Case Report

Hot Delivery

-- A Case Study in Online Food Delivery Platform

Lydia Li, Ryan Fu, Stark Wang, Sunny Chen, and Amber Hu

0. Executive Summary

This report discusses the strategic and operational problem of assigning drivers to orders based on estimates of their best-possible delivery routes for online food delivery platforms like Uber Eats, DoorDash, and SkipTheDishes. Although these companies have seen impressive year-on-year growth, they have never reported any profits due to heavy investments in expansion, ads, promotions, and discounts. The biggest expense these companies face is paying drivers, who receive a base fare based primarily on the distance they traverse. To address this issue, our team was hired by the data analytics group at Uber Eats to propose a solution that can reduce their delivery costs.

The problem is broken down into four parts that incrementally add more real-world features to the problem. The first part of our analysis focuses on the single-driver perspective, where we develop a model to determine the optimal delivery route for a single driver, given a set of pre-assigned orders. The second part incorporates time constraints into the model, taking into consideration factors such as food preparation time and customer wait time. This addition ensures that our proposed solution caters to the time-sensitive nature of the food delivery business.

The third part of our study generalizes the model to accommodate multiple drivers, allowing for a more realistic representation of the problem at hand. This extension enables us to explore the trade-offs between the total distance travelled by all drivers and the average waiting time for customers. Finally, in the fourth part, we develop a scalable heuristic to solve large-scale instances of the problem, applies k-means clustering to cluster orders based on geographical locations and applies the second part model to each cluster and the associated drivers to ensuring that our solution remains practical and applicable in real-world scenarios.

Throughout the report, we employ mixed-integer linear programming techniques using PuLP and Gurobi to solve the problem and derive insights into the various aspects of supply chain optimization for Uber Eats. Our objective is to provide a thorough understanding of the challenges faced by Uber Eats and propose actionable routing solutions that can help minimize delivery distance and improve customer satisfaction.

1. Introduction

The rapid growth of food delivery platforms such as Uber Eats, DoorDash, and SkipTheDishes has revolutionized the way people order groceries and interact with restaurants. With increasing market shares and expanding customer bases, these platforms face a critical challenge in optimizing their supply chain and delivery networks to minimize costs and ensure customer satisfaction. As the backbone of these services, efficient routing and assignment of drivers to orders play a vital role in reducing delivery costs and maintaining a high level of service quality.

In this report, we present a comprehensive analysis of the strategic and operational challenges faced by food delivery platforms, particularly focusing on Uber Eats, in assigning drivers to orders based on the best-possible delivery routes in the Greater Toronto Area (GTA). The study views the problem as a Traveling Salesman Problem (TSP) and is conducted in four parts, each addressing a unique aspect of the problem and incrementally adding real-world features to enhance the practical applicability of the proposed solutions.

2. Modeling Evolution

2.1. Model 1.0 – Single-driver perspective

2.1.1. Model Scenario Description

In Part I of the case study, Uber Eats wishes to preemptively assign a driver to multiple pickups based on the restaurant-customer distance. Our team is tasked with designing a model that produces the optimal delivery route for a single driver, assuming their next few future orders have already been assigned by another system.

Our team is operating under the assumption that Uber Eats has access to all relevant information regarding orders ahead of time which the objective is to minimize the distance traveled by the driver, who is assumed to be starting from the Rosedale neighborhood in Downtown Toronto. Furthermore, the driver can carry multiple orders, and once the driver finishes all deliveries, he/she will park the car in the last neighborhood where the last order has been delivered, waiting for future orders.

3 data files are available for this task. The Distance file contains information about the distances between various locations within GTA, including columns for the origin, destination, and distance between each pair of locations. To evaluate our model, we are provided with two instances in the dataset: part1_ordersA.csv and part1_ordersB.csv.

The Part1_OrderA file provides information on the location of a single restaurant and customer, meaning the route is designed for only one order, while the Part1_OrderB file contains information on the locations of multiple restaurants and customers, which means that multiple orders and pickup locations should be considered in the route.

We will describe our solutions for each case, including the route and total distance traveled by the driver. Overall, our model provides a solution that can significantly reduce delivery costs for Uber Eats by optimizing the delivery route for a single driver.

2.1.2. Mathematic Formula for the optimization of Model 1.0

Objective Function:

$$MIN \sum_{i=1}^{n} \sum_{j \neq i}^{n} \sum_{t=1}^{T} D(i,j) * x(i,j,t)$$

Subject to:

$\sum_{i=0}^{n} \sum_{t=0}^{T} x(i,j,t) = 1,$	$j = 1, \ldots, n$	(1)
$\sum_{i=0}^{n} x(i,j,t) = \sum_{k=1}^{n} x(j,k,t+1),$	$j = 1, . ., n, t = 0, \ldots, T-1$	(2)
$\sum_{j=1}^{n} x(0,j,0) = 1,$	$i = 0$	(3.1)
$\sum_{j=1}^{n} x(i,j,0) = 0\ ,$	$i = 1, \ldots, n$	(3.2)
$\sum_{m=0}^{T} \sum_{k=0}^{n} x(R_j,k,m) \geq \sum_{i=0}^{n} x(i,j,t),$	$j = customers, t = 0, \ldots, T$	(4)
$\sum_{i=0}^{n} \sum_{j=1}^{n_c} x(i,j,T) = 1,$	$j = customers$	(5)
$x(i,j,t) \in \{0,1\};$	$i = 0, \ldots, n;\ \ j = 1, . ., n;\ t = 0, \ldots, T$	(6)

In Model 1.0, we assume there are totally n unique spots that we need to visit, which includes both customers and restaurants. And our original spot is spot 0 in our model, and in general we will have at most T steps which equals the number of spots we had to visit. xijt is a binary decision variable that indicates whether the driver takes the route from location i t location j at step t (sequence of stops) during the delivery process, where location i and location j are not the same. d_{ij} represents the distance between location i and location j. The objective function minimizes the total distance travelled by the driver while completing all assigned deliveries, where dij is multiplied by the corresponding decision variables xijt to calculate the sum of the distances travelled in each route segment.

