

There are two important parameters to decide for DBSCAN: the minimum number of points around one point to label it as a core point, and the radius of the neighbor area around one point. We assume that 500m is a reasonable maximum walking distance, so we set the radius of neighbor as 500m. For the number of points around for one point to be labeled as a core point, we try different values from 50 to 200 and it turns out that with 100, there is a maximum number of clusters of 5. Therefore, with radius of neighbor 500m, number of points 100, we get the clustering result shown in Figure 2. There is one cluster covering the majority of the area. This is not reasonable for us as we cannot only pick one position as a station within this big cluster to cover that large area. This problem exists because in the dataset, most of the data points concentrate with similar density within a big area. Therefore, with DBSCAN, these points will always be clustered together. Moreover, compared to the original distribution of data points in Figure 1, we can see that many points disappear in Figure 2. This is because they are labeled as noise points, and this means that we cannot get the label for all the points we want by DBSCAN.

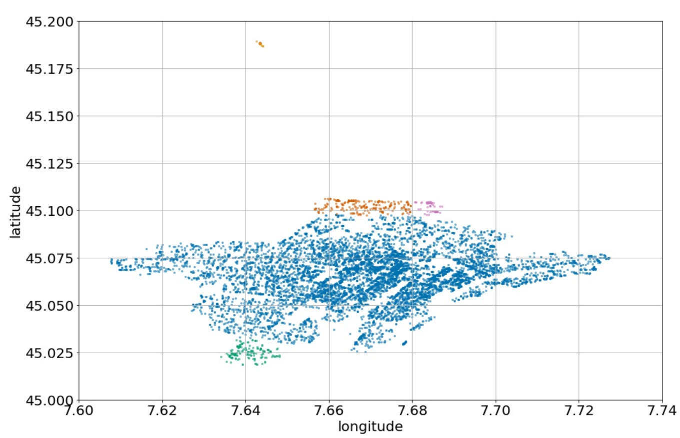

Figure 2: DBSCAN

3.2.2 K-means

K-means algorithm clusters data by separating points in groups of equal variance, minimizing within-cluster sum-of-squares (WCSS). Different from DBSCAN, this algorithm requires the number of clusters as an input parameter. Generally, the optimal number of clusters for a dataset is decided based on an elbow plot, which indicates the relationship between the number of clusters and WCSS. As WCSS always decreases with the increase in number of clusters, the elbow graph helps avoid overfitting. However, in our problem, the number of clusters depends on the number of stations we want when solving the models. Therefore, we do not use the elbow method. Instead, we set the number as required by the case study.

To ensure a reasonable complexity of the models so that most of the numerical instances can be solved within one day, we need to control the number of stations in the system. We test the solving time with 20 stations, but most of the instances cannot be solved within the time limit. Therefore, we cluster the points into 10 clusters, and we discuss more about the complexity for 10 stations in Section 5. The result of the clustering is shown in Figure 3. Although the size of all the regions is similar and all the points are assigned to a centroid within a reasonable distance, there exists some centroids in the lower left area, that do not have any data points around, which might mean that this region is not accessible for cars. Therefore, it is not realistic to pick this centroid as a station.

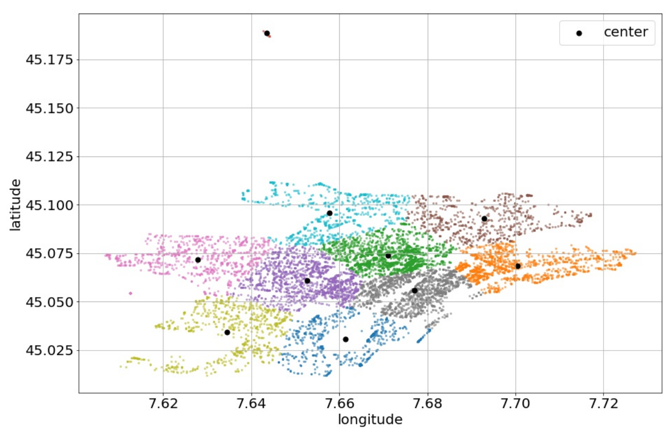

Figure 3: K-means

3.2.3 K-medoids

K-medoids algorithm is very similar to K-means. The only difference is that while K-means computes the mean of all data points in one cluster as the centroid, which is not necessarily one of the input data points, K-medoids chooses an actual data point as a centroid, thereby allows for greater interpretability of the cluster centers. This way, K-medoids can resolve the problem of unrealistic centroids by always ensuring that the centroid can be accessed by cars. With 10 clusters, the result of K-medoids can be seen in Figure 4.

Since DBSCAN works better for dataset with variance of density, which is not the case for our dataset, and K-means will possibly produce inaccessible centroids, we finally choose to use the centroids from K-medoids as the location for 10 stations. After clustering, we assign each geographic point to its nearest station and calculate the distance and traveling time between stations by Google Maps API.

After this step, we get the trip data in the form of origin and destination station numbers, start time, traveling distance and traveling time.

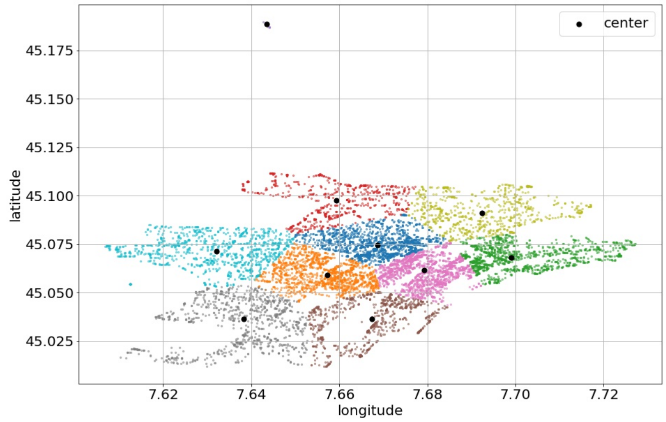

Figure 4: K-medoids

3.2.4 Data analysis

Based on the trip data from clustering, we analyze the usage pattern within one work day for all the 10 stations. The analysis focuses on how pick-ups, drop-offs and accumulated fleet size change over 24 hours per station. We pick the trips from Wednesday, September 13th, 2017. There are 2181 reservations in total during the whole day.

