MULTI-LEVEL ROUTE-OPTIMIZATION COMPUTER APPLICATION

by

RYAN SUTARDJI & FRANK NGUYEN

A Senior Project submitted

in partial fulfillment

of the requirements for the degree of

Bachelor of Science in Industrial Engineering

California Polytechnic State University

San Luis Obispo

Graded by: Tao Yang__________ Date of Submission: ___________

Checked by: __________________ Approved by: _________________

Table of Contents

Abstract

This report provides a detailed analysis on how to optimize driving routes by creating a computer application. There are many different route-optimization issues that logistical companies consistently face, as well as many different solutions and algorithms. With technology on the rise, pick-up, delivery, and transportation services are become a huge part of our everyday lives. When optimizing routes, reducing transportation costs by minimizing travel distance is always ideal, but other factors must be considered such as arriving at a location before or after a certain time. Our objective is to optimize driving routes based on travel distance and priorities. We approached this project by using SQL as our main source of determining the route order based on the given distances and priorities of each destination in the projected route. We also used VBA as a tool to support the calculations and ASP.net to insert Javascript and HTML code, which allows us to visualize the Google Maps route once the order has been determined.

Acknowledgements

We would like to thank Dr. Yang for helping and guiding us throughout the project these past two quarters. We would also like to thank Computer Engineering student Timothy Acorda and alumni Gregg Hanano for helping us get a better understanding of Javascript, which was an essential part of our application.

Introduction

This project seeks to design a route-optimization computer application using a combination of Microsoft Access and ASP.NET. This application will take in addresses, including city, state, and zip codes along with a weighted priority associated with each address, as inputs, and will show an optimized route as the output.

This project is being done because while many different algorithms have been created for delivery and transportation services, none have been able to accommodate for specific requests to arrive at different locations at a certain timeframe or window. We will be developing and analyzing our model based off of a current algorithm, and making changes to specify the model to our scope, and then create a prototype of our computer application using a pre-populated dataset that will confirm that our methodology and code works. A mix of Visual Basic, SQL, Javascript, and HTML code was used to complete this application. Before coding, a flow process map will be made to get a full understanding of how our code will be programmatically initiated. We believe this project is significant because many different companies are being brought to us today that use delivery and transportation services to solve many different problems.

Creating this algorithm to be as flexible as possible will be one of the main challenges. There are a lot of different factors that can be put into play as we move forward with this project. If a special request is made that doesn't follow our algorithm's rules, then we have to be able to override the algorithm in the easiest possible way. Other factors like this will be discussed in our analysis and how we overcome these issues.

The main content of our report will consist of a detailed explanation on why we chose our method, as well as our test results and analysis.

<u>**Background**</u>

Taxi services such as Uber and Lyft have grown exponentially throughout the past few years, bringing a new era of transportation services. The objective of the service is to get people from point A to point B. People have also shown interest in food services such as DoorDash and CafeRunner to get food or coffee delivered to them based on the restaurants/cafes in the customer's respective area. Although taxi services and food services already exist, they are all focused on a single goal. Businesses in the past few years have always only focused on focusing on one start and one end location. With the amount of single-dimensional services, there is a tremendous need for a multi-dimensional services to save transportation costs.

FedEx is an example of a business that focuses on having multiple destination points in a single route. A problem that many of us face is the fact that delivery time is out of the customer's control. Fed-Ex's mission is to get the package to you as fast as possible, so optimizing their route based solely on location makes sense. An example of a priority-based delivery service is a company called Doorman. The mission of this company is to provide the type of service for you that FedEx doesn't provide, a package delivery service where you can specify the time to have your package brought to you. This type of service revolves around not just generating a route based on location, but based on priority. With packages ordered through Doorman being delivered to a central location, delivery drivers all have a singular starting point. Another example of a business with even more complex priority systems is Instacart. This company is a startup that will buy your groceries and deliver them to you at certain time window requested by the customer. This adds even more layers to the priority system because there is always a predecessor to the delivery location – the grocery store. Instacart uses a combination of operations research and big data to determine the timeframe that they must buy the groceries and when they must be delivered to the customer’s doorstep. The problem that Instacart faces is that they still follow the Uber/Lyft model, where one driver is only in charge of picking up and delivering groceries to one customer at a time. Although Instacart and Doorman both provide better customer service by catering to customer requests, it comes at a cost.

