**Optimizing Bike Redistribution for Urban Mobility in Vancouver**

**Edie Cheng, Mengxu Pan, Jun Tan**

**1. Introduction**

Our team has identified a real-world challenge that excites us, drawing from techniques we've acquired in our class: addressing the bike-sharing issue in Vancouver, and ensuring each station maintains an optimal bike inventory to serve the city's demands. Our objective is to devise a regulatory system utilizing integer programming algorithms to facilitate efficient bike redistribution.

Central to our approach is the conceptualization of bike redistribution as a variant of the renowned Traveling Salesman Problem (TSP). In this adaptation, each station becomes a node, and the task at hand transforms into finding the most efficient route for a van to traverse, ensuring adequate bike inventory at every stop. However, the intricacies of our challenge necessitate a departure from the conventional TSP framework. The addition of constraints, such as the van's capacity limitations, endows our problem with the characteristics of a Constraints Optimization Problem. This hybrid nature demands innovative solutions that balance the twin objectives of minimizing travel distance while adhering to practical constraints.

Using the real-life dataset provided by Moby by Rogers Go (Mobi Bikes), one of the city's premier bike-sharing services, we analyzed the bike flow of all stations throughout February 2024. From this comprehensive analysis, we handpicked a subset of 20 strategically positioned stations to serve as the focal points of our modeling efforts. Our objective crystallizes into the development of a regulatory framework empowered by integer programming algorithms. By harnessing the potential of these algorithms, we aim to engineer a system that not only optimizes the distribution of bikes but also navigates the complexities of real-world constraints with finesse.

**1.1 Personal relevance**

Edie: I have lived in Shanghai, a densely populated city, and I experienced firsthand the challenges posed by the abundance of shared bicycles, especially near my home which was adjacent to a subway station. Each morning, swarms of bicycles would overflow the space, at times impeding traffic. This sparked my curiosity about how bicycles were dispatched across various stations. Now living in the beautiful city of Vancouver, where shared bicycles are also prevalent but appear more orderly at their stations, I am intrigued by the possibility of a systematic approach to managing these stations. This inspired me to undertake a computer science project to explore whether I could develop a system to optimally plan the distribution of shared bicycles at each docking station in Vancouver, drawing parallels to my observations in Shanghai and applying these insights in a new urban context.

Mengxu: Participating in the United Way Hackathon last fall left a lasting impression on me. Among the challenges highlighted by the United Way team was the shortage of resources for collecting and distributing donated tampons and pads to various agencies across Vancouver, particularly those situated far from the central office. Drawing parallels between this logistical issue and the bike distribution problem, I find both endeavors meaningful to address. Making a positive impact by applying theoretical knowledge to solve real-world problems is rewarding and fulfilling.

Jun: I used to be a frequent user of bike-sharing services when I lived in Beijing. There was a 99 Yuan deposit (equivalent to 19 CAD) that I doubt I'll ever get back due to the bankruptcy of a once-prominent bike-sharing company in China. Naturally, I'm interested in the sustainability of bike-sharing companies, particularly in minimizing operational costs. Studying the redistribution problem of Mobi by Rogers, the bike-sharing company in Vancouver, can help us estimate the operational costs of the company and gain a better understanding of this business. Besides, this would be a good opportunity to apply what we have learned in class to a problem outside our familiar scope.

**2. Methodology**

In our project, we leveraged integer programming to formulate and solve the repositioning route generation problem. We modeled the repositioning as a Constraints Optimization Problem(COP), and our aim was to find not only a shortest cycle that traveled all the stations needed to be balanced, but our van was also under the constraints that it needed to be able to finish the task at each station.(e.g. the van had enough bikes to unload or enough space to load the bikes from the station)

**2.1 Model Formulation**

The environment of the repositioning problem is described by the following set and constants:

S the set of stations, the depot is set to be station 0;

the distance between station i and station j;

the workload for the van at station i, a positive number indicates loading

bikes, a negative number means unloading bikes;

C capacity of the van.

**2.1.1 Assumptions of the Model**

1. Time is not a constraint in the bike repositioning process.
2. Stations are in a static status during the repositioning.
3. We assume the stations are balanced at the start of our datasets, so our task each day is to load excessive bikes if there is a net inflow at a station, and unload bikes to fulfill shortages if there is a net outflow at a station.
4. We designated a depot address(station 0 in S) as the starting and ending point of the repositioning route, the workload is 0 at this station.

Considering the general sense that bike demand is minimal after midnight, our repositioning actions are scheduled between 0:00 and 6:00 daily, there is enough time for any repositioning route. Given the negligible demand during this period, we assume stations are in a static status, with no bikes rented or returned.

Since our project focuses on determining an optimal repositioning route, the specific workload for each station does not impact the model’s effectiveness as long as it varies in each station. Bike-sharing companies typically develop models to predict customer needs based on factors such as time, weather, and special events, to get optimal bike stock for each station. We developed the model to handle different workloads at each station. In the future, if our datasets are equipped with a demand forecast, our routing model does not need any change.

**2.1.2 Main Model Formulation**

Decision Variables

a binary variable that equals one if the van travels directly from station i

to station j, and zero otherwise;

number of bicycles carried on the van before it arrives at station i;

auxiliary variables used to eliminate sub-tour results.

The following parameters need to be defined and used specifically in the model:

the distance between station i and station j;

the work for the van at station i;

C capacity of the van;

M auxiliary constant to eliminate separate cycles in the route.

Objective:

s.t

(1)

(2)

(3)

(4)

(5)

and j 0 (6)

(7)

+ *- C* <= 0 j = 0 (7)

(8)

+>= 0j = 0 (8)

(9)

(10)

(11)

(12)

(13)

**2.1.3 Explanation of the Objective Function and Constraints**

The objective function minimizes the total distance the van travels, which is our goal. Constraint (1) ensures that each station will be visited at most once on the route. Constraints (2) and (3) describe the main constraints to make the route a cycle. For any station i, the number of edges in and out from i must be equal. From constraint(1) we already have that if any station j is on the route, then there can be only one edge going into j, so j will be visited at most once, and if j is visited there must also be an edge going out from j. Constraint (3) states that the van will always return to station 0, so with Constraint (2) and (3) combined, the route will start from and end at station 0 and be a cycle. Constraint (4) guarantees there is no self-loop in the route. Constraints (5) and (6) take the workload of the stations into the planning of the route, that the van will visit a station in the route only if the workload at that station is not zero(except for the depot station 0). Constraints (7) and (8) limit the number of bikes on the van on the route to make the route realistic for repositioning, if the van travels from station i to station j, the number of bikes on the van before it arrives j should be , the number of bikes after finishing station i and that number should be between zero and the maximum capacity of the van.

In our early test of the model, Integer Programming will generate separate cycles as the optimal route. An example of the optimal route on Feb 20th is shown in Figure 2-1. On that day, the optimal route is two separate cycles without breaking any previous constraints, but it is not a practical route for a real repositioning process.