The fluctuation of number of pick-ups and drop-offs for each station is shown in Figure 5. The number in the upper right corner of each graph indicates the station number. From the figure, we see that stations 3, 4, 8 have a much lower number of pick-ups and drop-offs than that of the others. Station 0, 1 and 6 have a much higher value. The position of these low and high demand stations is shown in Figure 6. The low demand stations are distributed far from the region center and high demand stations are located around it, which is intuitive.

It can also be seen from Figure 5 that while most of the stations have synchronous pick-ups and drop-offs pattern, station 0 has the opposite: at station 0, there are few pick-ups when drop-offs are high and a high number of pick-ups when drop-offs are low. This indicates that the number of vehicles at station 0 can rebalance itself during a one-day period.

The number of vehicles at each station is shown in Figure 7. Since we do not have the initial distribution information, and we only concentrate on the fluctuation pattern of the available vehicles but not the absolute value, we assume that the number of available vehicles at each station is 0 at the beginning of the day. We then accumulate the number of pick-ups and drop-offs to get this figure. We define a station that has a positive accumulated number for the most of the time as a drop-off station because the positive numbers indicate that the overall number of drop-offs exceeds pick-ups. Similarly, we define a station that has a negative accumulated number for the most of the time as a pick-up station. From the figure, we see that there are pick-up stations such as 2, 7 and drop-off station as 0, 6. The different fluctuation patterns across the stations verify the possibility of doing rebalancing to satisfy more demand with limited fleet size. For example, rebalancing vehicles from station 0 to station 9 can help serve the pick-up demand from station 9 without placing more vehicles at station 9 at the beginning.

4 Methodology

In this section, we first talk about the two different strategies that we implement from the papers: one assumes that we need to serve reservations so as to maximize profit and the other assumes that all the reservations

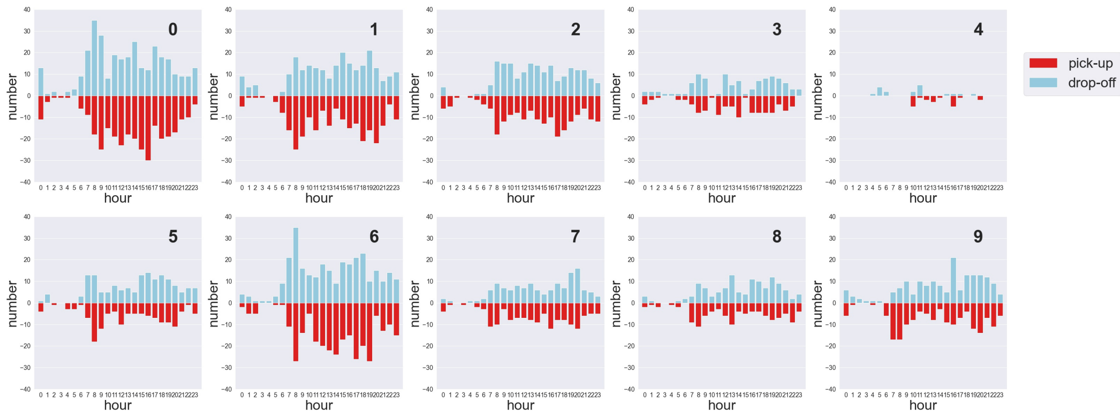

Figure 5: The number of pick-ups and drop-offs for one day

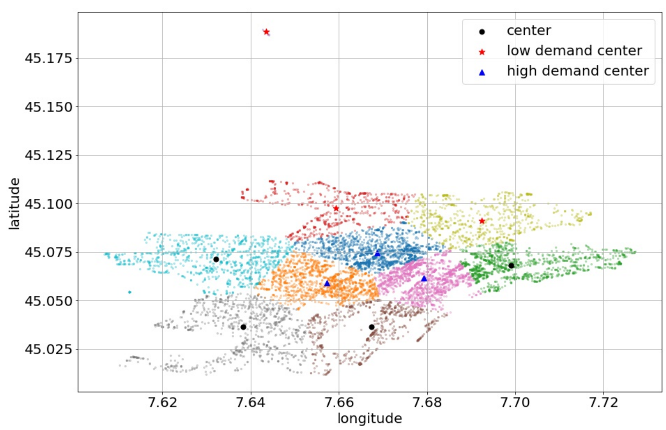

Figure 6: Low demand and high demand stations

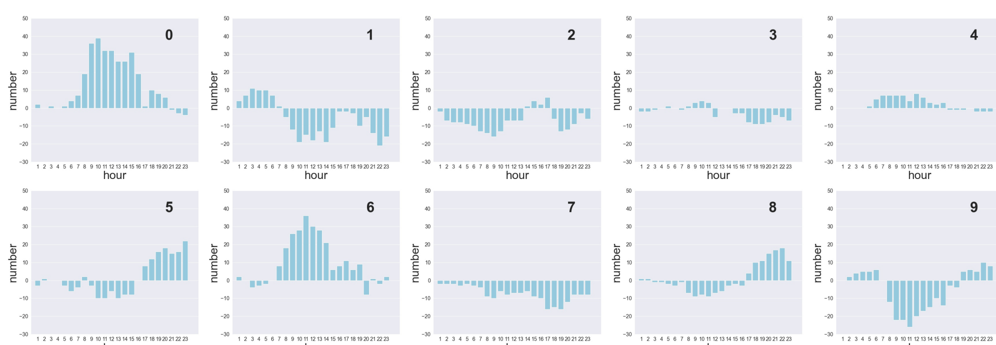

Figure 7: Number of vehicles

need to be satisfied. We first present the details about these two strategies and introduce the models based on them. Then, based on the first strategy, we introduce a third strategy, which assumes that a pre-defined set of prioritized trips must be served. By solving the three models with input data from Section 3, we finally analyze the results.