In the model, Constraint (1) is designed to ensure that each location is visited by the driver exactly once. However, there is a notable exception for a unique case where the driver's starting point coincides with one of the customer locations. In this scenario, the location we must visit will include the original spot 0. By ensuring that each location is visited by the driver only once, it eliminates the need for unnecessary backtracking or redundant travel between locations.

Constraint (2) guarantees that for each location visited by the driver, there is an equal number of incoming and outgoing routes. This constraint effectively eliminates any disconnected or disjointed sub-routes, ensuring that the driver's path forms a single, connected sequence of locations. Constraint (3) ensures that the Downtown Toronto (Rosedale) is the starting point of the route for the driver, in other words, driver cannot start with other location at time stamp 0. Constraint (4) ensures that the driver picks up the food from the restaurant before delivering it to the corresponding customer. Constraint (5) ensures that the driver will park the car in the customer's neighborhood at the last step.

2.1.2. Optimization Result and Limitations of Model 1.0

For Order set A, the model has produced an optimal solution with a total travelling distance of 4.65 km. Since there is only 1 order, the optimal route for Order set A starts from the origin, visits the restaurant to pick up the food, and finally delivers the food to the customer and park the car in the corresponding neighborhood.

(Figure 1):

Downtown Toronto (Rosedale) -> Downtown Toronto (Underground city) -> Downtown Toronto (Central Bay Street)

In the case of Order set B, which consists of several orders, the optimal solution leads to a minimum travel distance of 33.88 km. The route is as follows (Figure 2):

Downtown Toronto (Rosedale) -> Downtown Toronto (Central Bay Street) -> Downtown Toronto (Richmond / Adelaide / King) -> Downtown Toronto (Underground city) -> Downtown Toronto (St. James Town / Cabbagetown) -> York (Cedarvale) -> Central Toronto (The Annex / North Midtown / Yorkville) -> Etobicoke Northwest (Clairville / Humberwood / Woodbine Downs / West Humber / Kipling Heights / Rexdale / Elms / Tandridge / Old Rexdale) -> Etobicoke (South Steeles / Silverstone / Humbergate / Jamestown / Mount Olive / Beaumond Heights / Thistletown / Albion Gardens)

Model 1.0 has several limitations that need to be addressed for a more comprehensive and efficient food delivery system. The first limitation is that Model 1.0 does not take into account time constraints such as food preparation time, customer wait time, and driver's waiting time at pickup locations. This leads to inefficient routing and potentially dissatisfied customers. The second limitation is that the model assumes there is only 1 single driver, which oversimplifies the real-world scenario where multiple drivers start from different locations. The third limitation is that Model 1.0 does not easily scale to accommodate a larger number of orders, drivers, and locations, limiting its applicability to real-world, large-scale food delivery operations.

2.2. Model 2.0 – A matter of time

2.2.1. Model Scenario Description

The goal of Part II in the case study is to integrate time into the model established in Part I. In the context of food delivery, it is important to minimize the time between when the order is ready and when it is delivered to the customer. To achieve this, restaurants provide an estimated time for when the food will be prepared for pick-up, which is used to determine the customer's waiting time - the period between when the order is ready and when the driver arrives at the customer's location. The model accounts for a five-minute average wait time for the driver, and also imposes a constraint "W" on the maximum average waiting time of the orders assigned to the driver.

2.2.2. Mathematic Formula for the optimization of Model 2.0

Objective Function:

$$MIN \sum_{i=1}^{n} \sum_{j \neq i}^{n} \sum_{t=1}^{T} D(i,j) * x(i,j,t)$$

Subject to:

$D_0 \geq \sum_{j=1}^{n_r} availableTime_j * x(0,j,0),$ (7)

$D_t \geq D_{t-1} + \sum_{i=0}^{n} \sum_{j=1}^{n} D_{ij} * x(i,j,t) + 5 * \sum_{i=0}^{n} \sum_{j=1}^{n_c} x(i,j,t), \quad t = 1, \ldots, T$ (8)

$D_t \geq \sum_{i=0}^{n} x(i,j,t) * availabelTime_j, \quad t = 1, \ldots, T, \ j \in restaurant$ (9)

$Wj \geq D_t - 5 - availableTime_j - M * (1 - \sum_{i=0}^{n} x(i,j,t)), \quad t = 1, \ldots, T, \ j \in customer$ (10)

$Wj \leq D_t - 5 - availableTime_j + M * \left(1 - \sum_{i=0}^{n} x(i,j,t)\right), \quad t = 1, \ldots, T, \ j \in customer$ (11)

$\sum_{j=1}^{n_c} W_j \leq W * n_c,$ (12)

Model 2.0 has the same objective function as Model 1.0, which minimizes the total travelling distance. Besides the 6 constraints in Model 1.0, Model 2.0 has 6 more constraints.