Literature Review

Transportation Planning

The key to creating the most efficient route on a consistent basis is creating the right model and algorithm. Transportation Planning (TP), which was formulated by Hitchcock, is the method of finding optimal routes from suppliers to the customer. In Shinichro Ataka and Mitsuo Gen’s article “Solution Method for Multi-Product Two-Stage Logistics Network with constraints on Delivery Route”, this method was deemed to be unfeasible for three reasons:

1. Conventional TP models consider only one-stage routing from distribution centers to customers
2. All products are assumed to be the same type
3. Only the delivery cost is considered for every delivery section

E-Constraint Method

Although our overall goal would be creating the most consistent and efficient route, we need to keep in mind other aspects of our service such as routing cost and consistency. Attilla A. Kovac’s article “ The multi-objective generalized consistent vehicle routing problem”, proposed a multi-objective optimization approach to a vehicle routing problem, which is a method to most effectively satisfy multiple goals or criteria. This approach allows one to create a thorough trade-off analysis between any conflicting goals they may have. Their algorithm is based on an e-constraint method, which is where the focus turns to optimizing one goal, while restricting the other goals to a defined constraint. This allows one to fully optimize a specific goal without affecting the optimal level of other goals. The outcome of this showed that more often than not, arrival time and driver consistency can be improved simultaneously and increasing travel cost by four percent at most improved arrival time consistency by seventy percent. These results can help us specify whether or not our routing algorithm is the most efficient based on multiple constraints.

Light/Heavy Resources

In C.K.Y. Lin’s article, “A vehicle routing problem with pickup and delivery time windows, and coordination of transportable resources”, two different models are created that depend on the pickup or delivery of light and heavy resources. The deliveries of heavy resources were prioritized, with light resources subordinating to the rest of the operations. This means that the routes revolved around transporting heavy resources, then assigning light resource pickups and deliveries within the route. Whether it’s due to time windows, or max/min transportation time constraints, prioritization is a key aspect when determining the correct order of pickups and deliveries. We will be using a Microsoft Access lookup table to create priorities and use them to determine the order of destinations.

Clustering

A supporting method that can be used for problem size reduction is "clustering locations of similar pickup time windows" (Lin, 2011). This is a very useful solution when a larger radius is covered for pickups and deliveries, so clustering locations is highly taken into consideration. With food pickup and delivery services, drivers are only required to cover a smaller radius, usually sticking to one city at a time. In our prototype example, we will only be covering the east bay; so clustering won't be too necessary since each cluster will only cover a small area. Although, if arrivals were to span all the way out to north or south bay, then clustering will be necessary.

Genetic Approach

A genetic approach is used to accommodate a two-stage distribution model, where resources are transported from the manufacturing plant to the distribution center (DC), then are transported from the DC to the customer (Ataka & Gen, 2009). This approach only delivers to the customers, while Lin's approach will pick up resources from one customer and delivery to another. This model creates a few more problems than a traditional pickup and delivery system. With this two-stage model, the effects of storage are taken into consideration.

Constraints

There are many different algorithms when determining the most efficient route, but the formulation always revolves around minimizing time and transportation costs. The constraints in the formulation are what make one algorithm different from another. Lin's formulation minimizes both fixed and delivery costs and consists of constraints that include visiting each node exactly once and having a maximum capacity of resources that can be transported at once. The genetic approach has similar constraints, but as mentioned, has a constraint with having a maximum capacity of resources in each distribution center.

Business Model

Nitin Nayak and Anil Nigam wrote in their article, "Modeling Business Services for Implementing on Global Business Services Delivery Platforms" that when creating a service-oriented business, it is important for the business to integrate with other businesses and processes. Having partners is key to generating more revenue. Companies such as DoorDash and Café Runner partner with many fast food chains and restaurants where they get a portion of the total order payment. Generating revenue solely on service charges won't allow the company to live for a long time. Taking a portion of restaurant revenue as well as a portion of grocery revenue can generate a good amount of profit. A survey can be conducted to determine which grocery stores and restaurants are most popular. These businesses will be the most likely partners.