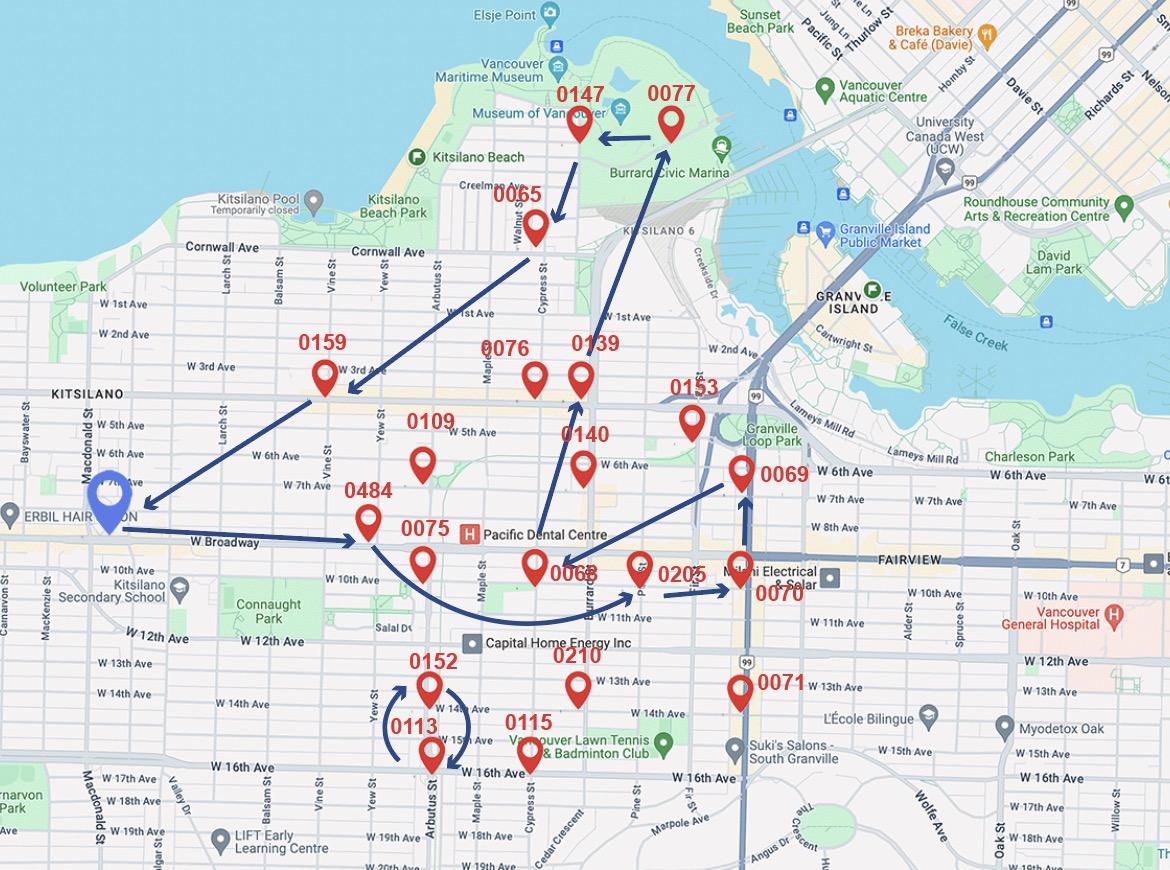


Figure 2-1

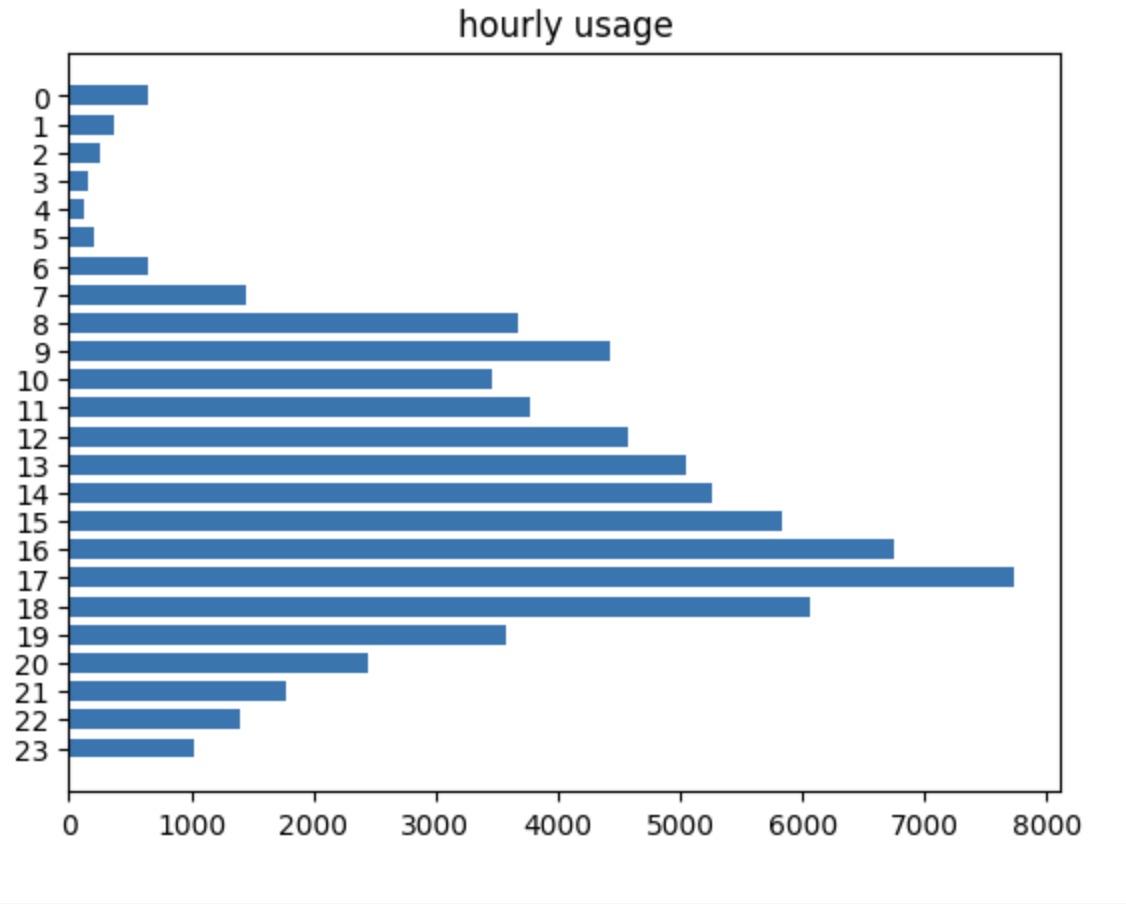
To ensure the optimal solution is one continuous cycle, we adopted the method in Ravis, Tzur, and Forma(2011), and we introduced constraint(9) with strictly increasing variables . In this constraint, if the van travels from station i to station j then , while the depot(station 0) is excluded from the range of j. This would raise a contradiction if there is a cycle that does not contain station 0. We set M to be the total number of stations (M = |S|).