We first standardize all the time stamp to minutes based on the earliest time records in the system, in addition, two more generic variables Wj and Dt are added to the model.

Dt variable indicates the time elapsed since the driver's departure from the starting point at step t and Wj variable indicates the jth customer's waiting time.

Constraint (7) ensures that the actual arrival time for the first step must be later than or equal to the ready time of the first selected restaurant. Constraint (8) guarantees that for any time t, the actual arrival in step t is later than or equal to the time in the previous step plus the travel time in step t, and if the destination is a customer, an extra 5 minutes of wait time will be added. Constraint (9) ensures that the actual arrival time for any step must be later than or equal to the ready time of the restaurant if the order was picked up at step t. Constraint (10, 11) calculates the wait time for each customer using the Big M method, which involves computing a binary variable and a continuous variable. It subtracts the order time from the arrival time, and additionally, the 5-minute customer waiting time is excluded from the calculation. Constraint (12) ensures that the driver does not exceed a maximum average wait time for customers of W minutes.

2.2.3. Optimization Result and Limitations of Model 2.0

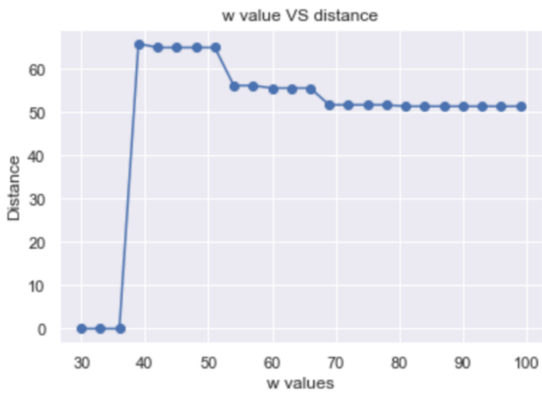
For Order A, the model has produced an optimal solution with a total travelling distance of 42.77 km. The generated optimal route for Order A is (Figure 3):

Downtown Toronto (Rosedale) -> Central Toronto (North Toronto West) -> Etobicoke (Westmount) -> Scarborough (Kennedy Park / Ionview / East Birchmount Park) -> Scarborough (Woburn)

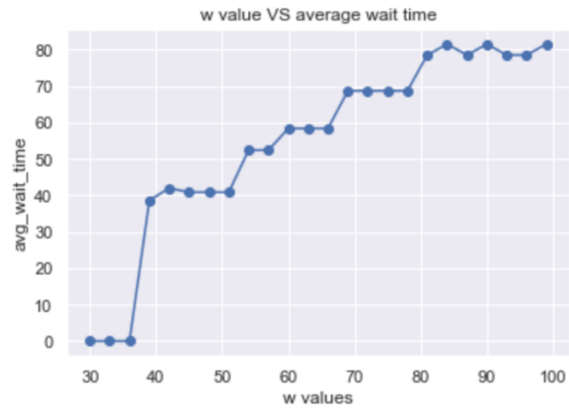
The optimal solution for Order B results in a minimum travel distance of 51.4 km. The generated optimal route for Order B is (Figure 4):

Downtown Toronto (Rosedale) -> North York (Sweeney Park / Wigmore Park) -> Scarborough (The Golden Mile / Clairlea / Oakridge / Birchmount Park East) -> Downtown Toronto (Rosedale) -> Etobicoke (Westmount) -> Etobicoke (Islington Avenue) -> Etobicoke (West Deane Park / Princess Gardens / Martin Grove / Islington / Cloverdale) -> Downtown Toronto (University of Toronto / Harbord) -> Downtown Toronto Stn A PO Boxes 25 The Esplanade (Enclave of M5E)

2.2.4. Sensitivity Test On W



(a) **W and Distance Tradeoff Curve**

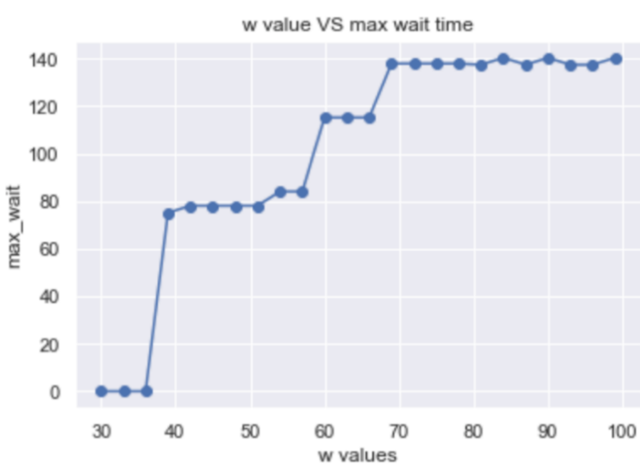


(b) **W and Average Wait Time Tradeoff Curve**

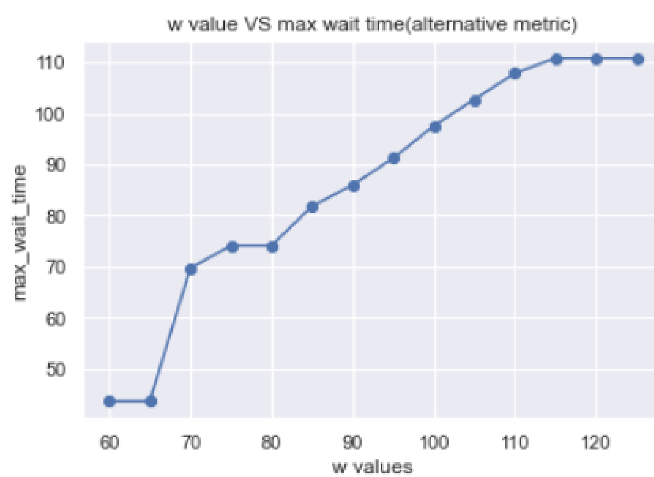
When W is smaller than 40 minutes, the model becomes infeasible because the limit on the maximum average waiting time of orders assigned to the driver is too small. This means that there is not enough time for the driver to reach the customers' locations and wait for them to pick up their orders.