4.1 First strategy

The objective of the first strategy is to satisfy the subset of reservations that maximizes the system's profit. The key assumptions for this strategy include:

- All the customer trips are reserved in advance.
- Each station has limited capacity.
- The number of vehicles and staff are known. In other words, it is a system with fixed configuration.
- When the staff are rebalancing the vehicles, there can be more than one staff in one vehicle.

Based on these assumptions, the inputs we need for the first strategy include the number of vehicles, staff and stations as well as the customers' trips during the time period under consideration. With these inputs, the strategy can generate the optimal initial distribution for both vehicles and staff, the rebalancing schedule for vehicles and the relocation schedule for the staff.

The model based on the first strategy that we implement is proposed in Gambella et al. (2018). In this paper, the authors construct a time-space network to model both the vehicle rebalancing and staff relocation. The notations for the model are shown in Table 1 - 3.

Indices and Sets	**Description**
h	indices of vehicle h
H	set of vehicles
q	indices of staff q
Q	set of staff
i, j	indices of station number
S	set of stations
t	indices of timestamp t
A_r	set of staff rebalancing arc
A_t	set of staff transferring arc
A_w	set of waiting arc
A_c	set of customer trips arc
i_t^+	arc starting from station i at time t
i_t^-	arc ending at station i at time t
a	arc in format $(i_t, j_{t'})$

Table 1: Indices and Sets for the first model

Parameters	**Description**
p_a	profit or cost of arc a
C_i	capacity of station i
B	maximum number of staff in one vehicle
d_a	customer demand of arc a
$T_{\max}$	maximum of timestamp

Table 2: Parameters for the first model

Decision variables	Description
x_a^h	taking value 1 if vehicle h travel on arc a
y_a^q	taking value 1 if staff q travel on arc a

Table 3: Decision variables for the first model

An example of the time-space network is shown in Figure 8. There are four types of arc in this network:

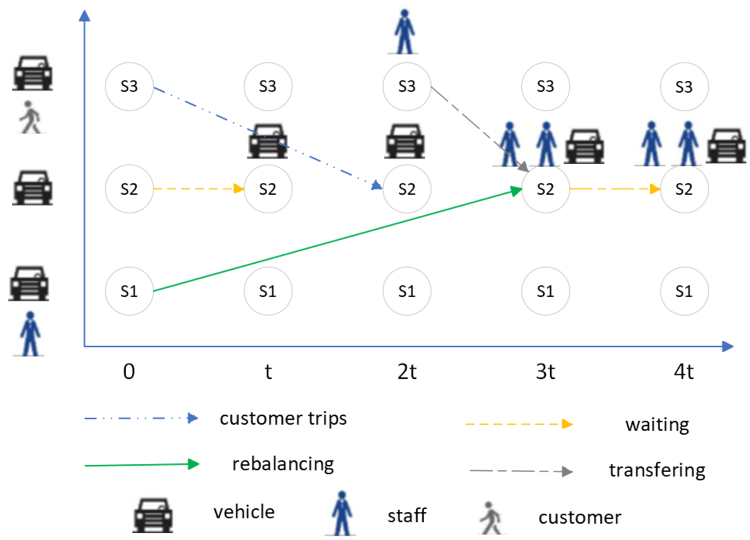

Figure 8: Time-space network for the first strategy

- **Customer trips arc** A_c: each arc $a = (i_t, j_{t'})$ in this set corresponds to a customer request of vehicle starts from station i at time t, ends at station j at time t'. The traveling time between the two stations is related to the distance and obtained from Google Maps API. The profit earned on the traversed arc is p_a>0. This type of arc is shown in blue dash line in Figure 8.

- **Waiting arc** A_w: each arc $a = (i_t, i_{t+1})$ corresponds to either a vehicle or a staff waiting at station i from time t to $t + 1$. Each arc has p_a=0. This type of arc is shown in yellow dash line in Figure 8.

- **Rebalancing arc** A_r: each arc $a = (i_t, j_{t'})$ corresponds to a vehicle being rebalanced from station i at time t to station j at time t' (at least one staff is on board). These arcs are defined for each pair of stations and time instance. As there is no profit when rebalancing and there is extra cost for the consuming of gas (electricity), these arcs have p_a<0. This type of arc is shown in green solid line in Figure 8.

- **Transferring arc** A_t: each arc $a = (i_t, j_{t'})$ corresponds to a staff relocating from station i at time t to station j at time t' without moving any car. Since it adds to the staff's workload, these arcs have p_a<0. This type of arc is shown in gray dash line in Figure 8.

The Mixed-Integer Programming (MIP) model is formulated as follows:

$$
\begin{aligned}
\max \quad & \sum_{h\in H}\sum_{a\in A_c\cup A_r} p_a x_a^h && && \text{(1a)}\\
\text{subject to} \quad & \sum_{h\in H} x_a^h \le d_a, && a\in A_c, && \text{(1b)}\\
& \sum_{h\in H}\sum_{a\in A_w} x_a^h \le C_i, && i\in S, 1\leqslant t< T_{\max}, && \text{(1c)}\\
& \sum_{h\in H}\sum_{a\in A_c\cup A_w\cup A_r} x_a^h \le C_i, && i\in S, t=0, && \text{(1d)}\\
& \sum_{i\in S}\sum_{a\in A_c\cup A_w\cup A_r} x_a^h = 1, && h\in H, && \text{(1e)}\\
& \sum_{a\in (i_t^+)\cup A_c\cup A_w\cup A_r} x_a^h = \sum_{a\in (i_t^-)\cup A_c\cup A_w\cup A_r} x_a^h, && h\in H, i\in S, 1\leqslant t, && \text{(1f)}\\
& \sum_{i\in S}\sum_{a\in A_r\cup A_w\cup A_t} y_a^q = 1, && q\in Q, && \text{(1g)}\\
& \sum_{a\in (i_t^+)\cup A_r\cup A_w\cup A_t} y_a^q = \sum_{a\in (i_t^-)\cup A_r\cup A_w\cup A_t} y_a^q, && q\in Q, i\in S, 1\leqslant t, && \text{(1h)}\\
& \sum_{h\in H} x_a^h \le \sum_{q\in Q} y_a^q, && a\in A_r, && \text{(1i)}\\
& \sum_{q\in Q} y_a^q \le B \sum_{h\in H} x_a^h, && a\in A_r, && \text{(1j)}\\
& x_a^h \in \{0,1\}, && h\in H, a\in A_c\cup A_w\cup A_r, && \text{(1k)}\\
& y_a^q \in \{0,1\}, && q\in Q, a\in A_r\cup A_w\cup A_t. && \text{(1l)}
\end{aligned}
$$

The objective function (1a) maximizes the total profit of the system which is equal to the total profit associated with customer requests minus rebalancing cost. There are four types of constraints: trips, capacity, vehicle flow conservation and staff flow conservation constraints. Constraint set (1b) is the trip constraints, it guarantees that the provided service does not exceed customers' demand. Constraint sets (1c) and (1d) are capacity constraints, (1c) ensures the number of vehicles parked at each station does not exceed capacity at time t>0, (1d) imposes the same condition for t=0. Constraint sets (1e) and (1f) are vehicle flow conservation constraints, (1e) imposes that each vehicle only departs from one station at time 0, (1f) impose vehicle flow conservation at every time-space network node. Constraint sets (1g) and (1h) are staff flow constraints. They impose the same conditions for the staff. Constraint sets (1i) and (1j) are used to match vehicles and staff on relocation arcs. Finally, constraint sets (1k) and (1l) are domain constraints.

The optimal initial distribution of both the vehicles and staff as well as the rebalancing and relocation schedules are parts of the output of the model, which means that this model can manage every decision that may have an impact on the final daily performance.

4.2 Second Strategy

Different from the first strategy which tries to serve as many reservations as possible to maximize profit, the second strategy assumes that all the reservations need to be satisfied. The goal of its associated model is to minimize the cost to achieve this. The key assumptions for this strategy include:

- The system is reservation-based. This is the same as the first strategy.
- Each station has unlimited capacity.
- All reservations can be served under the considered system configuration. Otherwise, there will be no feasible solution.

A remarkable difference with the first strategy is that the second strategy only requires upper bounds for the number of vehicles and staff, rather than their exact value. As a result, the output of this strategy includes the optimal number of vehicles and staff.

All other inputs and outputs are the same as those discussed in the first strategy.

The model built based on the second strategy is proposed in Zhao et al. (2018). The notation is shown in Table 4 - 6. Here we only present the notation that is different from the first model.

Indices and Sets	**Description**
V	set of vertexes in time-space network
T	set of timestamps
t, s	indices of timestamp
A	set of time-space arcs
A_r	$\{A \backslash \{(i_O, t_O, i, t_O), (i, t_D, i_D, t_D), (i, t, i, t+1)\} \| i \in S\}$
(i_O, t_O)	dummy origin
(i_D, t_D)	dummy destination

Table 4: Indices and Sets for the second model

Parameters	**Description**
c_{ij}^d	staff relocation cost between (i, j)
C^a	amortized cost of a vehicle
c_{ij}^l	vehicle traveling cost on between (i, j)
C^y	hiring cost of a staff
$H_{\max}$	maximum number of vehicles
$Q_{\max}$	maximum number of staff
N_{itjs}	number of customer trips on arc (i, t, j, s)

Table 5: Parameters for the second model

Decision variables	**Description**
H_i	initial number of vehicles at station i
Q_i	initial number of staff at station i
x_{itjs}^h	taking value 1 if vehicle h travel on arc (i, t, j, s)
y_{itjs}^q	taking value 1 if staff q travel on arc (i, t, j, s)

Table 6: Decision variables for the second model

The network is very similar to that of the first strategy, with the only difference that there is one dummy origin and one dummy destination (see in Figure 9). The excess vehicles and staff that are not required to satisfy all trips directly go from dummy origin O to dummy destination D and do not contribute to the main network.

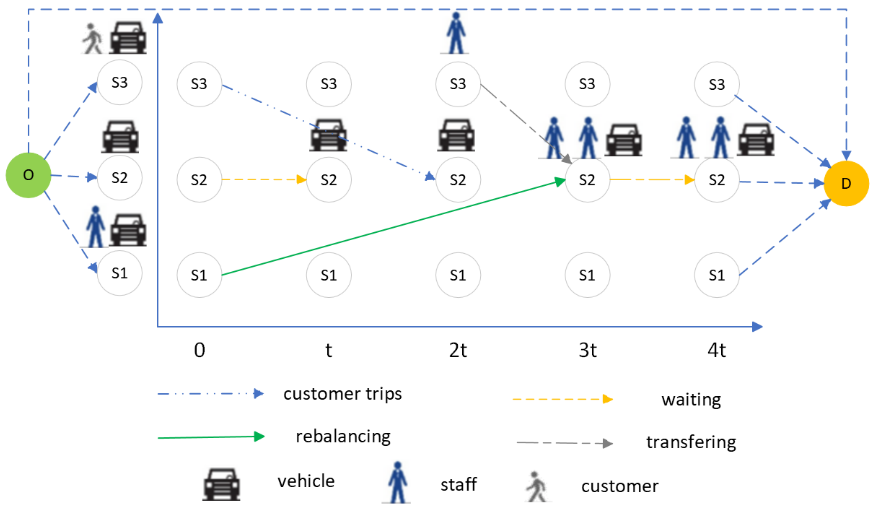

Figure 9: Time-space network for the second strategy

The resulting MIP model is formulated as follows:

$$\text{min} \quad c^a \sum_{i\in S} H_i + c^y \sum_{i\in S} Q_i + \sum_{h\in H}\sum_{A} c^l_{ij} x^h_{itjs} + \sum_{A} c^d_{ij} \Big(\sum_{q\in Q} y^q_{itjs} - \sum_{h\in H} x^q_{itjs} + N_{itjs}\Big) \tag{2a}$$

$$\text{subject to} \quad \sum_{i\in S} H_i \le H_{\max}, \tag{2b}$$

$$\sum_{i\in S} Q_i \le Q_{\max}, \tag{2c}$$

$$\sum_{(j,s)\in V:(i,t,j,s)\in A} x^h_{itjs} - \sum_{(j,s)\in V:(j,s,i,t)\in A} x^h_{jsit} = \begin{cases} 1, & (i,t) = (i_O, t_O) \\ -1, & (i,t) = (i_D, t_D), \\ 0, & \text{otherwise} \end{cases} \quad h\in H, (i,t)\in V, \tag{2d}$$

$$\sum_{h\in H} x^h_{i_O t_O i t_O} = H_i, \quad i\in S, \tag{2e}$$

$$\sum_{h\in H} x^h_{itjs} \ge N_{itjs}, \quad (i,t,j,s)\in A, \tag{2f}$$

$$\sum_{q\in Q} y^q_{itjs} \ge \sum_{h\in H} x^h_{itjs} - N_{itjs}, \quad (i,t,j,s)\in A_r, \tag{2g}$$

$$\sum_{(j,s)\in V:(i,t,j,s)\in A} y^h_{itjs} - \sum_{(j,s)\in V:(j,s,i,t)\in A} y^q_{jsit} = \begin{cases} 1, & (i,t) = (i_O, t_O) \\ -1, & (i,t) = (i_D, t_D), \\ 0, & \text{otherwise} \end{cases} \quad q\in Q, (i,t)\in V, \tag{2h}$$

$$\sum_{q\in Q} y^h_{i_O t_O i t_O} = Q_i, \quad i\in S, \tag{2i}$$

$$x^h_{itjs}, y^q_{itjs} \in \{0,1\}, \quad h\in H, q\in Q, (i,t,j,s)\in A, \tag{2j}$$

$$H_i, Q_i \in N, \quad i\in S. \tag{2k}$$

The objective function (2a) minimizes the total cost, in which $c^a \sum_{i\in S} H_i$ is the fixed cost of buying vehicles, $c^a \sum_{i\in S} H_i$ is the fixed cost of hiring staff, $\sum_{h\in H}\sum_A c^l_{ij} x^h_{itjs}$ is the cost of vehicle moving and $\sum_A c^d_{ij}(\sum_{q\in Q} y^q_{itjs} - \sum_{h\in H} x^q_{itjs} + N_{itjs})$ is the cost of staff moving without a vehicle. There are five types of constraints: maximum fleet size, vehicle flow conservation, staff flow conservation, initial distribution and all demand satisfied. Fleet size constrains include (2b) and (2c). The number of vehicles and staff are both upper bounded by the maximum numbers we provide as parameters. Vehicle flow constraint set (2d) and staff flow constraint set (2e) follow the same idea as those from the first model. The difference is that in the second model, we need to add two extra flow conservation constraints from the dummy origin and dummy destination. Initial distribution constraint set includes (2e) and (2i), it helps get the initial distribution of vehicles and staff at each station. Constraint set (2f) makes sure all demand satisfied. For the remaining constraints, constraint set (2g) imposes that the movement of vehicles except those caused by the customers' trips ($\sum_{h\in H} x^h_{itjs} - N_{itjs}$) is taken by the staff ($\sum_{q\in Q} y^q_{itjs}$). Finally, constraint sets (2j) and (2k) are domain constraints.

4.3 Third strategy

In this strategy, we assume that the car sharing service operator would like to guarantee that a specific set of trips will be served. For example, trips may be reserved by the customers who have paid to get higher priority, or the company needs to ensure a certain level of satisfaction rate to gain customers' confidence. This idea can be easily adapted based on the first strategy; this is what we call here the third strategy. Compared with the first strategy, the third strategy has an extra assumption that a prioritized set of trips must be served. So there will be an extra input of the prioritized trip set. All the other assumptions, inputs and outputs of the third strategy are the same as those in the first one.

The model built based on the third strategy includes all the constraints from (1a) - (1l), with one more set of constraints:

$$\sum_{h\in H} x^h_a \geq d^{'}_a \quad a \in P_r$$

in which P_r is the prioritized trip set, and $d^{'}_a$ is the number of trips needed to be satisfied on arc a.

5 Applications and analysis

In this section, we describe the numerical experiments and analyze the obtained results. For the first strategy, we test different numbers of vehicles and staff and compare the profit, demand satisfaction rate as well as the average trips per vehicle. Before starting the numerical experiments of the first strategy, we need to first get the range of vehicle number we need to test on. Therefore, we first solve one case of the second strategy by setting a high vehicle number upper bound and setting staff number upper bound as 0. In this way, we get the number of vehicles that we need to serve all the demands without rebalancing. For the second strategy, we start with a high upper bound on the number of vehicles and a low upper bound on the number of staff, then decrease the former while increasing the latter. This allows us to characterize the marginal rate of substitution between vehicles and staff that maintains the ability to serve all trips in the system. For the third strategy, we do similar experiments as for the first strategy.

The trip data we use for all the instances is from Wednesday, September 13th, 2017. There are 2181 reservations in total during the whole day. To make sure all the cases can be solved within 72 hours with an optimality gap smaller than 0.5%, we randomly select 25% of trips (418) from that day. The 24 hours are split in 96 intervals of 15 minutes.

Each instance is solved by CPLEX 22.10 on a server with Xeon(R) Gold 6140 CPU clocked at 2.30GHz and 36 processors with a time limit of 72 hours.

