TJHSST Computer Systems Lab Senior Research Project
Solving the Vehicle Routing Problem with Multiple Multi-Capacity Vehicles
2008-2009

Michael Sanders

October 28, 2008

Abstract

The Vehicle Routing Problem (VRP) has existed as long as a distributor has needed to deliver items. As such, the VRP has been solved with many different methods, including agent architecture and ant colony modeling. However, these methods have generally been set up for an established organization that has a specific number of vehicles with only a few unique capacities. This project aims to create a program that will solve the VRP, but in a case where all the vehicles could have different capacities. This is the situation faced by some volunteer groups that do not have established vehicle fleets and rely on people volunteering vehicles when something needs to be distributed.

Keywords: A* search, vehicle routing problem - List any special vocabulary words that will apply to this research area.

1 Introduction

1.1 Scope of Study

There are two portions for this project. The first involving creating a program to find the quickest route between two locations. This program will take into account both the time required to traverse a road and the distance traversed, as well as allot some time for stops. The second portion will map all the delivery points and find the most effecient grouping of delivery points to create the most efficient route. Knowledge required includes finding an efficient heuristic for both the route finder and the route selector.

1.2 Expected results

The expected result is a program, when given a list of delivery points, amount needed to be delivered to each point, and a list of vehicles with given capacities, returns a list of routes that, when totaled, result in the most efficient solution. The program will be capable of handling not only a multi-capacity multi-vehicle routing problem, but also the original VRP. The program should also not take much work to extend to the multi-depot VRP (MDVRP). The project may yield insight into new heuristics to determine the optimal routes when the program is assigning customers to routes and routes to vehicles.

1.3 Type of research

This project is almost pure applied research. While it is almost entirely designed solely to assist a volunteer group, some insight into heuristics may be gained.

2 Background and review of current literature and research

The VRP has been looked into extensively, including the variants such as the MDVRP and the VRP with Time Windows (VRPTW). Ant colony simulation and mobile sensor networks have been used to solve this problem and its variants. Mennell, Shmygelska, and Thangiah did a project to evaluate using

agents to solve the VRP. The program used agents to represent the vehicles and "auctioneers" that informed the vehicles of the current situation with regards to customers and deliveries. While the resulting program did not find the optimal solutions for any the sample problems, it was extremely adaptable for the MDVRP and VRP with multi-capacity vehicles. This project has given me some new considerations into how to write a route creator.

3 Procedures and Methodology

The first step is to create a program that, given a list of roads and intersections, can quickly find the quickest route between two points. The next step will be to create a program that creates routes by reading in the list of delivery points and number and capacity of vehicles and utilizing the previously-created route finder. The method of testing the final project will be evaluating the results of the program against a human-generated list of routes. While "most efficient" has not been defined, it will probably be measured by amount of product over distance traveled. The goal would be the largest number, although this result could possibly be very time-costly. The programming will be done in Ruby. The data needed is a list of all the roads that a delivery point could be on or could be required to get to a delivery point and a list of delivery points with amount needed to be delivered.

It may be possible at the end to have the program print out a list of routes in a format that a program such as ArcView could turn in to a graphic representation of routes.

The only method to test the final project will be to compare it against what a human could do. The program's result in a form such as product delivered over distance traveled would be compared against a human's result in the same form. Also factored in would be the time required for the program and the human to create their results.

Testing would involve using old databases of delivery points and amounts to be delivered. To test each portion of the project, inputs with known best results would be input and the result compared against the best known. The route finder can be compared against commercially available programs online. The route optimizer can be tested with data that would have intuitive best results (such as three delivery points right next to each other) and evaluating the results.

Requirements: Road data for the locality of deliveries-needs to have inter-

section data and geographic coordinates List of available vehicles and their capacities-capacities needs to be standardized, i.e. number of specified objects that vehicle can carry List of delivery points

You could describe particular algorithms you'll be using and learning about.

4 Expected Results

The program should be usable by a volunteer group or similar organization that needs to distribute items, but does not maintain a fleet of standardized vehicles. The program should, in a reasonable amount of time, provide a list of the most efficient routes with the best vehicle for that route. Efficiency will most likely mean either distance or time is minimized. Graphically, routes can be posted on a map. Routes can be displayed by statistics such as distance traveled, time required, and amount of product delivered. Future researchers can extend this problem to the case where the number and capacity of vehicles is not known ahead of time, which means the program will have to actively recompute the solution each time a new vehicle is added to the pool, which could be after a vehicle has left.