Encounter Management Capability

One of the biggest concerns we have for this project is the workflow and how the process will work to get from taking the customer request and completing them. In Nayak and Nigam's example, they used an insurance company as the service provider. They mapped out the workflow for filing a claim as seen below. This model is called the *Encounter Management Capability*. This model revolves around the process of taking a customer input and analyzing or calculating it before determining the financial decision. In our project, we aim to automate this model through the calculations of our criteria rankings and route-creating algorithm.

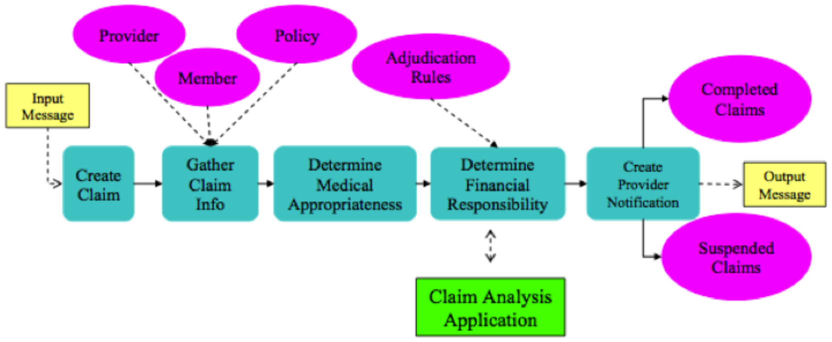


Figure 1 – Claim Adjudication Process Diagram (Encounter Management Capability)

Sustainability

Building a service-oriented business can be very difficult due to having a lot of problems with sustainability in profit. Although there are many opportunities to generate a lot of revenue, these opportunities can only be taken at the rate of the drivers. Drivers get an hourly wage, so having an incentive for serving every customer is important. DoorDash charges their customers a $5 to $7 flat fee per delivery (Entrepreneur.com), which encourages drivers to try to hit as many customers as possible within their shift. In the article “Value-added business models: linking professionalism and delivery of sustainability” written by Ilari Aho, having a performance-based business model can provide sustainability for a company. Unfortunately, having this type of model doesn’t always result in better performance, but can instead cause employees to just fall under pressure.

Incentives

Performance-based incentives are present mainly in sales, which require salesmen to hit a certain quota. An alternative to this is what Aho calls energy performance contracting, which is where the service provider implements a set of goals that were defined by the company to create improvement in the company. The job of all employees is to bring in revenue to the company, but those who can help the company grow are the ones that deserve the extra rewards. For our project, this model can’t be simulated, but a monitoring system to ensure all drivers are working honestly is an option.

Fare Collection

Yasuo Sasaki explains in his article “Optimal choices of fare collection systems for public transportation: Barrier versus barrier-free” that to make the most profit, there are different fare collection systems and one should pick accordingly based on the service. He explains that there are two major types of fare collection systems, which are barrier and barrier free. A barrier fare collection system means it would have a designated pay station and people would pay beforehand. A barrier-free system, or proof-of-purchase (POP), is where people check whether or not a rider has paid the fare or not, and charge them accordingly. An important point of this article is that the results shows by having an effective, more customer friendly fare system, has a big impact on travel demand. The article overall helped give a bit more insight on how we should charge customers because there could be different aspects we would need to take in account. These things can include the worker taking a longer route, therefore tricking the customer into paying more for a less efficient route. For our project, although we cannot implement a barrier or barrier free fare system, we can use some aspects of these fare systems into our own to rid of any potential problems we may have.

At some point, the problem of splitting fares may come up. This is where Douglas O.Santos comes in and tackles this problem in his article “Taxi and Ride Sharing: A Dynamic Dial-a-Ride Problem with Money as an Incentive”. They analyze an app that compute routes and computes requests to specific vehicles. The problem is considered dynamic because requests are processed online. Also, the computer also has to take into consideration the capacity of the vehicle and the maximum cost of the ride that the customer set. Santos solves this problem while meeting the constraints by dividing the day into separate time periods. To do this, he uses a procedure called greedy randomized adaptive search procedure (GRASP), which is an algorithm that is used to solve multi-level optimization problems. For our project, splitting the days into different time periods would not be relevant, but we can use GRASP to help us further optimize our routing system, while maintaining the rank of each task.