As W increases, the shortest distance decreases because the optimization algorithm can select a shorter path by allowing customers to wait longer with a relaxed average waiting time constraint. However, when W is larger than 70 minutes, the shortest distance does not decrease further because the shortest path has already been chosen, regardless of the amount of time provided.

2.2.2. Problem of Average Waiting Time and Alternatives



(a) **W and Max Wait Time Tradeoff Curve**



(b) **W and Max Wait Time Tradeoff Curve**

Considering that when we use average waiting time as a metric, the maximum waiting time is around 140 minutes as W increases, this indicates that some customers must wait a long time for their orders. We propose changing the wait time evaluation so that the longest waiting time for any order must be less than a certain value (W). Intuitively, this is fairer for customers, as they will wait less compared to others. The max time approach is generally more stringent than the average wait time, and as a result, we choose a different scope for the W value. In general, the max wait time in the alternative metric is 30 minutes less than the average wait time approach. This improvement is desirable as it provides a more equitable distribution of waiting times among customers, thereby improving the overall customer experience.

2.3. Model 3.0 – The more the merrier

2.3.1. Model Scenario Description

In Part III of the case study, the objective is to generalize the model from Part II to multiple drivers and assign orders to them while routing. Each driver has a specific velocity based on their mode of transportation and starting location. The aim is to minimize the total distance traveled by all drivers while considering the average waiting time constraint from Part II. Our solution will assign orders to multiple drivers and route them efficiently to optimize the delivery process.

2.3.2. Mathematic Formula for the optimization of Model 3.0

Model 3.0 has a similar objective function to Model 2.0, which minimizes the total travelling distance. Given that in this model we introduce more than one driver compared with the other models, we added variable *s* which indicates each driver. Here, SS (start_region) stands for the locations of all restaurants, customers and starting locations of each driver; D standards the number of drivers; n standards the locations of all restaurants, customers; R stands for all the restaurants; C stands for all the customers in the formula listed above.

Constraint (1) makes sure that in the end, the drivers will visit every restaurant and customer once: each restaurant and customer must be visited exactly once by any driver, except for the starting locations (starting regions), which can be visited zero or one times.

Constraint (2) ensures that for each restaurant, the number of drivers entering is equal to the number of drivers leaving. For customers, the number of drivers entering must be greater than or equal to the number of drivers leaving, allowing drivers to stop at customer locations.

Constraint (3) ensures that every driver, at the original step (t=0), if the start point of route is in start_region, only one route or no route is allowed. If they start at any other location, they are not allowed to visit next.

Constraint (4) ensures that for each driver at each step, they can choose to move to another location or not move at all. This ensures that a driver cannot visit multiple locations at the same time.

Constraint (5) is a Step Constraint: for each driver, the number of locations visited at step t must be greater than or equal to the number of locations visited at step t+1. This ensures that drivers do not visit more locations in later steps than in earlier ones.

Constraint (6) ensures that each driver visits restaurant before customer: A driver must visit the corresponding restaurant before visiting a customer. This ensures that food is picked up before it is delivered.

Constraint (7) makes sure that for customers, we allow inflow >= outflow so that drivers can stop at customers.

Constraint (8) makes sure that there is no self-loop drivers cannot visit the same location twice in the same step.

Constraint (9) guarantees that for any time t, the actual arrival in step t is later than or equal to the time in the previous step plus the travel time in step t, and if the destination is a customer, an extra 5 minutes of wait time will be added.

Constraint (10) calculates the wait time for each customer using the Big M method, which involves computing a binary variable and a continuous variable. It subtracts the order time from the arrival time, and additionally, the 5-minute customer waiting time is excluded from the calculation.

Constraint (11) ensures that the driver does not exceed a maximum average wait time for customers of W minutes.

2.3.3. Optimization Result of Model 3.0

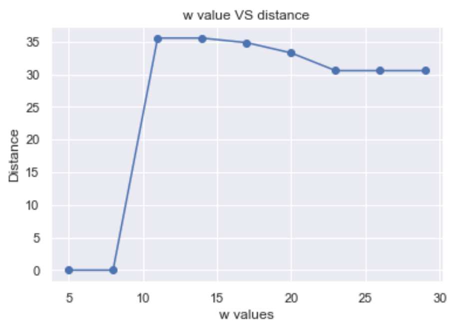
The optimal solution for the set of orders results in a minimum travel distance of 30.54 k m. This solution involves assigning the orders to three drivers, with each driver followi ng a specific route as outlined below (Figure 5):

Driver 0 (blue route as figure below showed): Downtown Toronto (Richmond / Adelaid e / King) -> Downtown Toronto (Kensington Market / Chinatown / Grange Park)->Dow ntown Toronto (Central Bay Street)->Downtown Toronto (Ryerson)->York (Fairbank / O akwood)->North York (Armour Heights / Wilson Heights / Downsview North)

