Approach a VRP case with Google OR-Tools

Alex Tyler Leveque, Yuting Peng

Faculty advisor: Dr. Huijun Yi

May 14. 2020

Abstract: The classical vehicle routing problem (VRP) aims to find a set of routes at a minimal cost (finding the shortest path, minimizing the number of vehicles, etc.) beginning and ending the route at the depot. The study of the VRP has a considerable importance on the logistic systems. Many works on the VRP and its variants have been reported. In this study, we are not dealing with proposing a new algorithm to solve the VRP. Instead, we apply and practice an open source software, Google OR-Tools, for combinatorial optimization to obtain a feasible solution for the Hodges Warehouse and Logistics (Hodges) transportation data.

1. Industrial Problem and Background

Transportation occupies one-third of the amount in the logistics costs, and accordingly, transportation systems largely influence the performance of the logistics system. Nowadays, as the distribution networks have become more complex and customer expectations have been constantly growing, many businesses companies started searching for more effective ways to solve a route optimization to increase their operational efficiency, reduce operation costs, and promote service quality. A vehicle routing problem is a common challenge faced by many businesses. Our industrial partner, Hodges Warehouse and Logistics (Hodges), is an Alabama third party logistics, warehouse, and distribution provider aiming to provide superior services across the Southeast. As for its trucking business, Hodges mainly has their fleets of vehicles leaving a centralized location and picking up loads from dispersed locations and delivering these loads to respective destinations within a certain time constraint. An affiliate of the industrial research project, Hodges, as a distribution provider, picks up loads and delivers them to designed destinations for its clients. The data provided by Hodges to be analyzed for case study shows the scheduled routes for a specific day; the schedule indicates that the pickup and delivery locations are mostly in Alabama and some are in Georgia. There are total 15 pickup/delivery locations (Customer (A), Customer (B), Customer (C), Customer (D), Customer (E), Customer (F), Customer (G), Customer (H), Customer (I), Customer (J), Customer (K), Customer (L), Customer (M), Customer (N), and Customer (O)) in the schedule, and there are some identical routes. All the routes are assigned with a specific time window. The task is to design optimal routes from a depot to a set of destinations with major business-specific constraints such as pickups and deliveries, time windows and other resource limitations. To minimize costs, Hodges's primary interest is to complete the deliveries with the least number of drivers. Secondary interests include minimizing total travel distances of each vehicle. This work presents an adaptive approach for solving the real-world vehicle routing problems (VRPs) in the field of logistics.

2. Short Results

With the Hodges data, we characterized the research as a multiple Vehicle Routing Problem (mVRP) with two main constraints considered: pickup and delivery and time windows. In the study, we utilized Google Optimization Tools (OR-Tools): open-source toolkit for routing problems to deal with the specific Hodges case. Python is the programming language being used. The generated solution indicates that five optimized routes would be enough to cover all the delivery tasks.

3. Introduction to Methodology

The Vehicle Routing Problem (VRP) can be defined as a problem of finding the optimal routes of delivery or collection from one or several depots to a number of cities or customers, while satisfying some constraints. VRP is not a brand-new research. In the early literature VRP had originally been described as a generalized problem of Travelling Salesman Problem (TPS). With time, the VRP is categorized into three common types: VRP with Pick-Up and Delivery (VRPPD), VRP with Time Windows (VRPTW), and Capacitated VRP(VRPTW). We refer the literature C.-Y. Liong, et. al. (2008) to introduce the Mathematical models for VRPPD and VRPTW.

I. VRP with Pickups and deliveries (VRPPD)

The VRPPD arises when a number of goods need to be moved from certain pickup locations to other delivery locations. The goal is to find optimal routes for a fleet of vehicles to visit the pickup and drop-off locations. The figure 1 gives a visual view of the problem.