**2.2 Dataset Description**

Mobi by Rogers, a community bike-sharing initiative in Vancouver, British Columbia, provides open access to its system data, which includes detailed records like trip start and end times along with the stations of departure and return. This transparency offers a deep dive into the utilization patterns of bicycles throughout the urban area.

In our project, the dataset covers the time span from Feb 1st to Feb 29th, 2024.[1] The data can be acquired at <https://drive.google.com/file/d/1TJmE1DHYsF2eJ3BC0584hDF_-Vuqt1z-/view?usp=sharing>

Figure 2-2 illustrates the hourly trend of bike usage throughout February.



During this period, a total of 70664 trips were recorded, with 1564 rentals taking place between midnight and 6:00 AM. This data confirms our initial hypothesis that there is minimal demand for bicycles after midnight.

According to our assumption(3), the inventory of bikes at each station was deemed optimal as of February 1st at midnight. From this baseline, each bike rental is seen as creating a deficit by removing a bike from the optimal inventory, whereas each return adds to the surplus, exceeding the optimal level by one bike. The day's end net flow at each station, therefore, indicates whether bikes need to be relocated—loaded onto the van if there is a surplus, or unloaded if there is a deficit.

For example, if a station experiences 5 bikes rented and 6 returned in one day, the total net flow for that station is +1. This means there is one extra bike at the station at the end of the day compared to the beginning, suggesting that this extra bike should be transported away to maintain optimal inventory levels. If the net flow is zero, the station's bike stock is already ideal, and no van visit is necessary.

We utilized the Google Maps API to ascertain the distances between stations, selecting the shortest driving routes to accurately determine the space between each station pairing.

The van assigned for bike repositioning is equipped with a capacity of 25 bicycles. We've implemented an adaptive system where the number of bikes transported each day is dynamically determined based on daily demand fluctuations.

**2.3 Data Selection**

In the comprehensive system composed of all 250 stations, the theoretical assumption is that each bicycle returns to a station at midnight (as depicted in Assumption 2). This creates a balanced system where the total number of bicycles remains constant, and the repositioning van only needs to transfer bicycles between stations without the need to introduce or remove any bicycles from the system.

However, the scenario changes when studying a subset of stations. Bicycles might be rented from these locations without being returned, or returned without being rented out. Among the total of 250 stations, due to computational constraints, we selected 20 stations from communities such as False Creek, Kitsilano, and Fairview in the western area of Vancouver, which overall can balance themselves. These stations represent a significant sample, allowing us to delve deeper into the relocation challenges and strategies applicable to a broader context. Table 2-2.D lists the sample stations we chose.

Table 2-2

| **Departure.station** | **Total\_Trips.x** | **Total\_Trips.y** | **avg\_dep** | **avg\_rtn** | **avg\_net** |
| --- | --- | --- | --- | --- | --- |
| 0115 Cypress & 16th | 42 | 26 | 1 | 1 | 0 |
| 0113 Arbutus & 16th | 114 | 113 | 4 | 4 | 0 |
| 0152 Arbutus Greenway  & 14th | 115 | 106 | 4 | 4 | 0 |
| 0071 14th & Granville | 200 | 189 | 7 | 7 | 0 |
| 0070 10th & Granville | 229 | 210 | 8 | 7 | -1 |
| 0205 Pine & 10th | 174 | 185 | 6 | 6 | 0 |
| 0068 Cypress & 10th | 173 | 155 | 6 | 5 | -1 |
| 0075 Arbutus & 10th | 175 | 167 | 6 | 6 | 0 |
| 0484 Yew & Broadway | 288 | 265 | 10 | 9 | -1 |
| 0109 7th & Arbutus | 166 | 139 | 6 | 5 | -1 |
| 0140 Burrard & 7th | 149 | 145 | 5 | 5 | 0 |
| 0069 7th & Granville | 215 | 171 | 8 | 6 | -2 |
| 0210 Burrard & 14th | 99 | 70 | 3 | 2 | -1 |
| 0159 Vine & 4th | 292 | 241 | 10 | 8 | -2 |
| 0139 Burrard & 4th | 124 | 136 | 4 | 5 | 1 |
| 0076 Cypress & 4th | 373 | 402 | 13 | 14 | 1 |
| 0153 Arbutus Greenway & Fir | 190 | 204 | 7 | 7 | 0 |
| 0065 Cypress & Cornwall | 254 | 308 | 9 | 11 | 2 |
| 0147 Chestnut & McNicoll | 136 | 128 | 5 | 4 | -1 |
| 0077 Vanier Park | 71 | 83 | 2 | 3 | 1 |

**3. Numerical Analysis**

**3.1 Overview**

We developed our bike repositioning route model using Python, specifically leveraging the ortools.linear\_solver module with pywraplp for optimization. The OR-Tools provides robust optimization capabilities, including Mixed Integer Programming (MIP) tools that are well-suited for addressing complex routing problems. Key components of our implementation involve extracting data from the dataset, creating an adjacency matrix for the stations with weighted edges (which represent distances), and formulating the model constraints, including the upper and lower bounds for each. The entire process is driven by the MIP solver from OR-Tools,and the detailed Python code can be found in Appendix A.

The day-to-day results are shown in Table 3-1, where the IP distance is the length of the generated repositioning route. We listed the total net flow for each day, with negative numbers indicating a shortage of bikes in the selected network requiring addition, and positive numbers indicating an excess of bikes for balancing. Additionally, the initial number of bikes the van should carry given by the model is also listed. As a comparison, we used the python-tsp library to generate the TSP route to visit all stations requiring rebalancing on each day.

The results indicate that our model consistently determined the optimal number of bikes for the van to carry initially. Additionally, on 28 out of 29 days in February, the distances of the repositioning routes matched those calculated by the TSP, proving the effectiveness of our model.