5.1 First strategy

We construct instances for the first strategy based on the following system characteristics. The profit and cost are unitless:

- 10 stations, each with a capacity of 10 vehicles

- Profit of customer trips arcs: 0.2/km and 0.1/min
- Cost of rebalancing arcs: 0.15/km
- Cost of transferring arcs: 0.1 (fixed)

The reason why we set the cost of transferring arc to be a fixed number is that in Gambella et al. (2018), the authors propose to set this number to 0, but after testing this parameter setting on small instances, we find that making transferring cost 0 makes the model generates unnecessary transferring actions. For example, we observe staff transferring back and forth between two stations without moving vehicles. We associate a small cost to the transferring arcs to prevent theses aberrant solutions.

Figure 10 shows the result obtained with the first strategy. In total, 28 instances are tested, with the number of vehicles ranging from 10 to 70 and the number of staff ranging from 0 to 3. For instances with 60 to 70 vehicles and 3 staff, the optimal solution is not reached within three days but the optimality gap has already dropped below 0.02% after one day. The result we use for these two instances is from the best bound we can get after solving them for 72 hours.

Satisfaction rate is the number of served trips over all trips, and number of trips per vehicle is the number of served trips divided by the number of vehicles. The Figure 10(a)(b) show that the marginal profit and satisfaction rate brought by one staff or one vehicle decreases as the total numbers of staff and vehicles increase. Figure 10(a) shows that while profit increases with number of vehicles at the beginning, it starts to decrease after the number of vehicles arrives to 60. The explanation for this is that because there is limited capacity in each station, when the number of vehicles is too high, sometimes the staff need to rebalance vehicles to other stations to free up parking spots for incoming vehicles. A simple experiment allows us to verify this. We set the capacity of each station to the total number of vehicles in the system, and we find that profit does not decrease with the increasing in fleet size. The solution from the model also verifies this. With 60 vehicles and no staff, the number of vehicles at each stations during the 96 time intervals is shown in Figure 11. The maximum of y-axis is the station capacity. Each of the stations reaches its capacity at some point during the 96 time intervals. This means that the capacity in the station is not sufficient, so sometimes staff need to rebalance vehicles between stations just to spare some space. As a result, the decreasing of the profit is brought by this type of extra rebalancing cost. Since we still have enough vehicles and staff to serve the trips under this case, we can see that in Figure 10(b), when there is a high number of vehicles, the satisfaction rate does not drop if there is at least one staff in the system. It does drop when there is no staff because under this scenario, it is not possible to move a vehicle to spare one space for another vehicle.

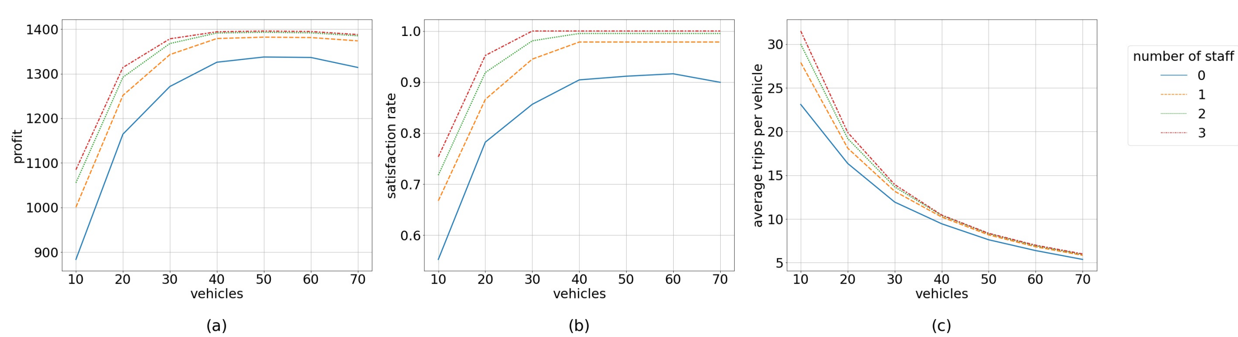

Figure 10: Result of the first strategy

Table 7 reports the computation time as a function of the number of vehicles and staff. The problem is solved easily when there is no staff, even with a high number of vehicles. However, when we increase the number of staff, the computation becomes heavy. We can conclude that the main complexity is brought by the staff and the rebalancing they perform.

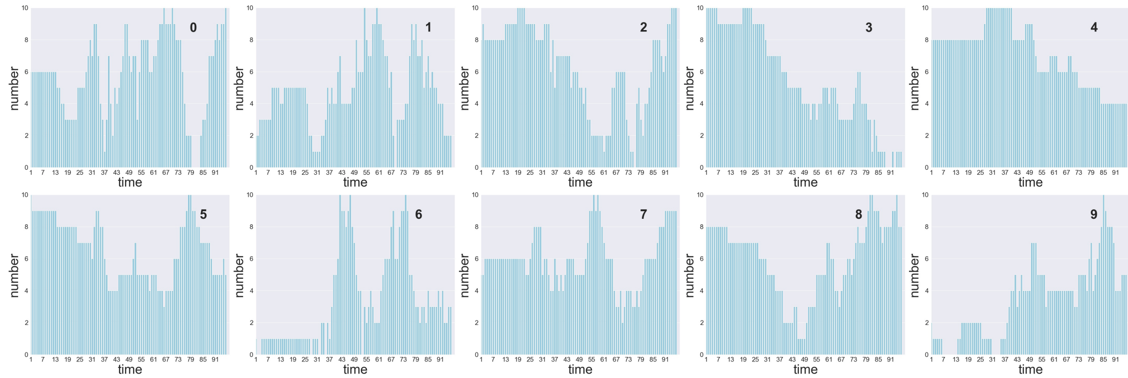

Figure 11: Number of vehicles at stations (60 vehicles and no staff)

Time/(hour)	0	1	2	3
10	<0.1	<0.1	<0.1	<0.1
20	<0.1	<0.1	<0.2	<3.5
30	<0.1	<0.3	<0.3	<1.0
40	<0.1	<0.3	<0.7	<1.1
50	<0.1	<0.8	<2.1	<13.5
60	<0.1	<0.8	<3.6	<24*
70	<0.1	<2.1	<6.8	<24*

Table 7: Computational time. $*$ denotes case for which the optimal solution is not reached within 72 hours, but after 24 hours a solution with a gap smaller than 0.02% is already reached.

