The Vehicle Routing Problem for Save-On-Foods E-Commerce Grocery Delivery

Ashutosh Dubal[1], Kevin He[1], Lacey Liang[1], Sherlyn Ng[1], and Tu Trinh[1]

[1]Department of Mathematics, Simon Fraser University, Burnaby, BC, V5A 1S6

ABSTRACT

E-commerce is an increasingly popular form of commerce for groceries chains, allowing customers to place grocery orders online for delivery within a specific time window. The success of an e-commerce platform depends on the reliability of deliveries over the specified time window, ensuring all orders are delivered on time. However, creating the vehicle routes becomes a challenge as the number of constraints scales exponentially with the number of orders placed, making it difficult to solve the vehicle routing problem (VRP)[1]. Moreover, feasibility issues arise as business considerations, including delivery drop-off time and driver labour constraints need to be included. To overcome these routing challenges, we create an integer programming model that employs aspects of the VRP with time windows and the capacitated VRP and use local search algorithms to find a feasible local optimal solution. We also employ a penalty system to drop infeasible locations. These locations are then re-introduced to the solution. The performance is compared to the current delivery routing system. The findings of this work can be applied to other delivery services for analysts to evaluate their current delivery fleet size and improve their service area design.

Keywords: Vehicle Routing Problem with Time Windows; Capacitated Vehicle Routing Problem; Delivery Routing; Grocery E-commerce

1 Introduction

We collaborated with Save-On-Foods, a Canadian supermarket chain that offers e-commerce services for customers to shop online for their grocery order, select a delivery slot in which the order will arrive, and they will be handpicked and delivered to their doorstep within the selected delivery time slot. The delivery areas that Save-On-Foods offers are divided into catchments based on the area's postal code, and each store is responsible for managing the e-commerce orders and deliveries for the orders placed within their assigned catchment.

The delivery routing software currently in place uses methods to cluster its customers then applies a routing algorithm within each cluster. Delivery routes are evaluated at a centralized location, taking approximately 5 minutes for the algorithm to generate a delivery route for a single driver. Employees have noted that half of the computed delivery routes require manual adjustments, with it taking 3 minutes per store.

This paper aims to propose a robust delivery routing model that accounts for the unique constraints at each location while reducing the total amount of adjustments needed, the total travel time of each vehicle, and the total number of drivers required to fulfill all deliveries. The performance of the proposed model is compared to the existing delivery routing software used by Save-On-Foods.

1.1 Background and Significance

Save-On-Foods's e-commerce platform has recently been one of the fastest-growing areas in terms of sales in the organization. With its significant growth, issues arise with the scalability of the e-commerce delivery system as additional resources are required to manage the increased e-commerce demand at each store[2]. With limited resources, drivers have reported an increase in deliveries per shift, with some drivers taking a shortened lunch break to complete all deliveries within their respective time windows. The reliability of delivery drop-off times is impacted by an increased number of deliveries, as outside factors including traffic and time spent fulfilling the order at each location are difficult to estimate. This leads to an increased number of deliveries arriving outside the chosen time window. Maintaining accurate delivery times is key to customer satisfaction and affects repeated usage of the Save-On-Foods's e-commerce platform[3].

The delivery routing software used by Save-On-Foods is a black box and does not contain features related to delivery driver workload. For store locations with more than one vehicle, the delivery load in each vehicle may vary significantly with delivery drivers reporting a maximum vehicle load difference of 11 units within the same shift. The large load difference provides drivers with an imbalanced workload where some drivers complete their deliveries within a significantly shorter time frame than other drivers. Incorporating delivery load balancing between the vehicles will create a more uniform workload for the delivery drivers, which will reduce the variance in labour between shifts and promotes fairness between delivery drivers[4].

The current delivery routing software also does not consider minimizing the number of drivers required for a given shift. Routes are generated with all of the available delivery drivers at the store location, despite the workload being able to be completed by fewer drivers. By minimizing the total number of drivers required for each shift, Save-On-Foods can reduce employment costs related to having more delivery drivers than required and reduce costs related to vehicle maintenance and gas usage.

A more robust and optimized delivery routing model would allow for reduced labor costs associated with adjusting the outputted routes and would improve the individual vehicle routes, allowing for additional buffer times between deliveries. The proposed model aims to ensure a full break duration for the driver and for deliveries to be made within their respective time window to maintain customer satisfaction.