Table 3-1

| **DATE** | **Daily Task** | **Initial Load** | **IP distance** | **TSP distance** |
| --- | --- | --- | --- | --- |
| 02-01 | 9 | 4 | 10.8 | 10.8 |
| 02-02 | -8 | 14 | 11.4 | 11.4 |
| 02-03 | 5 | 7 | 11.9 | 11.9 |
| 02-04 | 10 | 0 | 10.7 | 10.7 |
| 02-05 | -6 | 16 | 10.5 | 10.5 |
| 02-06 | -21 | 21 | 11.0 | 11.0 |
| 02-07 | 1 | 4 | 10.5 | 10.5 |
| 02-08 | 4 | 11 | 11.1 | 11.1 |
| 02-09 | -13 | 16 | 11.9 | 11.9 |
| 02-10 | -9 | 10 | 11.9 | 11.9 |
| 02-11 | -3 | 4 | 10.1 | 10.1 |
| 02-12 | 4 | 8 | 10.3 | 10.3 |
| 02-13 | -24 | 24 | 11.2 | 10.8 |
| 02-14 | -18 | 19 | 10.6 | 10.6 |
| 02-15 | -2 | 7 | 11.9 | 11.9 |
| 02-16 | -5 | 10 | 11.9 | 11.9 |
| 02-17 | -9 | 9 | 11.2 | 11.2 |
| 02-18 | -19 | 20 | 10.4 | 10.4 |
| 02-19 | -6 | 14 | 10.4 | 10.4 |
| 02-20 | 2 | 4 | 10.4 | 10.4 |
| 02-21 | 5 | 5 | 11.2 | 11.2 |
| 02-22 | -2 | 8 | 11.1 | 11.1 |
| 02-23 | -10 | 17 | 11.1 | 11.1 |
| 02-24 | -19 | 19 | 11.6 | 11.6 |
| 02-25 | -1 | 6 | 10.9 | 10.9 |
| 02-26 | -13 | 17 | 11.3 | 11.3 |
| 02-27 | -6 | 9 | 12.0 | 12 |
| 02-28 | 11 | 4 | 11.5 | 11.5 |
| 02-29 | 7 | 9 | 11.9 | 11.9 |

**3.2 An Example of the Repositioning Route**

To give an example of the repositioning route, Figure 3-1 shows the repositioning route on February 12nd, 2024. The sequential order of stations in this repositioning route is as follows:

inventory➔0484➔0075➔0152➔0115➔0210➔0071➔0070➔0069➔0153➔0139

➔0147➔0077➔0065➔0076➔0109➔0159➔inventory with a total distance of 10.3 km.

Station 0113, 0140, 0068 and 0205 had a netflow of zero bikes on that day(the bikes rented and the bikes returned were equal), hence they were skipped on the route.

A map with points and numbers

Description automatically generated

Total Distance is 10.3 km

Figure 3-1

Table 3-2 illustrates the tasks performed at each station along the route. On this particular day, the total net flow of bikes at all stations was 4. Given our van's capacity is set at 25 bikes, and the van begins its route with 8 bikes from the inventory, a figure also calculated by our program. Subsequently, at each station on the route, the van loads or unloads bikes based on the station’s net flow of the day.

The final destination is Station 0159, where the van arrives with 9 bikes. At this station, 3 excessive bikes are loaded, bringing the total to 12 bikes before returning to the inventory. This count is 4 bikes more than the original number, aligning with the total net flow addressed in this specific area.

| **Maximum Capacity = 25, and starts from the inventory with 8 bikes** | | |
| --- | --- | --- |
| **Station** |  |  |
| 0484 Yew & Broadway | -4 | 4 |
| 0075 Arbutus & 10th | 2 | 6 |
| 0152 Arbutus Greenway | 1 | 7 |
| 0115 Cypress & 16th | -2 | 5 |
| 0210 Burrard & 14th | -2 | 3 |
| 0071 14th & Granville | 2 | 5 |
| 0070 10th & Granville | -5 | 0 |
| 0069 7th & Granville | 2 | 2 |
| 0153 Arbutus Greenway & Fir | -1 | 1 |
| 0139 Burrard & 4th,0139 | 2 | 3 |
| 0147 Chestnut & McNicoll | -2 | 1 |
| 0077 Vanier Park | -1 | 0 |
| 0065 Cypress & Cornwall | 2 | 2 |
| 0076 Cypress & 4th,0076 | 4 | 6 |
| 0109 7th & Arbutus | 3 | 9 |
| 0159 Vine & 4th | 3 | 12 |