5.2 Second strategy

For the second strategy, we construct instances based on the following system characteristics:

- 10 stations with unlimited capacity
- Vehicle traveling cost: 0.15/km
- Staff relocation cost: 0.1/km
- Amortized cost of vehicle: 130/day
- Amortized cost of staff: 150/day

We already know that 68 vehicles are enough to serve all the trips. So in this part, we keep the upper bound of vehicles to 68, and decrease the upper bound of number of staff from 5 to 0. The result is shown in Figure 12. The figure shows the number of vehicles and staff needed to serve all the trips. Intuitively, when there are more staff in the system, fewer vehicles are needed to serve all trips. As the number of staff increases, we observe a decrease in the number of vehicles each additional staff is able to replace. The optimal combination of vehicles and staff therefore depends on the costs of hiring staff and buying vehicles.

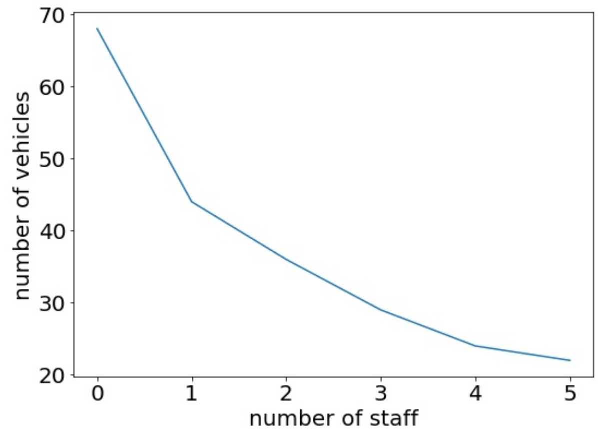

Figure 12: Result of the second strategy

5.3 Third strategy

The parameters we use for the third strategy are the same as the first one. We already see that with 10 vehicles and no staff, we can serve slightly more than 50% of the trip, so for the third strategy, we randomly sample 50% of the trips as the prioritized trip set and conduct 28 instances with the number of vehicles ranging from 10 to 70 and the number of staff ranging from 0 to 3. The result is shown in Figure 13. The result for the scenario without priority (orange lines) is taken from the result of the first strategy, and the blue dash line in the graph is the fixed profit we can get from the 50% of the prioritized trips. When there are only 10 vehicles or 20 vehicles without staff, it is impossible to serve all the prioritized trips. The minimum fleet size to satisfy them is 20 vehicles with at least one staff. When there are enough numbers of vehicles and staff, there is almost no difference between the profit of the two scenarios. This result is consistent with Figure 12, which shows that a single staff and approximately 40 vehicles can already serve all the trips, so with more than 40 vehicles and 1 staff, the constraint of serving the prioritized trips set is not restrictive.

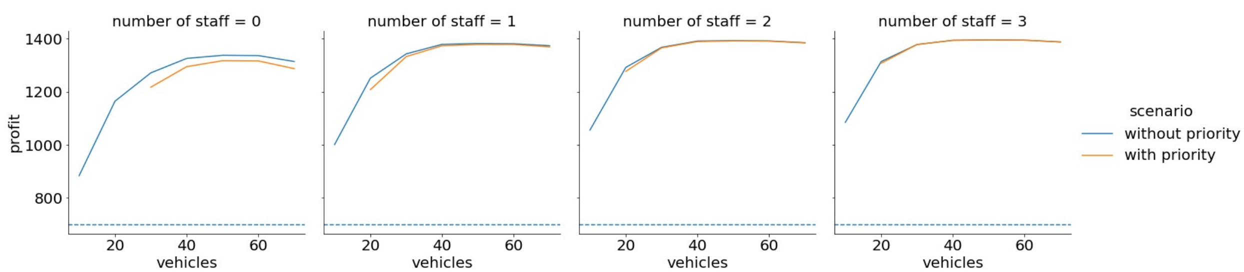

Figure 13: Result of the third strategy

6 Conclusion

In this work, we implement two strategies from the existing literature to solve the rebalancing problem in one-way station-based car sharing systems. The first strategy intends to maximize system profit with fixed numbers of vehicles and staff and limited station capacity, while the second strategy seeks to determine the optimal numbers of vehicles and staff to serve all demand while minimizing costs, with unlimited capacity at all stations. A third strategy is then derived from the first strategy by making a number of trips as prioritized and forcing the model to serve them. This strategy is also a possible application for a real world car sharing system in which customers are given the option to pay for higher priority.

A series of numerical experiments are conducted based on the three models. From the result of the first strategy, we find that the marginal profit brought by one vehicle and staff is decreasing with the increasing of the fleet size and staff number. Also, because the stations have limited capacity, when the number of vehicles became too large, very large fleets yield reduced profits. The result of the second strategy determines the necessary number of vehicles to serve all trips with different numbers of staff in the system. Finally, the result of the third strategy shows that prioritized trips do not affect the overall profit as long as the fleet is large.

