Warehouse Location Problem in Raleigh, NC

OR501 Fall 2022 Project Submission

Keenan Smith

12/3/22

Abstract

This project uses Dijkstra's algorithm and Linear programming to find the optimal location for five warehouses to service 1000 customers in the Raleigh area. The model is run twice to prove that the logic of the warehouse selection problem fits this problem statement. The problem takes as input an Openstreetmaps feature set of highways, transforms it into a graph network, selects 1000 random nodes as customers, selects 5 random nodes as warehouses, runs a linear optimization model and then outputs the road network showing that each customer is served by one warehouse located in the optimal location.

1 Introduction

When analyzing where to place infrastructure there is often much thought put into the location and the cost associated with that location. Today, we live in a society of online delivery and fulfillment centers. One of, if not the largest, distributor is Amazon, which has no department stores but purely fulfills customer orders online within 2 days if you have Amazon Prime or within a few hours, to a week or so depending on your location. I have watched numerous videos on YouTube about logistics and have always found the subject to fill me with awe.

When asked to do a linear programming / Operations Research (OR) Project for OR501 at North Carolina State University (NCSU), I thought about what I had learned in the class and then tried to balance it with my interests. I am interested in maps, history, and systems. Road networks are a giant system. They are the arteries of our modern day. It is how goods get from their intermediate point to their end point. With that being said, I decided to focus on a combination of Network Analysis and Dijkstra's Algorithm for shortest paths, with the Warehouse Location problem.

What follows is an analysis of the Raleigh road network and choosing a location for 5 warehouses to serve 1000 customers. All my code is written in R and can be adapted for more customers or more warehouses, though the code for the warehouses would need to be updated and refactored to ensure it can scale for growth. This is purely written as a university project and not written as a production ready product to go out into the world, though I would like to explore this model further in future projects.

2 Problem Description

This problem has two distinct programming challenges. The first is getting the street network for Raleigh, transforming it into a graph network, and then calculating the distances. The second is selecting the 1000 customer locations, the 5 possible warehouse locations and the linear program

that calculates the minimum cost. There is a third programming challenge which is getting the visualization of the network once warehouse problem has been optimized.

2.1 Network Program

My first challenge was to secure a usable map that I could use as a source. For this, I used the tutorial from *Spatial Networks with R with sf and tidygraph* (Meer, Lovelace, and Abad (2019)). This utilizes the sf and tidygraph libraries in the R ecosystem to bound a box for a location, in this instance Raleigh, NC, and then transform it into a graph network with edges and nodes from which to calculate the rest of the problem.

The programming is located within the Appendix, but to give a brief over view of the method used. I first pulled the bounding box for Raleigh, NC. I then pulled the highway data from the openstreetmaps API (Padgham et al. (2017)). I made sure only to use roadway networks. The API for highway will also pull pedestrian and cycling "highways" but that is not the network I required for the road traffic heavy United States. I cleaned the data of duplicates as well as other GIS specific issues by using the GRASS API (Bivand (2022)). I utilized the function developed in Meer, Lovelace, and Abad (2019) to transform the Raleigh road network into a network graph.

It was at this point that I had to solve the first of the techniques learned in OR501. I needed to develop a matrix of all the shortest paths between every single node. Fortunately, R uses a well known graph library known as igraph which has function called `distances()` that will do that using Dijkstra's algorithm. Since there were about 63k nodes in the graph, this was quite computationally heavy and ended up producing a matrix with a disk size of 9.2GB![1] This was the final step prior to solving the linear program.

[1] A point for optimization in the programming later would be to hold this matrix in a SQL database where only the distances required would be needed for the future calculations.