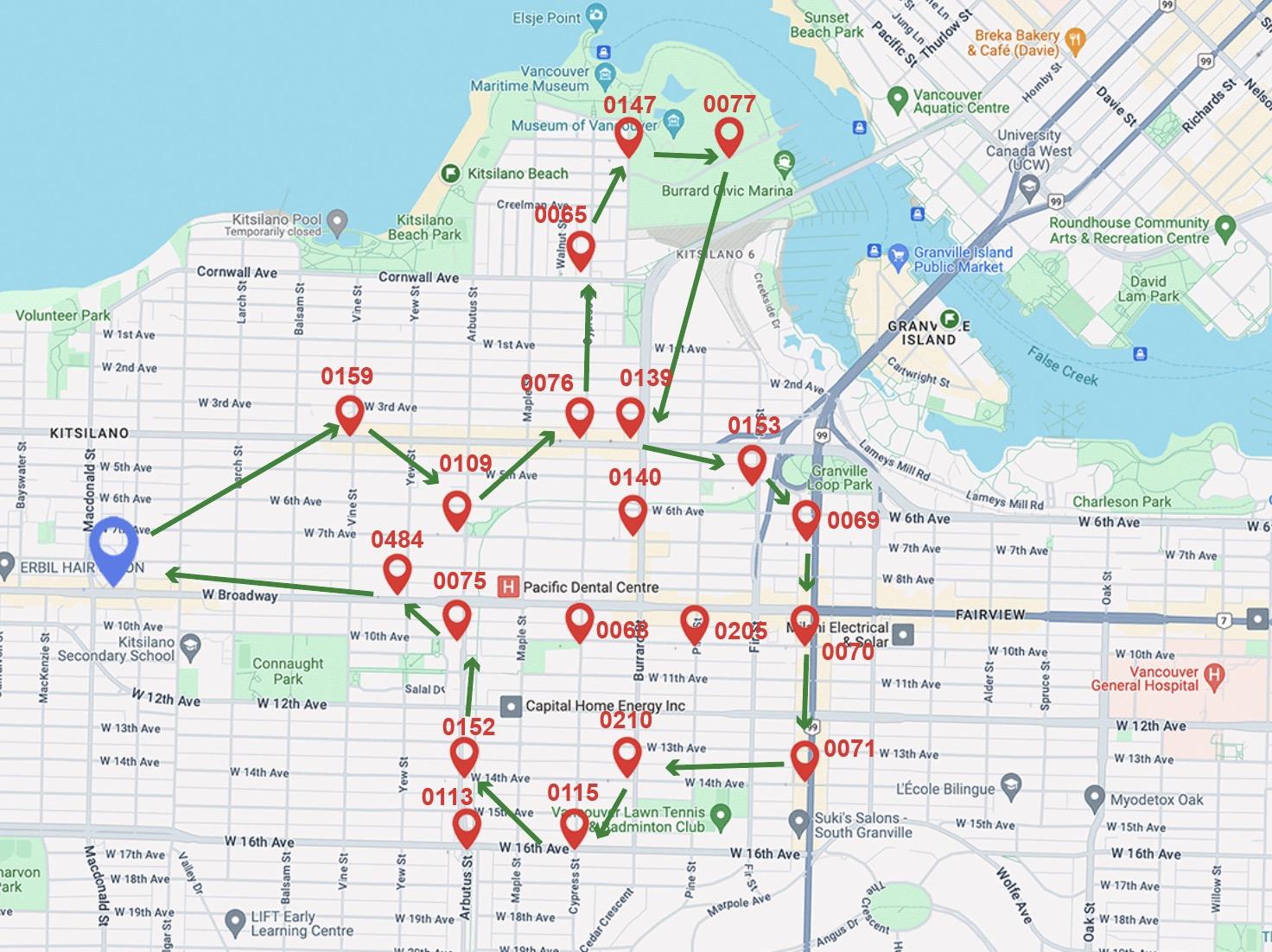
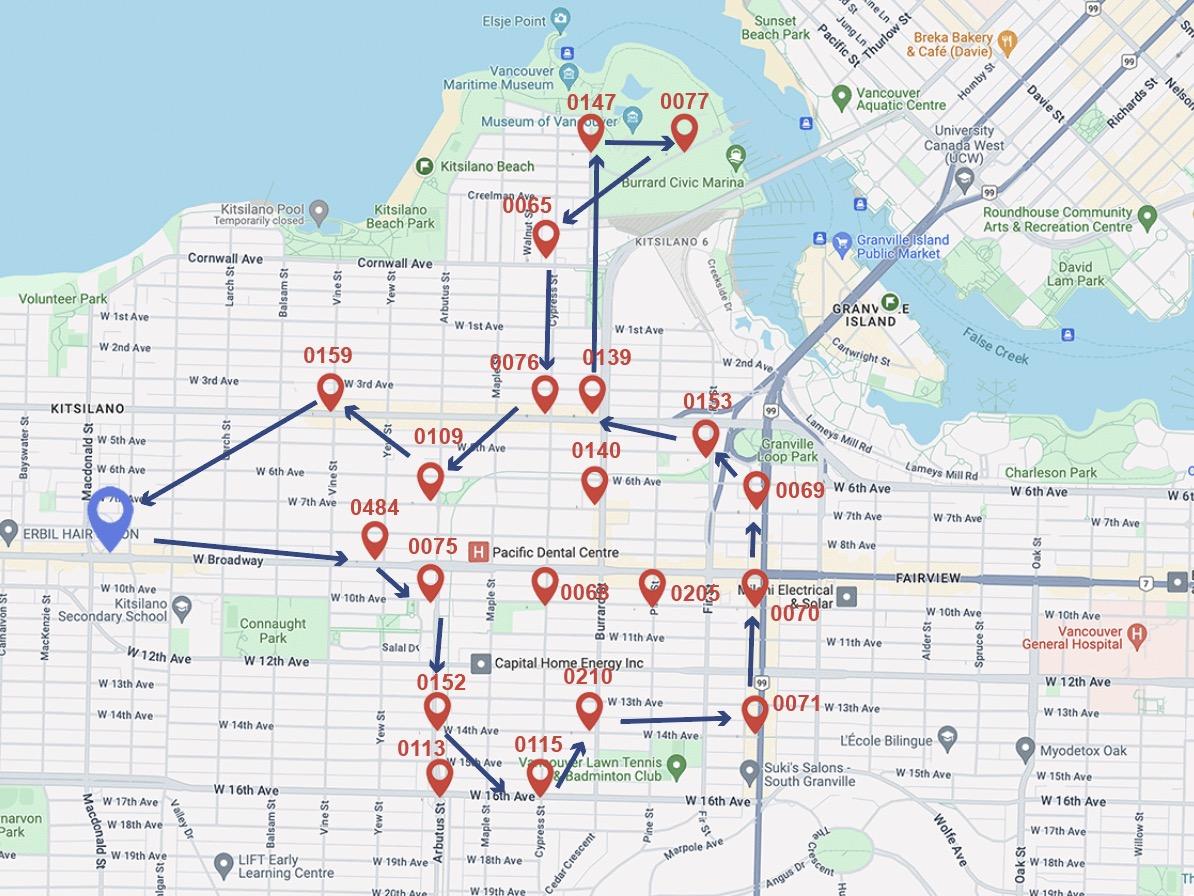
Table 3-2

If we analyze these 16 stations using the traveling salesman approach, we can identify a path

inventory➔0159➔0109➔0076➔0065➔0147➔0077➔0139➔0153➔0069➔0070

➔0071➔0210➔0115➔0152➔0075➔0484➔inventory with a total distance of 10.3 km

It has the exact same length as the one we have derived(Figure 3-2). This confirms that our chosen route is indeed the optimal one.