1.2 Scope of Study

Save-On-Foods operates over 170 stores across Canada. However, we focus on a subset of 4 stores spanning 2 Provinces covering high-demand high-density areas, high-demand medium-density areas, medium-demand medium-density areas, and low-demand low-density areas.

The model is evaluated based on synthetic e-commerce order data. The order data includes the delivery time slot for each customer location and a time matrix with the driving travel time between each pair of locations as its entries. The time matrix is calculated using the Google Distance Matrix API on the Google Maps Platform which factors in the maximum and average driving speeds on each street. Traffic conditions were estimated using the traffic data from the middle of the delivery shift and are assumed to be consistent throughout the entire shift[5]. Distance data is not considered in this study as the focus is to ensure that deliveries are completed within their respective time window.

A more accurate time matrix would allow better estimates of travel times and would provide a more accurate delivery routing schedule for the drivers to use. However, constructing a precise time matrix is difficult as traffic conditions change hourly, and delays from road closures and traffic accidents would require real-time adjustments to the delivery route. We assume that the time matrix is an accurate reflection of traffic patterns and road conditions in the entire delivery shift and that the time matrix is updated with forecasted road conditions prior to computing the driver delivery routes.

This study focuses on the delivery routing operation within a well-defined area for a single store. Each store is responsible for e-commerce delivery orders placed within their assigned catchment and the catchment areas are predefined by Save-On-Foods. The customer only has access to the store inventory within their catchment when placing their order and cannot place orders for items available at another store location, but not available at their assigned store due to business logistic constraints.

The model also assumes that after arriving at a delivery location, the driver will complete the delivery drop-off within a 12-minute window. This time window is known as the service time. The service time fluctuates by store location and delivery areas, with most stores having a 12 minute service time. We assume zero variability of the service time for this study.

2 Data Description

2.1 Store Data

The data provided is from Save-On-Foods. It contains information about four store locations with varying demand in various geographical regions.

Deliveries are completed from 7:00 AM to 10:00 PM, and two sets of drivers are required at each store to service over the entire delivery time. Driver shifts are split into two time intervals: one AM shift and one PM shift. Each shift requires a 30-minute lunch break, and this break is expected to be completed sometime between the fourth and fifth work hour after the shift start time. Information about the delivery time and depot departure time for each shift can be found in Table 1.

Customers must select a delivery slot time that the grocery Save-On-Foods offers. The orders will arrive within the delivery slot time frame, and customers are assumed to be available to receive the delivery. Each time frame spans over 2 hours except for two 7 hour time frames. After the delivery driver has arrived at a customer location, a service time is provided to allow the driver to unload the delivery vehicle and fulfill the order. The driver is expected to return to their

Table 1. Delivery Time and Earliest Departure by Shift

Shift Type	Shift Start Time	Earliest Departure Time	Delivery Times
AM	6:00 AM	6:45 AM	7:00 AM - 2:00 PM
PM	2:00 PM	2:45 AM	3:00 PM - 10:00 PM

respective depot after completing all deliveries.

Each vehicle has a maximum capacity of 21 orders, and each driver is expected to deliver up to the vehicle capacity during their shift. From this, each delivery slot is assigned a capacity by Save-On-Foods to prevent surges of deliveries within a time window.

At the end of the AM shift, the driver returns to the depot where the vehicles are inspected, and new delivery orders are loaded to the returned vehicles for the PM shift delivery drivers. Information about the maximum capacities for each time slot for a store with one AM and PM delivery driver is shown in Table 2. In this study, it is assumed that for more than one delivery driver, the maximum capacity per delivery time frame increases by a factor of the number of delivery drivers in that shift.

Table 2. Delivery Time Slots and Capacity for One Driver

Shift	Time Frame	Maximum Capacity
AM	7:00 AM - 2:00 PM	7
AM	7:00 AM - 9:00 AM	5
AM	9:00 AM - 11:00 AM	5
AM	12:00 PM - 2:00 PM	4
PM	3:00 PM - 10:00 PM	7
PM	3:00 PM - 5:00 PM	5
PM	5:00 PM - 7:00 PM	5
PM	8:00 PM - 10:00 PM	4

As stores are responsible for orders placed within their assigned catchments, we were provided data by Save-On-Foods for the catchment postal codes that each store serves.