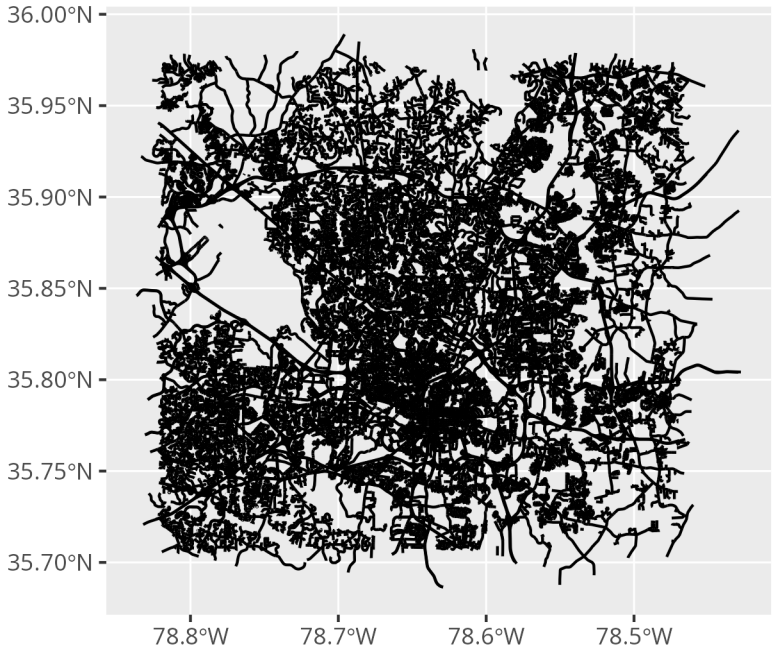

Figure 1: Raleigh City Streets

2.2 The Warehouse Location Problem

The warehouse location problem is a fairly established linear programming optimization problem. The crux of the version that I selected is that there are a set of customers and a set of possible warehouse locations. The objective of the function is to find the minimum total travel cost at the minimum number of warehouses. The linear program is denoted below and the Latex was sourced from Schumacher (2017).

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{n} \sum_{j=1}^{m} transportcost_{i,j} \cdot x_{i,j} + \sum_{j=1}^{m} fixedcost_j \cdot y_j \\
\text{subject to} \quad & \sum_{j=1}^{m} x_{i,j} = 1 && i = 1, \ldots, n \\
& x_{i,j} \leq y_j, && i = 1, \ldots, n \quad j = 1, \ldots, m \\
& x_{i,j} \in \{0, 1\} && i = 1, \ldots, n, \quad j = 1, \ldots, m \\
& y_j \in \{0, 1\} && j = 1, \ldots, m
\end{aligned}
$$

This is the part of the project that I really enjoyed. I had done all of this work to get the road network into a graph and then had a matrix with all of the shortest paths between each of the 63k+ nodes. Instead of arbitrarily selecting nodes, I decided to select a random sample of 1000 nodes for the customers, then I selected a random sample of 5 nodes for the possible warehouses. I utilized the code from Schumacher (2017) to make sure that I had the linear program in a nice space for calculation.

2.2.1 Assumptions

I found that it costs a UPS truck $0.422 per mile (@LammertWalkowicz12) and that the average cost of storage in a warehouse in the US in 2022 is $7.96 per sqaure foot (WNF (2022)). I made sure that the code multiplied the distance by 0.422 in the transport cost equation and that the fixed cost was randomly distributed around a normal distribution with 7.96 as the mean.[2] I did not factor in rent or locality of rent cost as the randomness of the Warehouse selection would have made this task computationally expensive given the scope.

The other assumptions of the warehouse problem is that every customer must be assigned a warehouse and that if a customer is assigned a warehouse, the warehouse must be built. This can be run as an integer problem, but since Schumacher (2022) has a category for binary, I choose that option when initializing the variables in the model since a customer is either 1, if they are assigned to that warehouse, or 0, if not.

Lastly, there are nodes that cannot be reached within the network despite my best efforts with creating the graph. I could have located these nodes within the graph and eliminated their selection from the random sample, but I did not. I did adjust the cost to be $1 * 10^{12}$ though when I recap the results in the Conclusion, I do believe that these node issues did result in a higher minimum cost than if I would have ensured they were not selected in the random sample. I also did not bar the ability for a customer and location to hold the same node. As these are street ends or street intersections, this is not outside the bounds of reality. There could very well be a

[2] I could have also changed the standard deviation a bit, but choose not to since this is just the fixed cost and the cost of transportation in such a large network was going to overshadow the fixed cost by quite a bit.

customer that is also an employee of the warehouse. Though a future interation of this project could take this into account in the programming.

3 Numerical Study

Due to the nature of the set up of this problem, it lends itself perfectly for numerical study. I made sure that the nodes were all selectable in the random sampling. This means that given a certain random seed, this study could be reproduced several times. In this project, I decided only to randomly select 1000 customers and randomly select 5 warehouse locations. The code could scaled currently as it sits to incorporate more customers. There is not a limitation on the max amount of customers that can be selected. The model can also incorporate less customers. However, for the visualizations of the shortest paths from the warehouse to the customer would have to be refactored for adding or subtracting the amount of possible warehouses.