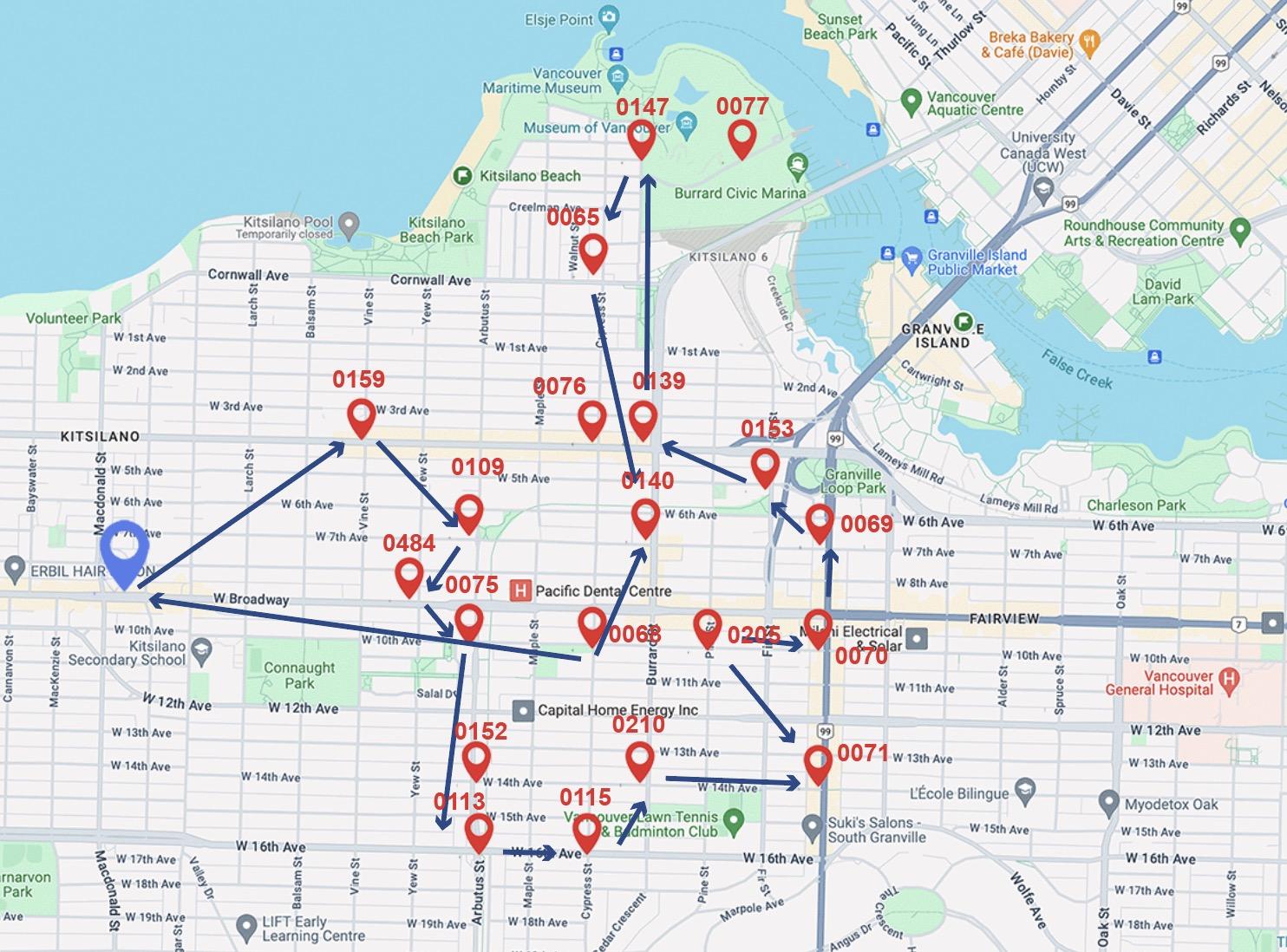


**COP (10.3km) TSP(10.3km)**

Figure 3.2 A comparison of the repositioning route and the TSP route on Feb 12rd

**3.3 Comparative Analysis of Repositioning Routes: Integer Programming under Task Constraints and TSP**

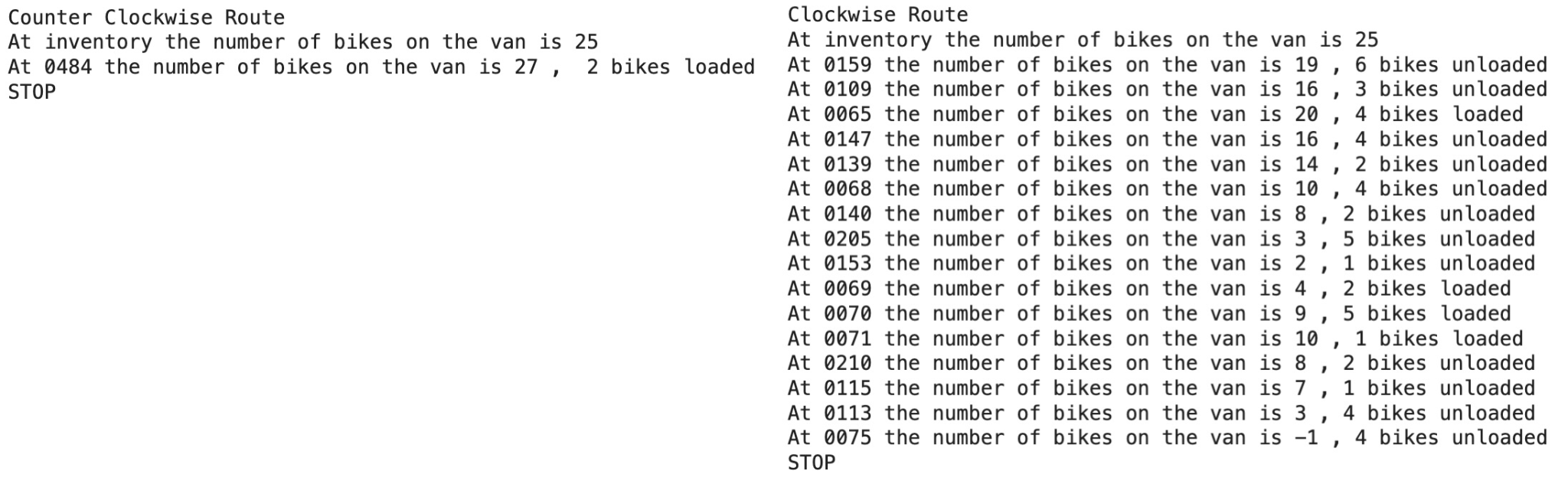
In this section, we compare the difference between the TSP route and our ILP route under the task constraints. The purpose is to illustrate why Integer Programming proves to be the optimal choice for our project, transcending the limitations in traditional TSP algorithms in solving this problem.

****

**ILP (11.2km) TSP(10.8km)**

Figure 3.3 A comparison of the repositioning route and the TSP route on Feb 13th

On February 13th, we observed that our repositioning route differed from the TSP route(as shown in Figure 3.3), spanning a distance of 11.2 km, which was longer than the TSP’s 10.8 km. On this day, the total net outflow was 24, indicating that the initial load on the van needed to be at least 24 to complete the task. Within the capacity limit of 25, we conducted repositioning work with initial load settings of both 24 and 25 along the TSP route, the result is shown in Figure 3-4.



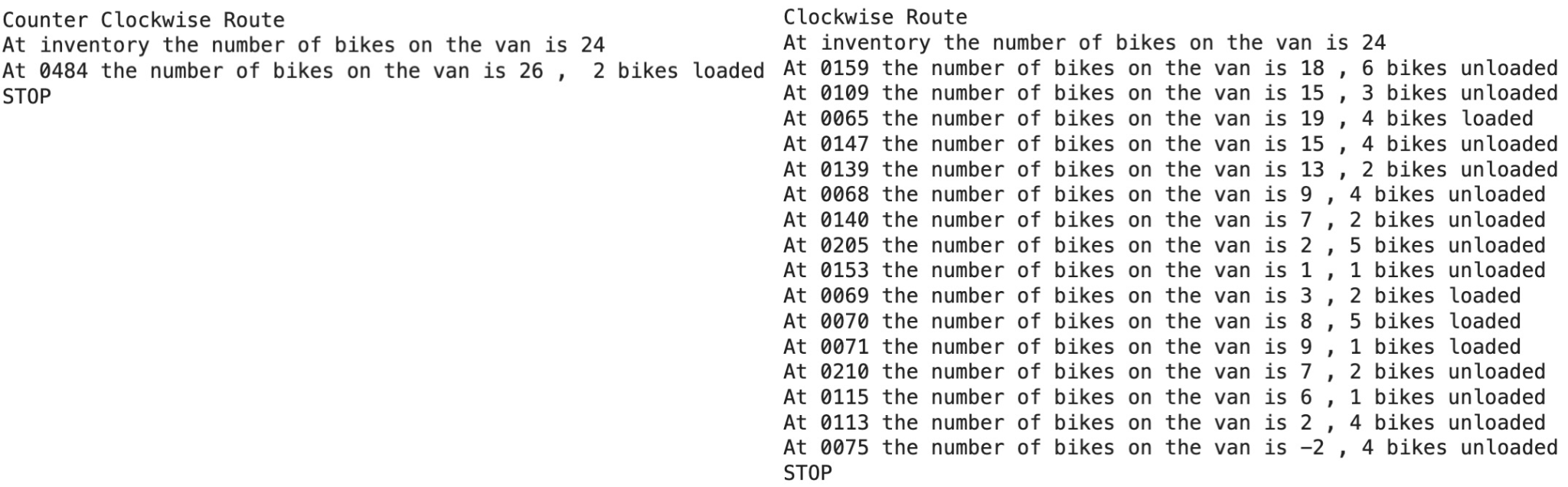


Figure 3.4 Testing repositioning on the TSP route on 13th Feb​

Regardless of whether we traversed the TSP route clockwise or counter clockwise and whether the initial load was 24 or 25, the van had to halt repositioning at a certain station due to space constraints or a shortage of bikes. This highlights the limitation of the TSP algorithm in addressing repositioning problems. While the TSP route can readily identify an optimal solution when it is possible to complete the task on this route, it fails to provide an alternative solution when the shortest route is not feasible for repositioning, leaving us unable to determine the shortest route capable of completing the task.