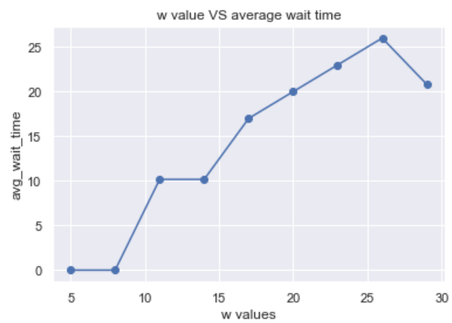
Driver 1 (red route as figure below showed): Downtown Toronto (St. James Park) -> Eas t Toronto (The Beaches)

Driver 2 (yellow route as figure below showed): Downtown Toronto (Church and Welle sley)-> Downtown Toronto (Christie)->West Toronto (Brockton / Parkdale Village / Exhi bition Place)

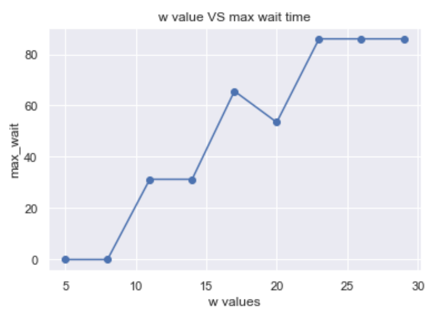
2.3.4. Sensitivity Test



(a) W and distance tradeoff curve



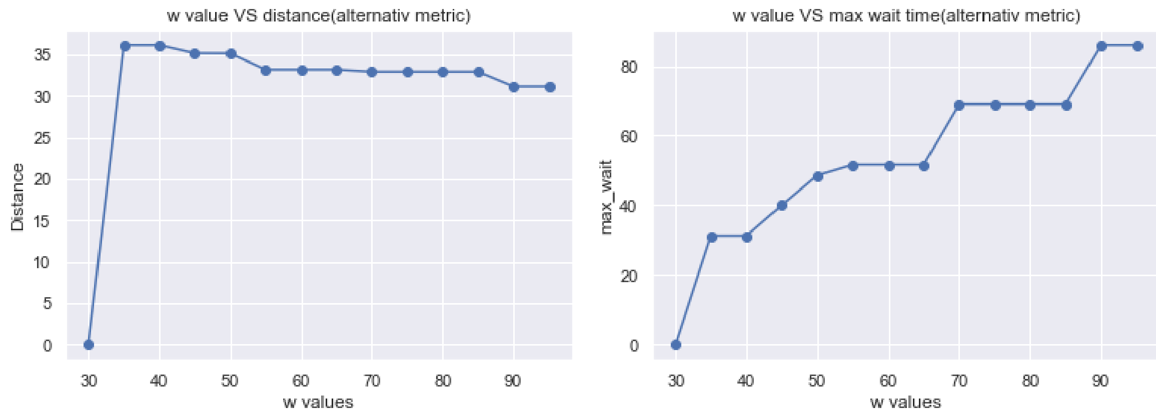
(b) W and average wait time tradeoff curve



(c) W and Max Wait Time tradeoff curve

Compared to the tradeoff curves generated by outputs from Model 2, Curve (a) demonstrates that the feasible distance occurs when W is above 8 minutes, which is significantly smaller than the W threshold in the second model, which is above 40 minutes. This indicates that an increase in drivers has considerably reduced the average waiting time overall. Moreover, the optimal minimized distance is achieved when W reaches approximately 23 minutes.

Curve (b) shows a monotonic increasing relationship between W and average wait time. The primary difference compared to the corresponding curve in Model 2.0 is that the feasible distances occur much earlier, indicating that an increase in drivers within the system significantly improves efficiency in terms of delivery time. However, a similar issue arises, as demonstrated by Curve (c): the maximum waiting time for an individual is much higher than the average waiting time, which may lead to low customer satisfaction. To address this issue, we have introduced a new metric, maximum waiting time, as a solution. The graphs displayed below indicate that the problem has been successfully resolved. Now, we need to relax W to be even above 90 to reach a maximum wait time of more than 80 minutes, which contrasts with the previous situation where this threshold was reached when W was only at 25.



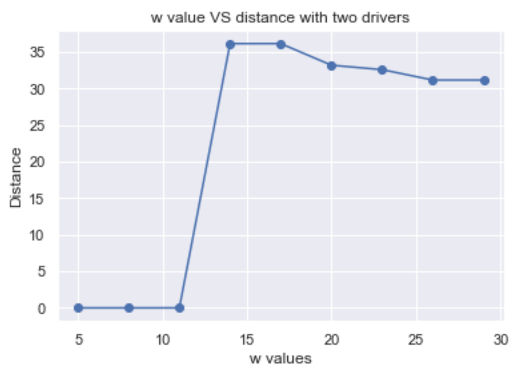
(a) Max wait time and distance tradeoff curve **(b) W and max wait time tradeoff curve**

When comparing different scenarios with varying numbers of drivers, it becomes clear that the number of drivers has a direct impact on the feasible solutions and the system's overall efficiency. As seen in the analysis of the curves, the fewer drivers there are, the longer it takes for the system to reach a feasible solution.