Another possibility for this model is that an inspection of the individual node locations could select already existing warehouses and then select the general location of an companies employees within the bounding box of the Raleigh area. I chose not to do this in this instance as I was more curious about the models ability to operate with a random selection of customers and warehouses and how it would allocate the customers to possible warehouses based on the shortest paths.

I can see various ways that this model could grow in scope. One thing that the Padgham et al. (2017) API operates is that it includes street names and also incorporates building points. This could lead to a future iteration of the project that allows the user to upload a csv of customer locations and possible warehouse locations into the program and then the optimal location could be selected. The next step in terms of linear programming and optimization would be for the given individual warehouse, what is the optimal low cost route from the warehouse to each of the nodes using the travelling salesman problem (wikipedia (2022)). I believed that this step was outside the scope of the project given time and resources I could devote to the project.

4 Conclusion

In the end, I ran the linear program two times. Each time, I randomly selected 1000 customers and 5 possible warehouse locations from the Raleigh road network graph.

4.1 First Optimization Model

The first objective value given the 6000 constraint was $4.642 * 10^{12}$. I believe this is a result of some nodes being selected in the customer base that could not be reached by any warehouse and selecting the $1 * 10^{12}$ default value for these problem cases. I do not think this a fault of the program, I told the program to do this as a quick solution to a problem with the solution not optimizing.[3]

I have already discussed possible solutions to this issue by excluding these nodes from the selection pool. I am also sure that there are clever ways to write the code to tell the program to select a different warehouse more explicitly. Unfortunately, due to time constraints, I decided against pursuing these options and instead focused on the end result and developing a usable result that was within the project scope. Below is a figure of the final result of the first linear programming model result.

This first model shows that two central locations can really reach large majorities of the city but that the cost of having 5 warehouses is still cheaper than consolidating down into less warehouses given the constraints of this model in particular. It was striking that the cost benefit of having one warehouse services one customer was so great that it incorporated an entire warehouse for it. Obviously this is not an objective result in the real world with stakeholders most likely incorporating this location into the blue network. It is an interesting discovery however.

I am extremely proud and excited for this solution. Each customer is accounted for and a shortest path from the customer (in yellow) to the warehouse (in black) is plotted. From this image, you can really analyze the application of the warehouse selection problem in a real-world application

[3] This was because the default value was infinity.

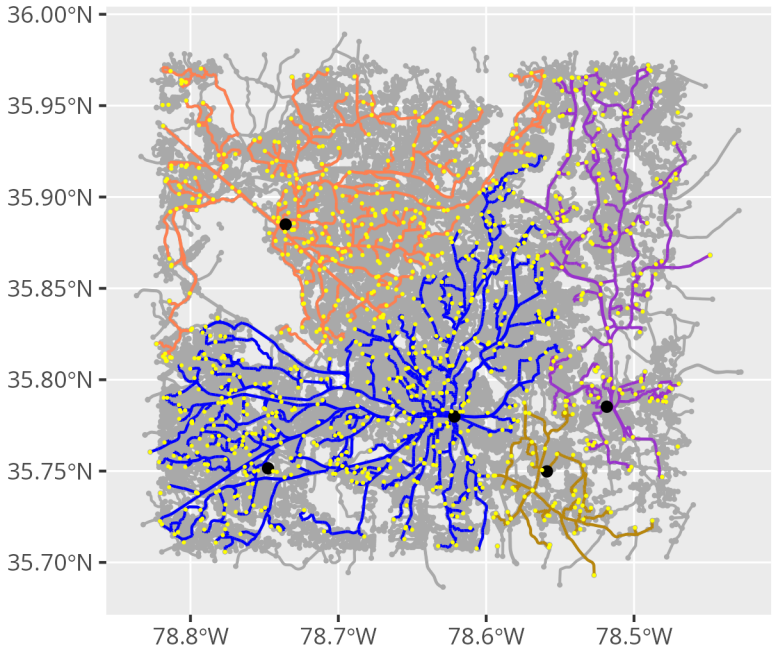

Figure 2: First Warehouse Location Model Solution

and space. A lot of my coding effort was dedicated to achieving this visual representaion as I felt like it really added a lot to the project in terms of tangible results that could be shown to colleagues and others that are interested in the benefits of linear programing and optimization.