Next, we tested repositioning using the Integer Programming route, with the results depicted in Figure 3.5. By treating the repositioning task at each station as constraints in our COP model, our solution ensures the completion of work at each station in a realistic manner, resulting in the generation of the shortest possible route.

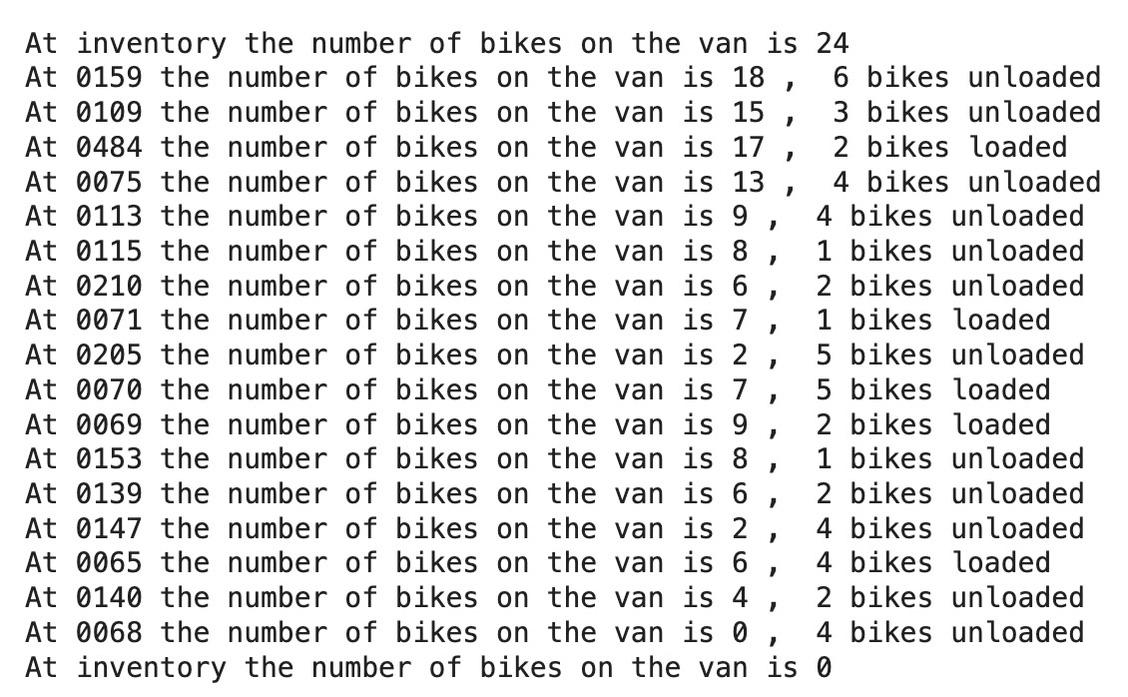


Figure 3.5 Testing repositioning on the model's route on 13th Feb​

Our comparative analysis between our Integer Programming (IP) routes and Traveling Salesman Problem (TSP) routes has demonstrated that our IP model is capable of generating TSP repositioning routes when feasible. Furthermore, in cases where a TSP repositioning route is not viable, our IP model identifies the shortest route that can complete the task.

**3.4 Exploring a Multiple-Vans model**

On certain days, such as February 13th, we encountered spikes in bike demand, with a total net outflow of 24 bikes, nearly reaching the van's capacity. To address this challenge, we explored the implementation of a multiple-vans model capable of efficiently managing time-sensitive and capacity-constrained repositioning tasks.

The key modification in this model involved expanding the decision variables to incorporate van-specific movements, denoted by the addition of a van index (v). For instance, X[i, j, v] now represents the movement of van v from station i to station j.

However, during the implementation of this expanded model, we encountered significant computational challenges. With the program failing to return a solution within a reasonable timeframe, we settled to compute routes for a two-vans fleet to rebalance the first 15 stations in our station set on 13th February, as depicted in Figure 3.5.