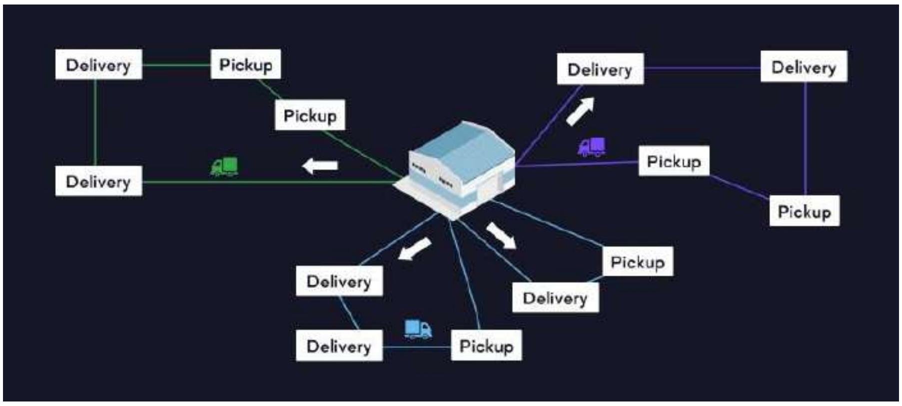

Figure 1 Diagram of VRPPD

The VRPPD can be formulated using the Generalized Assignment Procedure (GAP), which is used to find the minimum cost assignment of v vehicles to n clusters such that each vehicle is assigned to exactly one cluster, subject to its available capacity. The problem can be formulated as follows:

$V = \{1, 2, \ldots, v\}$ a set of vehicles

$N = \{1, 2, \dots, n\}$ a set of customers

C_n: the cost of assigning a vehicle to cluster n, for $n \in N$

u_n: the maximum load that will have to be carried in customer n

t_v: remaining capacity of each partially loaded vehicle v

$$X_{vn} = \begin{cases} 1 \; if \; vehicle \; v \; assigned \; to \; customer \; n \\ 0 \quad otherwise. \end{cases}$$

The mathematical formulation of the GAP is:

$Minimize \sum_v \sum_n C_n X_{vn}$

Subject to

$\sum_v X_{vn} = 1$, for $n \in N$,

$\sum_n u_n X_{vn} \leq t_v$, for $v \in V$,

$X_{vn} \in \{0,1\}$ for $n \in N$ and $v \in V$.

The first constraint ensures that each customer is assigned to exactly one vehicle while the second constraint ensures that the maximum load in a customer does not exceed the capacity of the vehicle assigned to that customer.

II. VRP with time window constraints (VRPTW)

We consider the variant of the VRP with time windows (VRPTW), where each customer must be visited within a specified time interval, called a time window. We consider the case of hard time windows where a vehicle must wait if it arrives before the customer is ready for service and it is not allowed to arrive late. The figure 2 shows the visual illustration of VRPTW.