2.2 Synthetic Customer Data

Customer order data is generated synthetically for the initial development of the model. The generated solution is presented to Save-On-Foods's E-Commerce Data Science department for evaluation before evaluating the model's performance on historic customer order data.

For the provided Save-On-Foods store locations, we used Google Maps to query each catchment's postal codes and collected addresses of residential services such as daycares, schools, residential businesses, and housing addresses within the area. The use of residential service addresses provides an estimate of population density and demand within an area as they are often situated in community gathering locations and personal residences. We collected addresses up to the maximum delivery capacity for each store location. Delivery time frames are assigned to the collected addresses by permuting the list of addresses and assigning time frames for each address up to the time frame's maximum delivery capacity.

The synthetic customer address data is transformed into a time matrix using API calls from the Google Distance Matrix API in the Google Maps Platform[5]. This API produces metrics on the distance and time travelled for various modes of transportation which include walking, biking, driving, and transit. It uses historical traffic data to estimate the driving times and can estimate travel time at a specified point in the day. To ensure accurate road conditions and driving times, we use the past recorded traffic data for March 3, 2022, which factors in traffic accidents and road closures that occurred throughout the day.

To account for the service time at each location, we factor it into the time matrix as part of the driver's travel time. For each pair of locations in the time matrix that is not leaving from the depot, the service time is added. With this addition, the travel time to the first delivery location is its estimated travel time, and the travel time between every other pair of locations, including returning to the depot, comprises of the service time to deliver the order then the estimated travel time to the next location.

3 Mathematical Model

Two integer programming models that form the basis of the VRP for e-commerce grocery delivery (VRPEGD) are the VRP with Time Windows (VRPTW) and the Capacitated VRP (CVRP).

The VRPTW is an extension of the VRP where there is a fleet of vehicles departing from a centralized depot and each customer location in the network must be visited within a specific time interval before finally returning to the depot[6]. The problem assigns customers to a delivery vehicle and minimizes the cost of the arcs travelled subject to time constraints. The VRP is an NP-hard problem making it difficult to solve for large networks, thus heuristic solutions are used to find a feasible local minimum solution[7].

The VRPEGD and the VRPTW both aim to minimize the arc costs in the network given that each delivery location must be serviced within a specific time window. Service time requirements are accounted for as they are included in the time matrix. The deviation from the VRPTW comes with the vehicle capacities as each vehicle can only deliver up to its capacity. As well, driver breaks, driver load balancing, and the secondary objective of minimizing the number of vehicles are not included in the VRPTW formulation.

To address the vehicle capacity constraint, we consider the formulation of the CVRP model. The CVRP is also an extension of the VRP where each customer location has a demand corresponding to the number of items to be delivered and each delivery vehicle has a maximum capacity it can hold[8]. The problem objective is similar to the VRPTW, with it aiming to group customers together to a vehicle while minimizing the cost of the arcs travelled subject to the demand and capacity constraints. The CVRP is also NP-Hard and heuristic solutions are used to compute an optimal feasible solution.

Combining the VRPTW model formulation and the demand and vehicle capacity constraints from the CVRP model provides the Capacitated VRP with Time Windows (CVRPTW) formulation[9]. The objective is consistent with the VRPTW and the CVRP, with it minimizing the total arc costs of all vehicle routes subject to time window, capacity, and demand constraints. We use the CVRPTW as the base model for the VRPEGD and programmatically include features for driver breaks, load balancing, and minimizing the total number of drivers required. The concluding model is referred to as the multi-objective Capacitated VRP with Time

Windows, Load Balancing, and Breaks (CVRPTW-LBB).