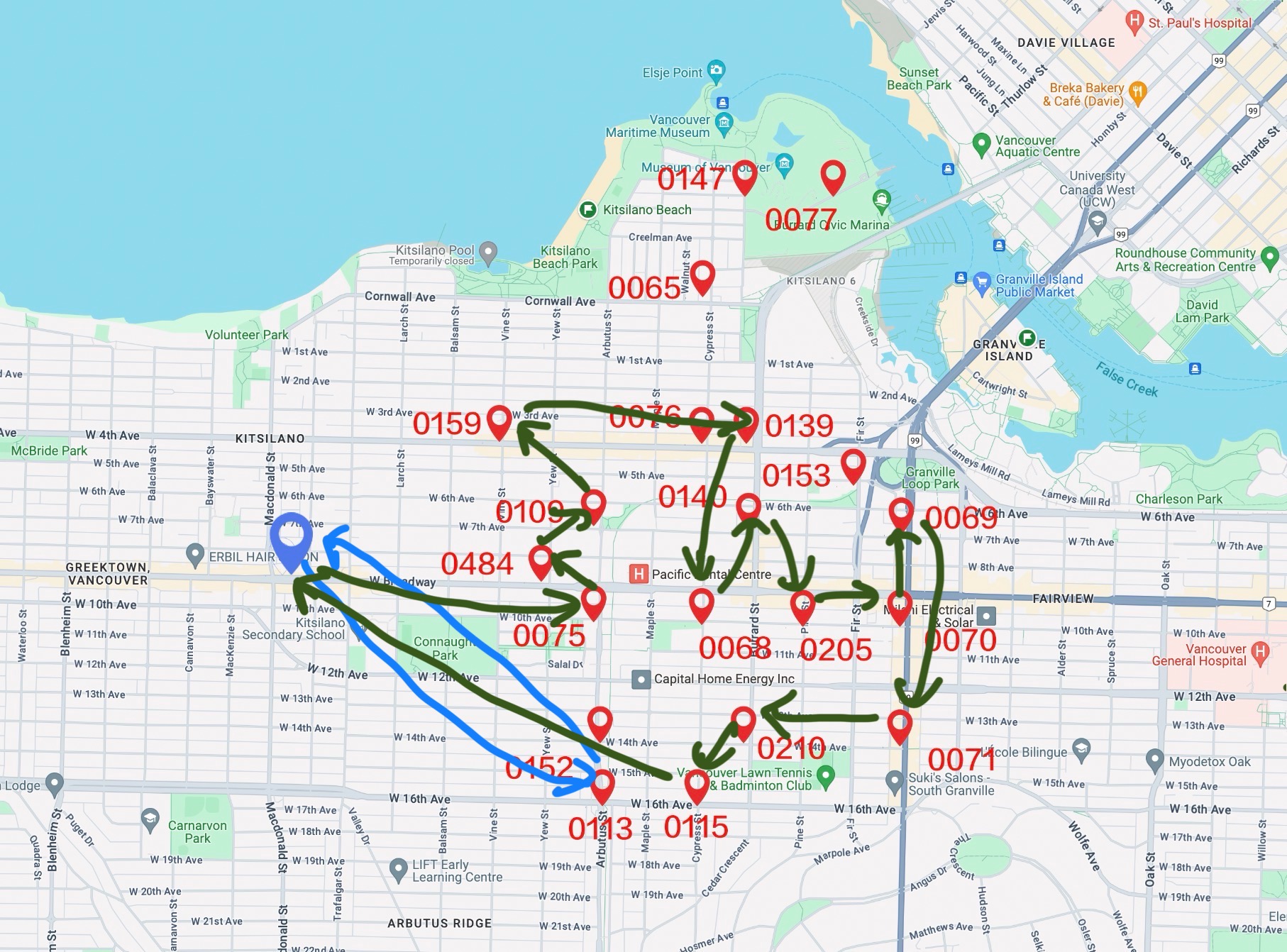


Figure 3.5 Two-Vans routes on 13th February

The routes generated by the model revealed that to minimize the total distance traveled, one van would be assigned to one station only, while the other van would follow a similar one-van model route to cover the remaining stations. While this solution effectively addresses the capacity constraint, prioritizing completion time poses a new challenge. In this scenario, the longer route of the solution should be minimized, contrasting with the current approach. This suggests the need for a new objective function to be formalized.

**4. Conclusions and discussion**

In conclusion, our project has provided valuable insights into optimizing the redistribution of bikes in a bike-sharing system. Through the utilization of integer programming, we have developed a robust model capable of generating efficient repositioning routes while considering various constraints of the task and objectives.

Our comparative analysis between the integer programming and traditional Traveling Salesman Problem (TSP) approaches has demonstrated the superiority of our model in addressing real-world constraints and achieving optimal solutions. By leveraging integer programming, we have not only minimized the total distance traveled by the repositioning vans but also ensured the completion of tasks at each station, thus enhancing the overall efficiency and effectiveness of the bike-sharing system.

**4.1 Limitations**

Firstly, the computational constraints that led us to analyze a subset of 20 stations instead of the entire pool of 250 stations could potentially impact the generalizability of our findings. The dynamics of bike-sharing systems may vary across different station groupings, and one future direction could be to incorporate clustering algorithms to divide the whole station system into several zones and

research could explore ways to overcome these computational constraints to allow for a more comprehensive analysis.

Additionally, our assumption that we redistribute bikes after midnight each day may oversimplify the real-world scenario. While our static approach aligns the minimal demand during this period, it does not consider unforeseen demands or variations in user behavior during daytime. Future studies could explore more dynamic models that consider real-time repositioning during the day.

**4.2 Future Work**

Several avenues for potential future developments emerge from our project. Firstly, integrating machine learning to predict bike-sharing demand based on factors such as special events and weather patterns holds significant promise. By harnessing advanced algorithms, we could enhance our model's predictive accuracy, thereby optimizing redistribution strategies to better serve fluctuating demand.

Secondly, expanding the scope of our studies to include all 250 stations, as well as considering the deployment of multiple vans instead of a single one, represents a crucial step towards simulating real-life scenarios more accurately. This broader approach would provide a more comprehensive understanding of system dynamics, enabling us to refine our optimization strategies to better accommodate the complexities of urban mobility.

Thirdly, incorporating detailed information on carbon footprints into the system could further enhance its efficiency and sustainability. By quantifying the environmental impact of bike redistribution strategies, we can provide valuable insights to the company, empowering them to make informed decisions that prioritize eco-conscious practices.

In conclusion, these potential future developments have the capacity to significantly enhance the effectiveness and sustainability of bike-sharing systems. By embracing machine learning, broadening our studies, and prioritizing environmental considerations, we can work towards creating more resilient and environmentally friendly urban transportation solutions.

**4.3 Reflections**

Edie:

Through this project, I gained valuable experience in transforming theoretical concepts such as integer programming and constraint problems into tools for solving real-world issues. Previously, my application of integer programming was limited to scheduling issues, but now I realize there are many more scenarios where it can be applied.

The collaborative process was particularly enjoyable. The project topic was initiated by Jun, and we continued to refine it through constant brainstorming. For instance, we encountered difficulties when our vehicle's route formed two loops, prompting us to consult academic papers to find a solution.

We also incorporated methodologies learned in other courses, such as using APIs to measure distances and employing Python’s turtle library for visualization, enriching the outcomes and presentations of the project. Compared to previous course programs, I find this project to be more comprehensive and engaging.

Mengxu:

Throughout our project on designing a bike-sharing system using linear programming and constraint satisfaction techniques, my role primarily focused on processing real-life data from the Mobi by Rogers website and discerning the underlying trends in bike flow. This task proved to be enlightening, offering valuable insights into the complexities of urban mobility dynamics. One of the most intriguing aspects of our findings was the striking resemblance to a standard Traveling Salesman Problem (TSP) solution in certain scenarios. It was a fascinating process to discuss with teammates how our data selection may have contributed to this resemblance. Ultimately, we were able to expand our studies based on the similarities found. I thoroughly enjoyed working on this real-life problem and collaborating with this amazing team.

Jun:

This project has been an enlightening experience for me. As someone who enjoys solving puzzles, I found the challenge of an optimization problem in a real-world setting to be particularly intriguing. The project allowed me to stretch my knowledge on integer programming to solve a new practical problem, which was intellectually stimulating. Through the process of modeling the repositioning problem and implementing integer programming techniques, I gained a deeper understanding of optimization algorithms and their applications in real-life scenarios.

**References**

[1] “System data,” Mobi, https://www.mobibikes.ca/en/system-data (accessed Apr. 25, 2024).

[2] T. Raviv, M. Tzur, and I. A. Forma, “Static repositioning in a bike-sharing system: Models and solution approaches,” *EURO Journal on Transportation and Logistics*, vol. 2, no. 3, pp. 187–229, 2013. doi:10.1007/s13676-012-0017-6

**Appendix - A**

Python Code Notebook