4.2 Second Optimization Model

The second optimization model had an objective value of $5.064*10^{12}$. This further proved to me that I need to go back into the code and fix the issue with nodes being selected that cannot be reached by any warehouse. However, to focus on the result. This shows that a different random selection of customers and possible warehouse locations yields a different result entirely. This result is 4.22 greater than the first model. This shows that there are model configurations that are much greater or much less than others.

As you can see from the second solution, this yields a very different solution to the problem. There are longer and more sprawling networks. There are no outliers in terms of one warehouse only serving a single or couple of customers, like in the first model. Even still, this model yields

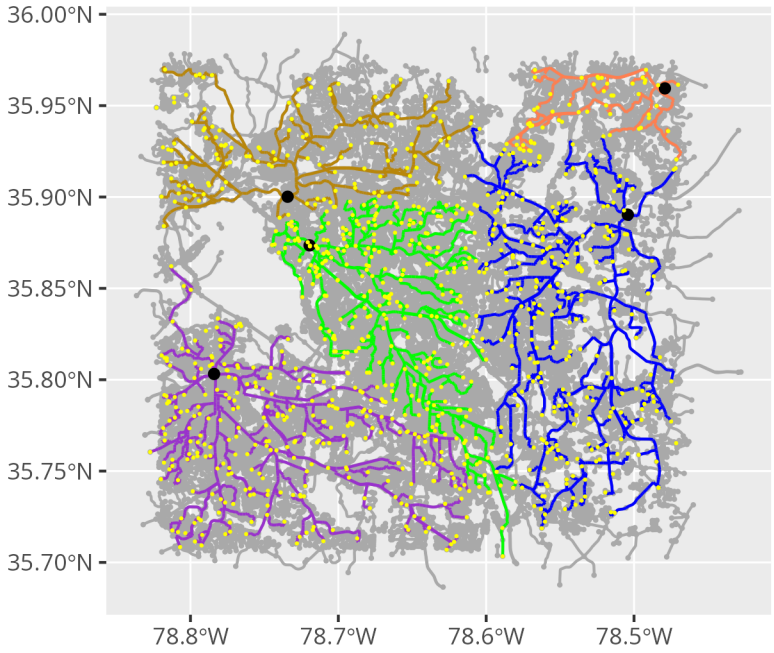

Figure 3: Second Warehouse Location Model Solution

a higher objective value so there are aspects to it that require more driving cost.

4.3 Going Forward

I believe this is a great demonstration on its own of the benefits of linear programming and the warehouse location problem. I think that it shows truly how it can be applied to our real world in a place that we live and go to school. That being said, there are a few things I would like to do going forward to advance the model.

- Fix the selection of isolated nodes. These are an issue that leads to large objective values that should be addressed in the next version of the program.
- Locate the matrix of all distances in an SQLite database so that 9.2GB does not have to reside within the RAM of the program for each run.
- Be able to easily select a list of variables for customers and possible warehouse locations for potential stakeholders to examine.

- Run a monte carlo simulation where the program runs 1000s of times with randomly selected warehouse locations and randomly selected customer locations to see if their are optimum warehouse location nodes within the Raleigh road network that provides an overall optimum solution to this particular problem set.

5 References

Bivand, Roger. 2022. *Rgrass: Interface Between 'GRASS' Geographical Information System and 'r'*.

Meer, Lucas van der, Robin Lovelace, and Lorena Abad. 2019. "Spatial Networks in r with Sf and Tidygraph." 2019. https://r-spatial.org/r/2019/09/26/spatial-networks.html.

Padgham, Mark, Bob Rudis, Robin Lovelace, and Maëlle Salmon. 2017. "Osmdata." *The Journal of Open Source Software* 2 (14). https://doi.org/10.21105/joss.00305.

Schumacher, Dirk. 2017. "The Warehouse Location Problem." 2017. https://dirkschumacher.github.io/ompr/articles/problem-warehouse-location.html.

———. 2022. *Ompr: Model and Solve Mixed Integer Linear Programs*. https://github.com/dirkschumacher/ompr.

wikipedia. 2022. "Traveling Salesman Problem." 2022. https://en.wikipedia.org/wiki/Travelling_salesman_problem.

WNF. 2022. "Warehousing Services Costs, Pricing, Rates and Fees." 2022. https://www.warehousingandfulfillment.com/resources/warehousing-services-costs-pricing-rates-and-fees/.