3.1 Notation

The CVRPTW-LBB is formulated as an integer programming model on a complete directed network graph G = (L,E), where L is the set of nodes and E is the set of directed edges. The following notation is for the constants used within the formulation:

- D = Set of drivers for the shift
- L = Set of delivery locations including the depot
- $V_{i,\max}$ = Maximum order capacity for driver i, $i \in D$
- t_{ij} = Driving travel time from node i to node j, $(i, j) \in E$
- $O_i = \begin{cases} 1 & \text{Demand amount if location i is a delivery location} \\ 0 & \text{Demand amount if location i is the depot} \end{cases}$
- B = Allotted driver break time
- $[B_i, E_i]$ = Delivery Time window for node i, $i \in L$, $B_i \leq E_i$
- $\text{Penalty}_i = \begin{cases} 1 & \text{if node } i \text{ is dropped, } i \in L/\{\text{Depot}\} \\ 0 & \text{else} \end{cases}$
- M = Penalty cost for dropping a node

The decision variables for the formulation is as follows:

- $x_{ijd} = \begin{cases} 1 & \text{if vehicle d travels through arc } (i, j) \\ 0 & \text{else} \end{cases}$

3.2 Model Objective

The objective for the CVRPTW-LBB is to minimize the sum of the total amount of time each vehicle takes to complete their delivery route. The objective function is as follows:

$$min \sum_{d \in D} \sum_{i \in L} \sum_{j \in L, i \neq j} t_{ij} x_{ijd}$$

3.3 Constraints

The CVRPTW is used as the base formulation for the CVRPTW-LBB. We use the CVRPTW formulation as it supports routing multiple vehicles, delivery time windows at each location, vehicle supply, and delivery demands. Time windows are set to be $[B_i, E_i]$ for delivery location i, $i \in L$, the number of available delivery drivers is set to be $|D|$, and the number of delivery location is set to be $|L| - 1$.

The capacity of each vehicle i, $i \in D$, is set to be $V_{i,\max}$ and the demand at each location $l, l \in L$, is set to be O_l. This ensures that each vehicle is allowed to service several delivery locations up to its capacity and prevents overloading vehicles.

Driver breaks will also need to be considered in the CVRPTW-LBB formulation. There exists a one-hour gap between the delivery time frames in both the AM and PM shifts where no deliveries are completed. This one-hour gap allows the drivers to catch up on deliveries if they are behind schedule and drivers are expected to complete their 30-lunch break sometime within the hour-long gap. We assume that no deliveries can be completed within the hour gap and create a constraint to prevent deliveries within the time frame.

3.4 Programmatic Techniques

The vehicle load balancing feature and the secondary objective of minimizing the number of delivery drivers required to complete a delivery shift are included programmatically on top of the described CVRPTW-LBB model.

The load balancing feature is an optional parameter for situations where the total number of deliveries is significantly less than the sum of the vehicle capacities, $V_{total} = \sum_{i \in D} V_{i,max}$. When load balancing is applied, the total load of each vehicle is reduced by a factor of $f, 0 \leq f \leq 1$. The factor f is determined by evaluating the feasibility of the proposed model, with a lower factor suggesting a more balanced vehicle load and a more constrained model.

Deciding the minimum number of delivery drivers necessary is done by checking the feasibility of the CVRPTW-LBB solution for varying numbers of drivers. The lower bound of required drivers, D_{min}, is computed as $\lceil \frac{|L|-1}{\sum_{i \in D} V_{i,max}} \rceil$ and the upper bound is set as $|D|$. To determine the minimum number of required drivers, we compute solutions for the CVRPTW-LBB. We first set the number of drivers as D_{min} and iteratively increase it by one until it reaches $|D|$ or a feasible solution is found. Heuristic techniques are applied to find feasible solutions and up to 100 solutions are computed for each iteration. The minimum number of required delivery drivers is found when the first feasible solution is found. If no feasible solutions are found, the minimum number of drivers is set to be D and the optimization problem is considered to be infeasible. Penalties will be used to methodically remove infeasible customer delivery nodes.

4 Solution Techniques

We use an open-source solver and employ a guided local search meta-heuristic for the CVRPTW-LBB[10]. The solver computes solutions up to a specified solver limit and returns the best solution from the computed solutions. The solver limit varies by the size of the underlying network. Through iterative testing on the solution limit using varying network sizes and evaluating the performance of the computed results, a maximum solution limit by network size is determined. By increasing the solution limit, there exist minimal improvements with the objective function. Thus the solution limit is set as shown in Table 3.

Table 3. Solution Limit and Run Time By Total Deliveries

Total Deliveries	Solution Limit	Compute Time
21	500	< 30 seconds
42	500	< 30 seconds
84	1000	1 minute
126	1000	< 2 minutes

The solver was run using a Windows computer with an AMD Ryzen 5 2600 processor and 16 GB of ram. Details about solver run time can also be found in Table 3.