GPS

On the more technological side, Kevin M. McKay's article on "Integrated Automatic Vehicle Location Systems" goes over what programs typical transit agencies use to track each of their vehicles. Most if not all transit agencies use Automatic Vehicle Location (AVL), which geographically determines the location of a specific vehicle, combined with Computer Aided Dispatch (CAD), which dispatches vehicles based off of a computer. AVL has grown exponentially over the past few years because it has many benefits and improvements in areas including computer dispatching, promoting reliable on-time service, providing inputs to customer information systems and increasing passenger safety. A GPS system can give a head start on how to develop the route as well as many other perks such as on-time service and safety. However, implementing this type of live-feed system can be costly.

Customer Satisfaction

When it comes to customer satisfaction, service consistency definitely plays a big role. In Zhixing Luo's article "On service consistency in multi-period vehicle routing", he discusses the vehicle routing problem and explains why it is a requirement to have customers served by only a few different vehicles within a specified area. He does this with a mixed integer-programming model while creating a three-stage heuristic approach, which entails making educated guesses on certain aspects of the service. A mixed integer-programming model is an optimization method that includes continuous and discrete variables. By doing studies on service consistency, Luo found that the vehicle capacity does affect demand as much, but having a consistent service exponentially increases customer satisfaction, while only slightly increasing operational cost.

Design

Project Scope

The project scope of this project is to develop a working application that can simulate our model. We will also go over the advantages and disadvantages of using this model by comparing it to another, simpler model and calculating the financial impact of each.

Objectives:

1. Develop and analyze our route-optimization algorithm model
2. Design the process flow chart for the model
3. Prototype the computer application to follow the decision making process
4. Analyze the pros and cons of having a priority-based algorithm model

Model Development

Our proposed algorithm model will revolve around certain priorities of arrivals, and might not provide the minimum travel distance compared to a simpler model without priorities. We looked at multiple location-based algorithms to influence our own model. The first model we looked at was the Traveling Salesman Problem. This algorithm is based on the fact that the "salesman" must arrive at each node once, and then return to the start point. The priority system will be based off of a 0-10 scale, where 0 is the starting point, 1 is a high priority node, and 9 is a low priority node. The nodes with higher priority must be hit before considering any other node with a lower priority. A priority of 10 will be assigned to the same node with a priority of 0 since that node will be the start and end point of the route. Before running the TSP algorithm, the distance between each node must first be determined. A simple example of how a TSP diagram is set up is shown below in Figure 2 using eight cities. Once the diagram has been set up, the following steps are executed:

1. Place each salesman in a randomly chosen node
2. For each salesman, choose next node, updating the "tour cost" after every move
3. Once all salesman have returned back to their starting node, find the optimal tour

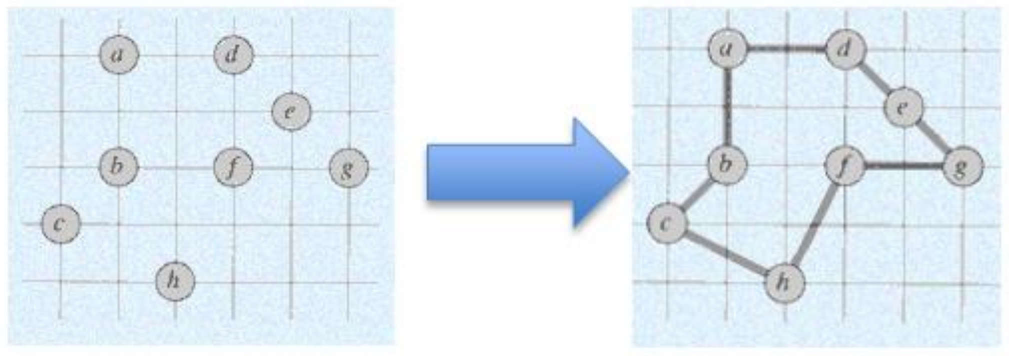


Figure 2 – Traveling Salesman Problem Diagram

The TSP algorithm's concept of making a stop at all nodes and coming back to the beginning heavily influences the way we will model our algorithm. Looking back at the diagram, we can treat nodes A, C, B, and D as a group of nodes with the priorities of 0, 3, 5, and 8 respectively. The algorithm must arrive at all priority nodes, and arrive back at priority node A, which is why node A will be given a priority of 0 and 10.