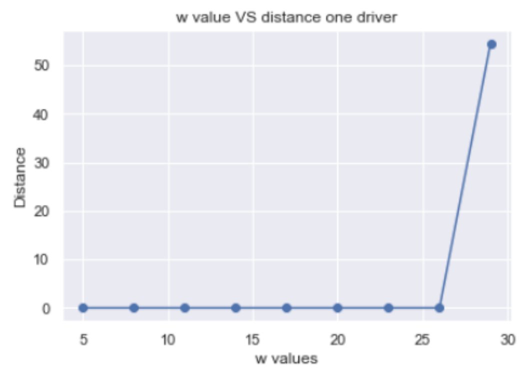
In Curve (a), the feasible solution occurs at around 14 minutes, while in Curve (b), it occurs even later, at around 27 minutes. This trend indicates that as the number of drivers decreases, the overall efficiency of the delivery process is negatively affected which takes longer time to reach the optimal solution.

Moreover, as the feasible solution is reached later and later, the convergence time for the optimal solution increases. In other words, it takes more time for the system to reach the point where it is operating at its most efficient level, given the constraints of driver availability and delivery demand.

In summary, the number of drivers has a substantial impact on the efficiency of a delivery system. As the number of drivers decreases, the W value needed to find a feasible and optimal solutions increases, which can negatively affect customer satisfaction and the overall performance of the delivery service. Consequently, it is crucial for delivery service providers to maintain an adequate number of drivers to ensure efficient operations and high levels of customer satisfaction.



(a) **W and distance tradeoff with 2 drivers**



(b) **W and distance tradeoff with 1 drivers**

2.4. Model 4.0 - Scaling

2.4.1. Model Scenario Description

In Parts I-III, we developed MILP models that tackle single-driver and multi-driver routing problems while considering time constraints. However, these models might face difficulties when applied to large-scale instances. In Part IV, we propose a scalable heuristic approach that builds upon the insights gained from the previous parts to address this challenge, while keeping W fixed at 120.

2.4.2. Methodology

Firstly, we preprocess the data by merging the provided datasets containing information on orders, drivers, and regions. This creates a unified dataset that includes essential order and driver information along with their respective latitude and longitude coordinates. This step ensures that we have a clean and organized dataset to work with, which is critical for the success of our proposed heuristic approach.

The second step involves using K-means clustering to group orders based on their average latitude and longitude, calculated as the midpoint between the restaurant and customer locations. We set the number of clusters equal to the number of available drivers and define the initial centroids as the drivers' starting locations. The clustering process assigns each order to a cluster, effectively dividing the orders among the drivers. This step allows us to partition the orders in a manner that minimizes the overall distance covered by all drivers.

Following the clustering process, we move on to the driver assignment. In this step, we calculate the Euclidean distance between each driver's starting location and the centroids of the clusters. Then, we assign each driver to the nearest unique cluster. The driver assignment step aims to minimize the initial distance traveled by drivers before starting their delivery routes, ultimately enhancing the overall efficiency of the delivery process.

Once the orders are assigned to drivers, the next step is to determine the optimal delivery routes within each cluster. We employ Model 2.0 for this purpose, which prioritizes orders based on their proximity to the current location and their time constraints, allowing drivers to complete deliveries efficiently and in a timely manner.

2.4.3. Optimization Result of Model 4.0 and Comparison with Model 3.0
In comparing the results of Model 3.0 and Model 4.0, we can observe some differences in the optimal routing solution using the part 3 data. Model 3.0 yields a total distance travelled by drivers of 19.37 and an average customer wait time of 21.78 minutes. In contrast, Model 4.0 results in a longer total distance travelled by drivers, amounting to 38.34, and a slightly higher average customer wait time of 22.24 minutes. These differences may be attributed to the updated constraints and variables introduced in Model 4.0 to better optimize the routing solution.

When applying Model 4.0 to the larger instance of part4_large.csv and part4_drivers.csv with a wait time constraint (W) of 120 minutes, the optimal route solution reveals a total distance travelled by drivers of 100.66 and an average customer wait time of 83.47 minutes. This showcases the model's ability to adapt to larger datasets while still providing reasonable results. In conclusion, Model 4.0 offers a more comprehensive routing solution, albeit with some trade-offs in terms of total distance and wait time, as it better accounts for various factors involved in the routing process.

3. Conclusions

In conclusion, this report addresses the strategic and operational challenges faced by online food delivery platforms like Uber Eats, DoorDash, and SkipTheDishes in optimizing delivery routes to reduce costs and improve customer satisfaction. By breaking down the problem into four parts and incrementally adding real-world features, our team developed a comprehensive model that caters to the single-driver and multi-driver perspectives, incorporates time constraints, and provides scalable solutions for large-scale instances. Through the use of mixed-integer linear programming techniques and employing PuLP and Gurobi, the report offers actionable routing solutions that balance delivery distance and customer wait time.

Comparing the results of different model iterations, we observed trade-offs between total distance traveled by drivers and average customer wait time. The final model, Model 4.0, better accounts for various factors involved in the routing process and adapts to larger datasets while providing reasonable results. Although it yielded slightly longer distances and wait times compared to Model 3.0, the comprehensive nature of Model 4.0 makes it a more effective solution for real-world applications. The insights derived from this study can be instrumental in driving operational efficiency and enhancing the overall performance of online food delivery platforms.