4.1 Additional Solver Considerations

In situations where the CVRPTW-LBB is infeasible due to its constraints, the vehicles must drop visits to some delivery locations. We employ the dropping visits feature built within the OR-Tools library to implement this functionality[10]. An additional penalty cost is introduced at all locations where when a location is dropped, then the penalty is added to the objective function. This penalty is set to be a very large number, such that the algorithm will search for routes that contain as many nodes as possible. Thus our objective function is transformed to be the following equation:

$$min \sum_{d\in D}\sum_{i\in L}\sum_{j\in L, i\neq j} t_{ij}x_{ijd} + M\sum_{i\in L}\text{Penalty}_i$$

The number of dropped nodes is considered the minimum number of customers that will receive their delivery outside of the selected time slot. The dropped nodes must be reintroduced to the solution either manually or programmatically.

5 Results

We implemented the CVRPTW-LBB model in Python using the OR-Tools library by Google and the Glop optimizer[10]. Routes were computed for 4 store locations for both the AM and PM shifts using historic customer data provided by Save-On-Foods. The store constraints and the CVRPTW-LBB model output is displayed in Table 4. All deliveries arrive within their respective time window and no nodes were dropped. The vehicles used and the load difference by Save-On-Foods's existing routing software are displayed in Table 5. Total travel time is not included in Table 5 due to limitations in the existing routing software with incorporating driver breaks and delivery service times within the total travel time. To have an accurate comparison, we instead compare the efficiency between models and the resources used by the proposed and the current in-force model.

Table 4. Travel Time, Resources Used, and Satisfied Deliveries by Shift

Store ID	Shift	Total Nodes	Load Difference	Max Vehicles	Vehicles Used	Total Travel Time
Store 1	AM	21	0	1	1	428 min
Store 1	PM	13	0	1	1	338 min
Store 2	AM	29	0	2	2	750 min
Store 2	PM	25	2	2	2	598 min
Store 3	AM	75	1	4	4	1622 min
Store 3	PM	56	2	4	3	1229 min
Store 4	AM	125	1	9	6	2586 min
Store 4	PM	70	1	7	4	1662 min

Table 5. Resources and Load Differences By Existing Routing Software

Store ID	Shift	Vehicles Used	Load Difference
Store 1	AM	1	0
Store 1	PM	1	0
Store 2	AM	2	4
Store 2	PM	2	2
Store 3	AM	4	1
Store 3	PM	4	4
Store 4	AM	6	2
Store 4	PM	4	8

6 Discussion and Conclusion

In this paper, we proposed the multi-objective CVRPTW-LBB integer programming model with programmatic features in the context of grocery e-commerce delivery for Save-On-Foods.

All solutions were generated in under 2 minutes using heuristic techniques and routes with less than 48 delivery locations were computed in less than 30 seconds. The proposed model's run time for a single delivery route improves on the existing delivery routing software's run time by 80% through the use of heuristic solutions. This is a significant reduction in computation time, which allows for employees of Save-On-Foods to evaluate the effectiveness of the computed routes at a more efficient pace.

No delivery locations were dropped from the proposed model and no additional manual adjustments were required for the outputted solution. This cuts down on the time spent by employees adjusting and re-optimizing the routes produced by the routing software as re-optimization is required for under half of the solutions. The time saved can be allocated towards other tasks within Save-On-Foods, providing better work efficiency within the E-Commerce department.

The load difference between vehicles in the same shift is greatly reduced and provides a more balanced workload distribution between the delivery drivers. For shifts with more than one driver, the vehicle load difference in the proposed model differs by 2 deliveries at maximum, with the routing software having load differences of up to 8 deliveries between drivers. The load balancing feature has aided in providing consistency between shifts and ensures that no drivers are provided with unfair preferential treatment.

The proposed model is developed using free open-source libraries with constraints tailored for Save-On-Foods. The performance of the proposed model greatly improves on the results and capabilities of the existing delivery routing software. Shifting from an industrial software to building an in house model significantly cuts down on software subscription costs and allows for capabilities to include features specific to Save-On-Foods.