The next model we looked at is the minimum spanning tree, also known as Kruskal's algorithm. This is considered a greedy algorithm in graph theory. This algorithm only requires the traveler to arrive at each node once, but does not require him/her to return back to start. This algorithm can be visualized with another diagram example, as shown below on Figure 3.

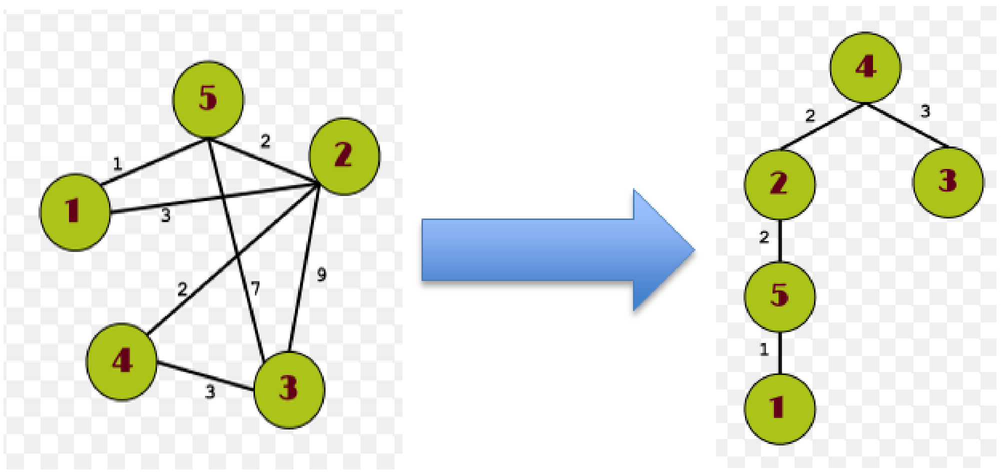


Figure 3 – Minimum Spanning Tree Diagram

When looking deeper into the priority group node, arrivals only need to be made at each node once, so the MST algorithm can be applied to this part of our new algorithm. This algorithm looks at one node at a time and finds the closest destination over and over again until every node has been reached. Once the “spanning tree” has been completed, it will move on to the next priority group.

Our implementation of the TSP algorithm is a little bit different than how it is generally used. A rule of thumb when using this method is to not have any paths cross throughout the whole route. If we have a group of nodes scattered on a map as shown below – with distances between each node not drawn to scale, the goal will be to optimize the route so that the driver must start at node 0, make a stop at all other nodes, and then arrive back to the beginning node.