The investigated models in this report are built based on some assumptions, which can be relaxed and extended in future research: (1) The models do not take the distance limitation for a vehicle into consideration. Nevertheless, the limited fuel or battery capacity of a shared car is an important component that should be considered in establishing rebalancing schemes. (2) We assume that when a reservation cannot be served due to station capacity, it is directly canceled. However, it's possible that the customer would like to change the origin or destination to another nearby station. This not only ensures a higher probability for a customer to be served, but also increase the system's profitability. To adopt this idea, a set of alternative trips for each reserved trip can be added into the model. When the reserved trip cannot be served, serving any of the trip in the alternative trip set still brings profit. (3) We assume that the customers need to make reservation at least one day before. But in real world, under certain cases, the customers may change their reservations. This is not taken into account in the models we implemented in this work. Solving the model on a rolling basis can help it react to this dynamic of the system. However, this requires the model to be solved within a short period of time, and we already see that with 10 stations, the models we implement in this work take about one day to solve. Therefore, it's also a possible research direction to develop more efficient solving algorithms.

References

Amine Ait-Ouahmed, Didier Josselin, and Fen Zhou. Relocation optimization of electric cars in one-way carsharing systems: modeling, exact solving and heuristics algorithms. *International Journal of Geographical Information Science*, 32(2):367–398, 2018. doi: 10.1080/13658816.2017.1372762. URL https://doi.org/10.1080/13658816.2017.1372762.

Selin Ataç, Nikola Obrenović, and Michel Bierlaire. Vehicle sharing systems: A review and a holistic management framework. *EURO Journal on Transportation and Logistics*, 10:100033, 2021. ISSN 2192-4376. doi: https://doi.org/10.1016/j.ejtl.2021.100033. URL https://www.sciencedirect.com/science/article/pii/S2192437621000054.

Burak Boyacı, Konstantinos G. Zografos, and Nikolas Geroliminis. An integrated optimization-simulation framework for vehicle and personnel relocations of electric carsharing systems with reservations. *Transportation Research Part B: Methodological*, 95:214–237, 2017. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2016.10.007. URL https://www.sciencedirect.com/science/article/pii/S0191261515301119.

Georg Brandstätter, Claudio Gambella, Markus Leitner, Enrico Malaguti, Filippo Masini, Jakob Puchinger, Mario Ruthmair, and Daniele Vigo. *Overview of Optimization Problems in Electric Car-Sharing System Design and Management*, pages 441–471. Springer International Publishing, Cham, 2016. ISBN 978-3-319-39120-5. doi: 10.1007/978-3-319-39120-5_24. URL https://doi.org/10.1007/978-3-319-39120-5_24.

Maurizio Bruglieri, Ferdinando Pezzella, and Ornella Pisacane. An adaptive large neighborhood search for relocating vehicles in electric carsharing services. *Discrete Applied Mathematics*, 253:185–200, 2019. ISSN 0166-218X. doi: https://doi.org/10.1016/j.dam.2018.03.067. URL https://www.sciencedirect.com/science/article/pii/S0166218X18301665. 14th Cologne-Twente Workshop on Graphs and Combinatorial Optimization (CTW 2016).

Leonardo Caggiani, Luigi Pio Prencipe, and Michele Ottomanelli. A static relocation strategy for electric car-sharing systems in a vehicle-to-grid framework. *Transportation Letters*, 13(3):219–228, 2021. doi: 10.1080/19427867.2020.1861501. URL https://doi.org/10.1080/19427867.2020.1861501.

Daniel Chemla, Frédéric Meunier, and Roberto Wolfler Calvo. Bike sharing systems: Solving the static rebalancing problem. *Discrete Optimization*, 10(2):120–146, 2013. ISSN 1572-5286. doi: https://doi.org/10.1016/j.disopt.2012.11.005. URL https://www.sciencedirect.com/science/article/pii/S1572528612000771.

Claudio Gambella, Enrico Malaguti, Filippo Masini, and Daniele Vigo. Optimizing relocation operations in electric car-sharing. *Omega*, 81:234–245, 2018. ISSN 0305-0483. doi: https://doi.org/10.1016/j.omega.2017.11.007. URL https://www.sciencedirect.com/science/article/pii/S0305048317302803.

Stefan Illgen and Michael Höck. Literature review of the vehicle relocation problem in one-way car sharing networks. *Transportation Research Part B: Methodological*, 120:193–204, 2019. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2018.12.006. URL https://www.sciencedirect.com/science/article/pii/S0191261518300547.

Vassio Luca, Giordano Danilo, Mellia Marco, and Cocca Michele. Data for: "free floating electric car sharing design: Data driven optimisation". anonymized datasaset of 2 months of trips of car sharing users in the city of turin, 2019.

Mehdi Nourinejad and Matthew J. Roorda. A dynamic carsharing decision support system. *Transportation Research Part E: Logistics and Transportation Review*, 66:36–50, 2014. ISSN 1366-5545. doi: https://doi.org/10.1016/j.tre.2014.03.003. URL https://www.sciencedirect.com/science/article/pii/S1366554514000362.

Mehdi Nourinejad, Sirui Zhu, Sina Bahrami, and Matthew J. Roorda. Vehicle relocation and staff rebalancing in one-way carsharing systems. *Transportation Research Part E: Logistics and Transportation Review*, 81:98–113, 2015. ISSN 1366-5545. doi: https://doi.org/10.1016/j.tre.2015.06.012. URL https://www.sciencedirect.com/science/article/pii/S1366554515001349.

Tal Raviv, Michal Tzur, and Iris A. Forma. Static repositioning in a bike-sharing system: models and solution approaches. *EURO Journal on Transportation and Logistics*, 2(3):187–229, 2013. ISSN 2192-4376. doi: https://doi.org/10.1007/s13676-012-0017-6. URL https://www.sciencedirect.com/science/article/pii/S2192437620301175.

Meng Zhao, Xiaopeng Li, Jiateng Yin, Jianxun Cui, Lixing Yang, and Shi An. An integrated framework for electric vehicle rebalancing and staff relocation in one-way carsharing systems: Model formulation and lagrangian relaxation-based solution approach. *Transportation Research Part B: Methodological*, 117: 542–572, 2018. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2018.09.014. URL https://www.sciencedirect.com/science/article/pii/S0191261517302965.