The programmatic implementation of the secondary objective to reduce the total number of vehicles has slight improvements compared to the existing routing software. The number of vehicles used was able to be reduced by one for one of the eight computed routes; all other routes used the same number of vehicles as the routing software. In the context of Save-On-Foods's operations, with over 100 store locations participating in e-commerce grocery delivery and with each store having two delivery shifts per day, the marginal improvement in driver reduction accumulates and notably reduces staffing costs in the long run.

Finally, the proposed model's objective function aims to reduce the total time travelled by all delivery drivers within a shift. An efficient route reduces vehicle maintenance and gas consumption costs for Save-On-Foods while ensuring that all deliveries are completed within their respective time window and that drivers are provided with a full break duration. Maintaining accurate on-time deliveries positively affects repeated usage of the e-commerce platform. Ensuring a full driver break duration reduces driver burnout and provides better working conditions[11].

7 Future Improvements

The scope of the study focuses on proposing a robust model that optimizes delivery routing schedules within a well-defined area for a single store.

The model uses a time matrix that is assumed to be an accurate estimate. Future improvements in the accuracy of the proposed model can be completed by factoring in the delivery time windows within the distance matrix. As each location must be served within their respective time windows, the travel time can be computed using the traffic data within the time window. This will generate a more accurate travel time estimate and other additional work can be done to improve the estimate's accuracy.

A closely related problem to this study is the design of store delivery catchments. From the abundance of stores locations of Save-On-Foods, catchment design can improve the delivery service reliability, as some delivery locations may be easily accessible by a store location that does not serve the area. This potentially creates a more constrained network due to the increased travel times. Creating an optimized store catchment can improve the efficiency of delivery vehicles.

A further improvement related to the model computation time is the use of cloud and parallel computing. The models were run using personal computers with limited hardware capabilities. With over 100 store locations in Canada participating in e-commerce grocery delivery and with two separate shifts for each store, the computation power and solution computation time are important factors to the e-commerce business' success. Cloud computing can be introduced to run the model on a powerful cluster and parallel computing capabilities can be introduced in the computation of routing solutions. This would allow routes to be run simultaneously to reduce the overall computation time.

Finally, for infeasible networks where not all deliveries can be completed within their selected time frame, the proposed model drops delivery nodes and the user is provided with a file of dropped nodes. These nodes can be reintroduced manually back into the solution or can be re-optimized with a larger time window. Further work can be completed in the post-processing of the model solution to reintroduce the dropped nodes back into the solution such that the amount of manual work required to produce feasible routes is reduced.

References

1. Braekers, K., Ramaekers, K. & Nieuwenhuyse, I. The vehicle routing problem: State of the art classification and review. *Comput. Ind. Eng.* **99**, DOI: 10.1016/j.cie.2015.12.007 (2015).
2. Genly, B. Safety and job burnout: Understanding complex contributing factors. *Prof. Saf.* **61**, 45–49 (2016).
3. Biscaia, A. R., Rosa, M. J., e Sa, P. M. & Sarrico, C. S. Assessing customer satisfaction and loyalty in the retail sector. *Int. J. Qual. & Reliab. Manag.* (2017).
4. Inegbedion, H., Inegbedion, E., Peter, A. & Harry, L. Perception of workload balance and employee job satisfaction in work organisations. *Heliyon* **6**, e03160 (2020).
5. Google maps platform. https://developers.google.com/maps. Accessed: 2022-02-02.
6. Kallehauge, B., Larsen, J., Madsen, O. & Solomon, M. *Vehicle Routing Problem with Time Windows*, 67–98 (2006).
7. Laporte, G., Ropke, S. & Vidal, T. *Chapter 4: Heuristics for the Vehicle Routing Problem*, 87–116 (2014).
8. Borcinova, Z. Two models of the capacitated vehicle routing problem. *Croat. Oper. Res. Rev.* **8**, 463–469, DOI: 10.17535/crorr.2017.0029 (2017).
9. Carić, T., Galić, A., Fosin, J., Gold, H. & Reinholz, A. *A modelling and optimization framework for real-world vehicle routing problems* (IntechOpen, 2008).
10. Perron, L. & Furnon, V. OR-Tools. https://developers.google.com/optimization/. Version 7.2.
11. Söderlund, M. Employee display of burnout in the service encounter and its impact on customer satisfaction. *J. Retail. Consumer Serv.* **37**, 168–176, DOI: 10.1016/j.jretconser.2016.10.014 (2017).