References

Model 3.0

Objective Function:

$$MIN \sum_{s=1}^{D} \sum_{i=1}^{SS} \sum_{j\neq i}^{n} \sum_{t=0}^{T} D(i,j) * x(i,j,t,s)$$

Subject to:

$\sum_{i=1}^{n} \sum_{t=0}^{n} x(i,j,t,s) = 1,$ $\quad j = 1,\dots,n$ *and not in starts* (1)

$\sum_{i=1}^{n} \sum_{t=0}^{n} x(i,j,t,s) \leq 1,$ $\quad j = 1,\dots,n$ *and in starts* (1)

$\sum_{s=1}^{Drivers} \sum_{i=1}^{n} x(i,j,t,s) = \sum_{s=1}^{Drivers} \sum_{k=1}^{n} x(j,k,t+1,s),\ j = 1,..,n, t = 0,1,\dots,T-1$ (2)

$\sum_{j=1}^{j\ not\ in\ starts} x(starts[s],j,0,s) = 1,$ s = 1, 2, …, D (3)

$\sum_{j}^{n} \sum_{i=1}^{i\neq starts[s],} x(i,j,0,s) = 0\ ,$ s = 1, 2, …, D (3)

$\sum_{t=1}^{T} \sum_{j=1}^{n} x(starts[s],j,t,s) = 0\ ,$ s = 1, 2, …, D and $starts[s]$ not in spots (3)

$\sum_{j=1}^{n} \sum_{i=1}^{i=starts} x(i,j,t,s) \leq 1\ ,$ s = 1, 2, …, D, $,t = 1,\dots,T$ (4)

$\sum_{j=0}^{t+1} \sum_{i=0}^{SS} x(i,j,t,s) \geq \sum_{j}^{n} \sum_{i=0}^{SS} x(i,j,t+1,s),$ $\quad s = 1,..,D, t = 1,\dots,D$ (5)

$\sum_{t=0}^{T-1} \sum_{j=1}^{C} \sum_{s}^{D} x(i,j,t,r) \geq \sum_{i}^{SS} x(i,j,t,s),$ $\quad j = 1,..,n, t = 1,\dots,T,$ s = 1, 2, …, D (6)

$\sum_{\substack{i=1 \\ i\neq j}}^{SS} x(i,j,t,s) \geq \sum_{k}^{n} x(i,k,t+1,s),$ $\quad j = 1,..,n, t = 1,\dots,T,$ s = 1, 2, …, D (7)

$\sum_{\substack{i=1 \\ i=j}}^{SS} \sum_{j=1}^{n} x(i,j,t,s) = 0,$ $\quad t = 1,\dots,T,$ s = 1, 2, …, D (8)

$D(0,s) \geq \sum_{j=1}^{R} x(S(s),j,0,s) * available\ time(j)\ ,$ s = 1, 2, …, D (9)

$D(t,s) \geq D(t-1,s) + available\ time(j) * \sum_{\substack{i=1 \\ i\neq j}}^{SS} \sum_{j=1}^{C} x(i,j,t,s)\ ,$ $\quad t = 1,\dots,T,$ s = 1, …, D (9)

$W_j = D(t,s) - 5 - available\ time(j) - M * \left(1 - \sum_{i=1}^{SS} x(i,j,t,s)\right), j\ = all\ customers, 1,\dots,T,$ s = 1, …, D, $j = 1,\dots,R$ (10)

$\sum_{j=1}^{customers} W_j \leq W,$ $\quad t = 1,\dots,T$ (11)

$Wj \geq 0,$ $\quad j = 1,..,n$ (12)