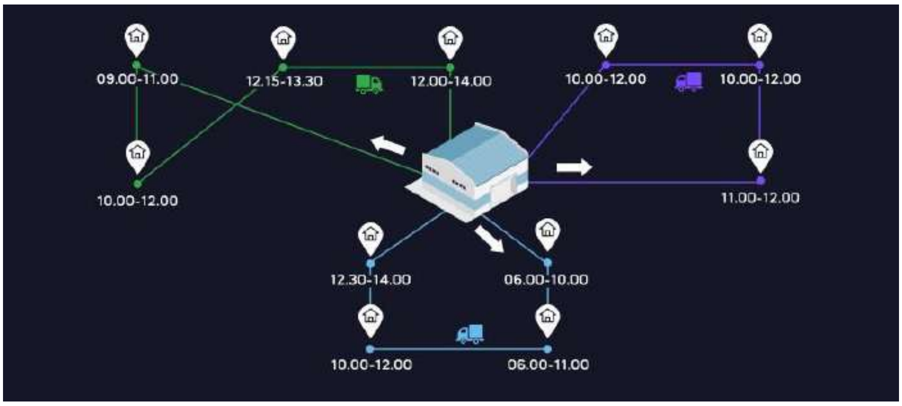

Figure 2 Diagram of VRPTW

Assume a single vehicle of capacity Q delivers goods from a depot to a set of customers $N = \{1,2, \dots, n\}$ in a complete directed graph with arc(i, j) in the set A that corresponds to possible connections between the customers. The distance d_{ij} and the travel time t_{ij} are associated with every arc(i, j). Each cluster $i \in N$ is characterized by a demand q_i, a dwell time s_i and a time window $[a_i, b_i]$, where a_i is the earliest time to begin service and b_i is the latest time. Accordingly, the vehicle must wait if it arrives at cluster i before time a_i. In a route $r \in K$, the optimal problem can be formulated as follows:

$Minimize\ \sum_r \sum_{(i,j)} d_{ij}\ X_{ij}^r$

Subject to

$\sum_{j \in N^+} X_{ij}^r = y_i^r$, where $X_{ij}^r = 1$ if arc(i, j) in route r, 0 otherwise;

$\sum_{r \in K} y_i^r = 1$, where $y_i^r = 1$ if cluster i in route r, 0 otherwise;

$\sum_{h \in N^+} X_{ih}^r - \sum_{j \in N^+} X_{hj}^r = 0$

$\sum_{i \in N^+} X_{oi}^r = 1$

$\sum_{i \in N^+} X_{i(n+1)}^r = 1$

$\sum_{i \in N} q_i\, y_i^r \leq Q$

$t_i^r + s_i + t_{ij} \leq t_j^r$, where t_i^r is the time of beginning of service at cluster i in route r.

4. VRP Computing Algorithm Review

Since the first VRP presented by Dantzig and Ramser in 1959 (Kallehauge 2006), many algorithms have been proposed for solving either the classical VRP or its variants. Genetic algorithm, greedy algorithm, and mixed integer programming are some decent algorithms to VRP.

I. Greedy algorithm

We refer the literature G. Lu, et. al. (2016) for greedy algorithm. Greedy algorithm always makes the choice which seems to be the best at that moment. On the one hand, considering only the instant situation while making decisions can save a lot of program running time, and it is easy to come up with a feasible solution this way too; however, assuming the locally optimized solution will lead to the globally optimized solution can actually omit the most efficient solution. For example, since we are trying to minimize the total travel distance in Hodges's case, the greedy algorithm will pick the closest location as the next destination for a trunk while scheduling the routes. However, the loads at the closest location will not be ready to be picked up until later in some cases, which will cause a long time window when the drivers has nothing to do but wait. Since we have multiple constraints and need to consider the gross cost, a greedy algorithm is not the best solution to our case.

II. Genetic algorithm

We refer the literature A. Kapoor (2019) for genetic algorithm. Genetic algorithm derives from the natural evolution principles. In order to come up with an optimal solution using genetic algorithm, we need to encode the problem variables into gene first, and then the algorithm will maintain a population of chromosomes and use the fitness function to find an optimal solution among multiple successive generation solutions. The advantages of this algorithm are that it can evaluate both continuous and discrete variables and it works well for large-scale optimization problems. However, designing a fitness function can be very fussy and disheartening. If we use genetic algorithm to solve our problem, then we need to encode our variable into a string of real numbers or a binary bit string. This algorithm will come up with some random combinations of routes and cross them over to get new generation of the solutions. The fitness function is used to test the efficiency and cost of each solution, and with the repetition of this procedure, we can get an optimal solution eventually. However, producing a practical fitness function for our case is complex and intimidating, and thus we did not choose to use this algorithm.

