Optimizing Vehicle Routing

Rachel Cai[1] and Kai Zhu[2]
Henry M. Gunn High School[1], Software Engineer[2]

RESEARCH QUESTION

What is the most efficient way to route vehicles to maximize profit?

INTRODUCTION

Optimizing vehicle routing has many real world applications. For example, with large ride-sharing companies like Uber that rely on dispatch optimization for profit, it is important to find the best way to dispatch drivers to passengers while maintaining efficiency. This is an NP-hard problem in computer science. This project will study several algorithms implemented through computer programs that will mimic a real-time taxi operation. By running different algorithms through computer simulations, it is possible to evaluate the business benefit and running cost of each algorithm under different system loads.

BACKGROUND AND SIGNIFICANCE

Vehicle routing optimization can be used in ride sharing, taxi dispatch, package delivery. Many papers have been published on this topic, such as studies on ride-sharing, dial-a-ride, and route optimization (Lin). Uber has released blog posts on how they have been using Artificial Intelligence to help understand how to optimize their algorithms. However, there still is not an optimal solution to the vehicle dispatching problem. The closest they can get is attempting to make it more "economically viable without having to cut down on service for the customers" (Hansen). Because the issue is so complex, there is no polynomial algorithm that can directly solve this problem. In other words, the most optimal answer will most likely be exponential and computationally expensive. However, this approach to the problem is unique and different from work that has previously been done. Whereas previous attempts have been focused on finding the one most optimal solution, this research is focused on studying the behavior of different algorithms and finding a close-to-optimal solution while keep the system running time under the polynomial constraint to save the cost.

RESEARCH METHODOLOGIES

After developing the algorithms, each algorithm was run through a simulation of a real-life taxi operation. In each simulation, the algorithm would be given an input file with the road map, the passengers' call times, destinations, and the taxi locations.

Figure 1: Sample Input

Passenger	Pickup Site	Dropoff Site	Call-in Time	Dropoff Time
Alex	A	D	8:00	8:12
Bob	C	D	8:01	8:40
Callie	A	B	8:02	8:30

Figure 1: This is an example of a sample list of passengers and information about them that would be provided in an input file.

The first algorithm is a Greedy algorithm that sends the closest car at that moment to each passenger.

Figure 2: Pure Greedy

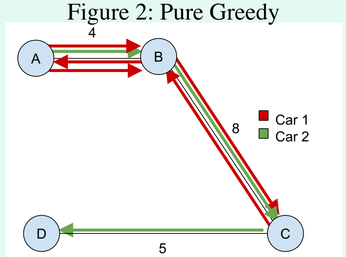

Figure 2: Pure Greedy algorithm being run through the example simulation.

The second algorithm: "Crystal Ball" can see future events and calculate the most optimal combination.

Figure 3: Crystal Ball

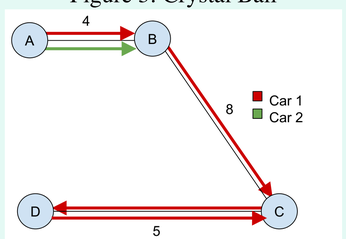

Figure 3: Crystal Ball algorithm being run through the example simulation.

The third algorithm "Slacker" procrastinates until the last minute to dispatch the taxi, in hopes that a better deal will appear.

Figure 4: Slacker

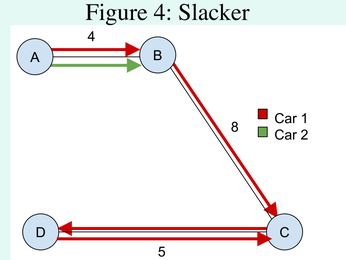

Figure 4: Slacker algorithm being run through the example simulation.

DATA ANALYSIS AND RESULTS

Table 5: Profit/Time Comparison Between Algorithms

Algorithm	Average Revenue	Average Expense	Average Profit	Time Complexity
Pure Greedy	520	414	106	P * C
Crystal Ball	520	380	140	P^C * P^C
Slacker	520	392	128	P * C * P

Table 5: This shows the business profit/loss for the above example under three algorithms.
P stands for the number of passengers and C represents the number of cars.

As seen in the table, the Pure Greedy Algorithm made the least profit. This is because it can only see and optimize for the current customer, leading to extra idle driving. Crystal Ball made the most profit, because it improves by seeing into the future and can figure out the one best way to dispatch cars. However, Crystal Ball is an exhaustive search and cycles through all the possible combinations, so, as seen in the table, it will require an exponential cost. Because it takes so long, it will be slow and ineffective for a large scale business that has to deal with hundreds of passengers at the same time. Slacker is a compromise between the simple Pure Greedy algorithm and the expensive Crystal Ball algorithm, and achieves a close to optimal result while managing to run under a polynomial time. The difference between the three algorithms will be magnified on a much larger scale with more passengers and vehicles.

The current work makes a few simplifications, such as assuming no traffic in the road system and no carpooling. In the future work, we will study the system behavior under varying traffic, add car-pooling and ride-sharing, and use historical data to predict future requests.

ACKNOWLEDGEMENTS / REFERENCES

Special thanks to my mentor, Kai Zhu, and my AAR teacher Ms. Durquet for helping make this project possible.

Works Cited:

Hansen, Jesper Bjerring. "The Dial-a-Ride Problem and Exploiting Knowledge about Future Customer Requests." *The Dial-a-Ride Problem* (2010): n. pag. 2010. Web. 7 Nov. 2016.

Kuo, Marc. "Taxi Dispatch Algorithms: Why Route Optimization Reigns."*Routific*. N.p., 24 Mar. 2016. Web. 26 Sept. 2016.

Lin, Yeqian, Wenquan Li, Feng Qiu, and He Xu. "Research on Optimization of Vehicle Routing Problem for Ride-sharing Taxi." *Research on Optimization of Vehicle Routing Problem for Ride-sharing Taxi*. Science Direct, 30 May 2012. Web. 08 Nov. 2016.