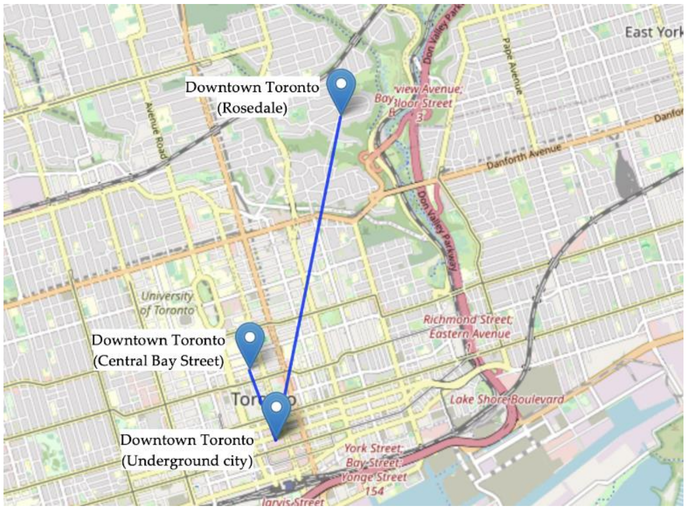


Figure 1. Optimal Route for Part 1 Order A

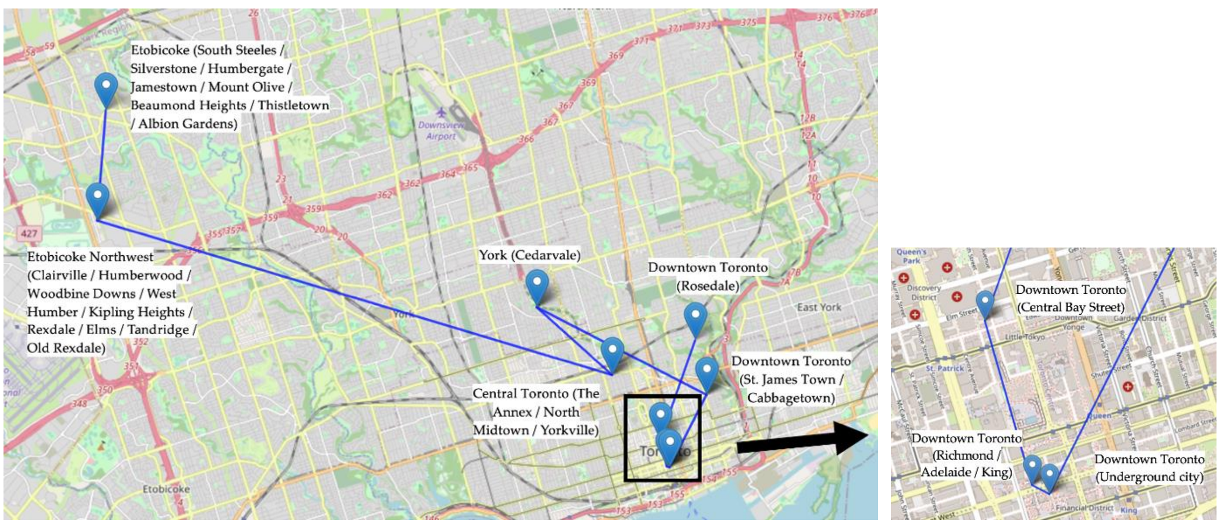


Figure 2. Optimal Route for Part 1 Order B

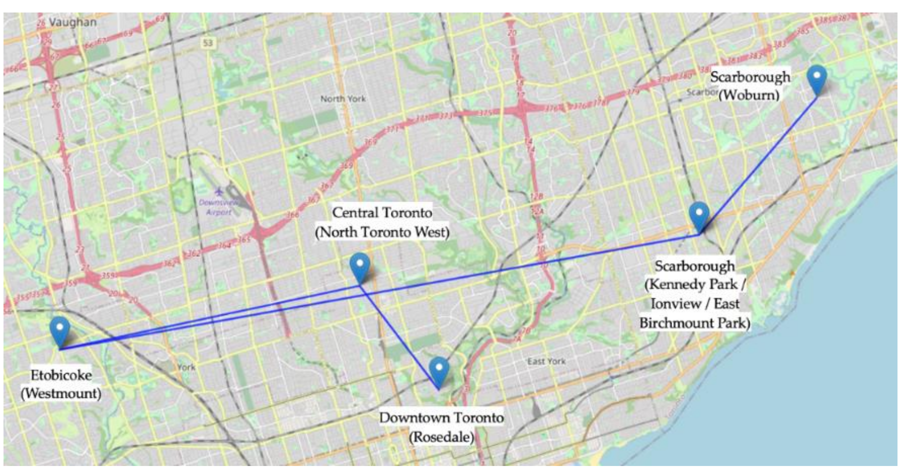


Figure 3. Optimal Route for Part 2 Order A

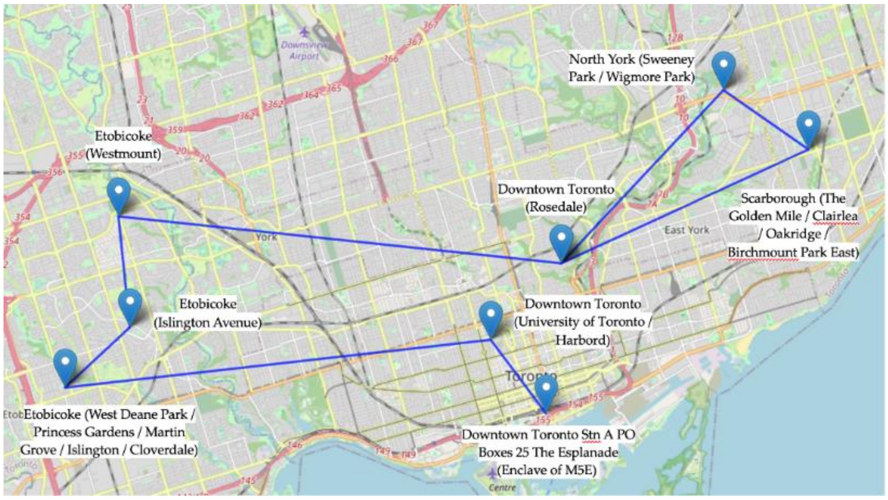


Figure 4. Optimal Route for Part 2 Order B

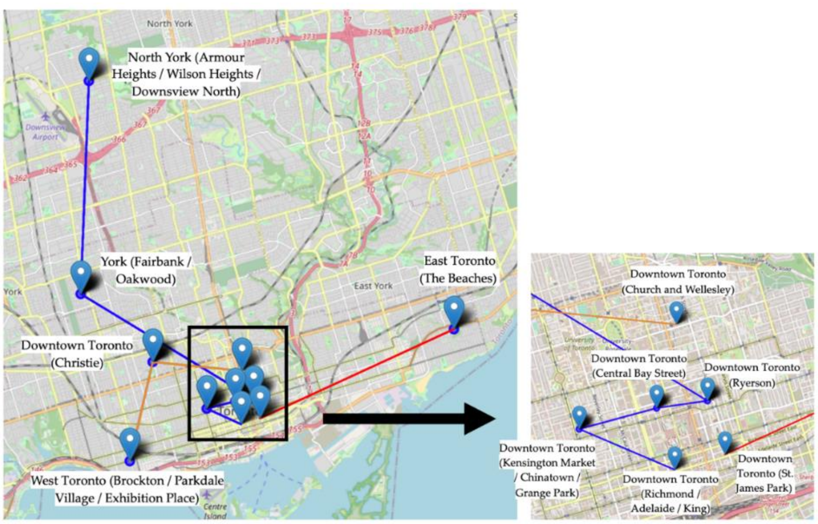


Figure 5. Optimal Route for Each Driver in Part 3