III. Mixed integer programming

We refer the literature H.Y. Kang and A.H.U. Lee (2018) and Google OR-Tools official website for mixed integer programming (MIP). MIP aims to find the optimization using objective function and the constraints for the variables of the function. The limitation of this programming is that it requires at least one of the variables only takes on integer values. While we are using Google OR-Tools to solve Hodges's case, MIP is actually a part of the behind-scene computing algorithm involved in the solution. When we are calculating the total distance and hour traveled by the trunks, we are setting boundaries such as the total time should not exceed 2000 minutes for each vehicle.

5. Problem Approach and Results Analysis

VRP constitutes one of the most challenging combinatorial optimization problems. A large number of different approaches have been developed over the years and a number of software packages are available on the market. Many fleet management software provides route optimization functions (e.g. Verizon Connect, Teletrac Navman, and Fleetio). In this study, for Hodges data, we use Google OR-Tools, an open source software for combinatorial optimization that seeks to sort the best solution out of a very large set of possible solutions. We refer the official website of Google OR-Tools for detail information. OR-Tools generally divided the optimization problems into eight categories: linear optimization, constraint optimization, integer optimization, routing, bin packing, network flows, assignment, and scheduling. For each category, Google OR-Tools has provided a set of matured programming code to solve a sample problem under that specific category.

All of the methods above are decent ways to get the optimizations for different type of problems. However, each one of these methods has its own disadvantage that make it not the best solution to solve our case. Hodge's case falls into the routing and constraint optimization categories among all that Google OR-Tools has provided. Under the routing category, Google OR-Tools has different subcategories such as traveling salesman problem, vehicle routing problem, capacity constraints, and etc. We combine the code for vehicle routing problem, pickups and deliveries, and time window constraints to generate a better program to solve our distinct case.

First, we use the distance between locations to generate a distance matrix and then turn it into an equivalent traveling time matrix. Then, we create an array that stores the information of each pickup and delivery locations related to our matrix, and another array to store the information of each time window. Google OR-Tools has well-developed programs that contain their own solver (function) to extract data from the matrix and search for feasible solutions that satisfy all the constraints. Then, all the solutions are compared to find the optimal one. The following is a snapshot a feasible solution that the Python program we have provides.

The output of our code indicates that five trunks would be enough to cover all the tasks. The total traveling time estimated for all the routes would be forty-six hours and forty-eight minutes; the schedules for each vehicle are as following:

1) vehicle 1, takes around nine hours and twenty-nine minutes
2) Vehicle 2, takes around eight hours and eleven minutes
3) Vehicle 3, takes around ten hours and thirty-three minutes
4) Vehicle 4, takes around eights hours and thirty-three minutes
5) Vehicle 5, takes around ten hours and two minutes

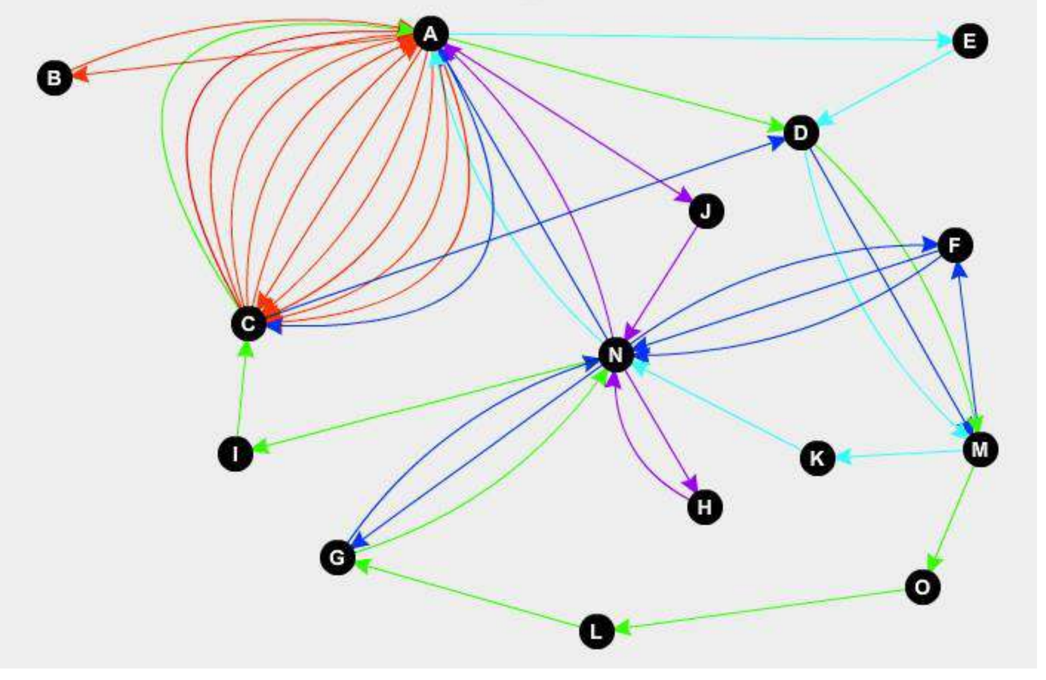

Figure 3 Diagram of assigned routes

Figure 3 represents a visual interpretation of these results, where nodes represent unique locations and directed edges represent traveled paths of trucks. Edges of the same color represent the path of a single unique truck. Note that the paths distinguished by the colors Red, Dark Blue, Green, Light Blue, and Purple represent the routes for vehicles 1, 2, 3, 4, and 5 respectively.

It is clear from the figure that many deliveries are made between locations A and C. Hodges initially had all these deliveries performed by a single truck. One of their initial questions was if it would be more efficient to have several trucks performing these deliveries, fitted between their other deliveries. From the graph, we can see that most of these deliveries are performed by one truck, represented by red edges; however, several deliveries are performed by other trucks, suggesting there is added efficiency in performing these deliveries inside other routes, confirming their originally hypothesis that it is efficient to use multiple trucks for this section of deliveries.

6. Conclusions and future research

Although this programming can give a feasible solution with pickups and delivers and time window constraints, more study needs to be done to add more constraints to this basic code. Every vehicle has a limited carrying capacity, meaning that it can carry only a certain number of items that all together should not exceed the threshold weight and volume. The placement of both pickup and delivery in a route depends on whether you have room in your vehicle for another pickup. It also dramatically affects whether you will meet all time window constraints. In addition to refining constraints, better estimating the drive times and costs of operations will provide more accurate results. This paper used primitive estimates for drive times; analyzing collected data for truck performance is suggested to improve efficiency. Thefuture study can be extended to design such a route that would allow a vehicle to pick up and/or deliver the maximum quantity at the lowest costs within the given capacity while satisfying the time windows and other related constraints.

7. Acknowledgements
We would like to give a special thanks to our industrial partner Hodges Warehouse and Logistics for providing the opportunity to work on this industrial problem. We appreciate the support of PIC Math, which is a program of the Mathematical Association of America
(MAA). Support for this MAA program is provided by the National Science Foundation
(NSF grant DMS-1722275) and the National Security Agency (NSA).
Reference
Emir Žunić , Dženana Đonko, and Emir Buza. (2020) An Adaptive Data-Driven Approach to Solve Real-World Vehicle Routing Problems in Logistics
Liong Choong Yeun, Wan Rosmanira Ismail, Khairuddin Omar & Mourad Zirour. (2008) Vehicle Routing Problem: Models and Solutions
Lu, G., Xiong, Y., Ding, C., & Wang, Y. (2016). An Optimal Schedule for Urban Road Network Repair Based on the Greedy Algorithm. PLoS ONE, 11(10), 1.
OR-Tools 7.2. Laurent Perron and Vincent Furnon.
https://developers.google.com/optimization/.
Kapoor Amita. (2019) Hands-On Artificial Intelligence for IoT.
Kang, He-Yau & Lee, Amy. (2018). An Enhanced Approach for the Multiple Vehicle