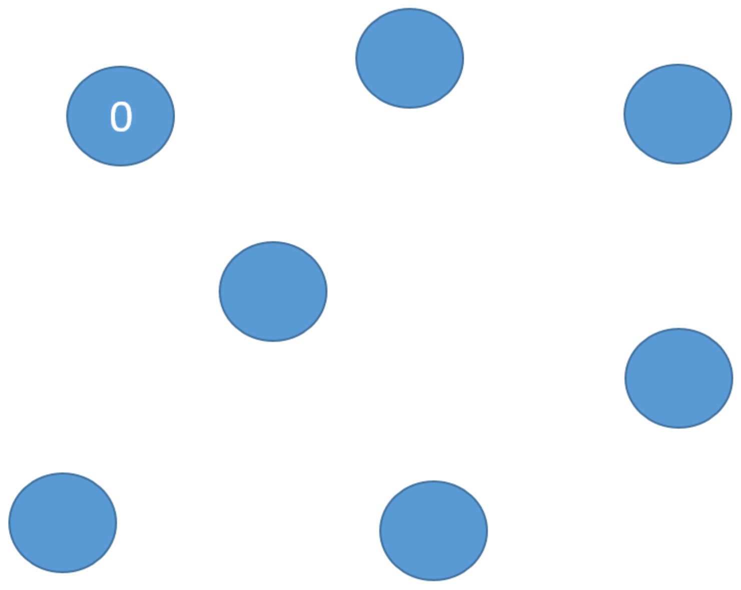


Figure 4 – Node Diagram Example

Using the TSP algorithm, the route would look something like Figure 5, where all the paths do not cross and the traveler would have made a complete "circle," minimizing the travel distance in the route and returning back to node 0.

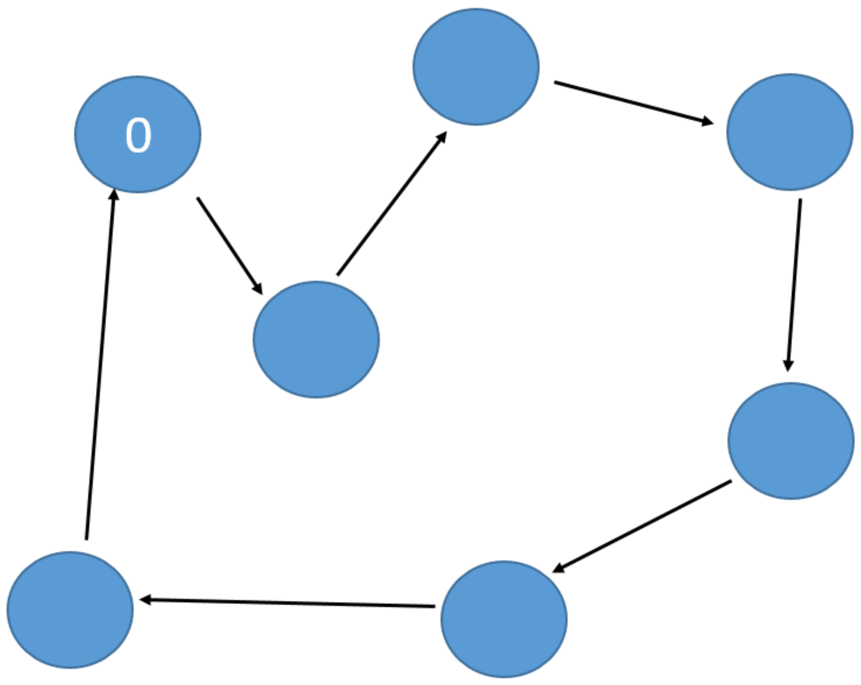


Figure 5 – Traveling Salesman Problem Diagram Example

However, with our model, crossing paths is necessary due to our model forcing the "salesman" to take priority into consideration before looking at distance. For our example in Figure 6, the model follows the priority rule. When starting at node 0, the route will make its way to the same two nodes, but will then arrive at a different node, due to the fact that there are still other destinations with a high priority. When arriving to the first node with a priority 1, the algorithm will know that there are two other nodes with the same priority, but will look for the closest node from that location.

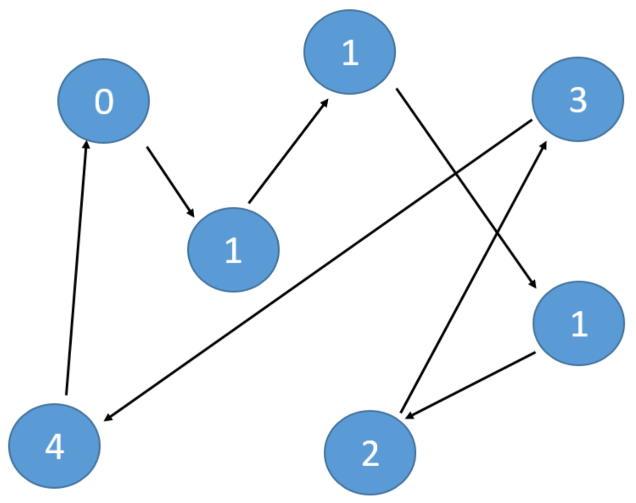


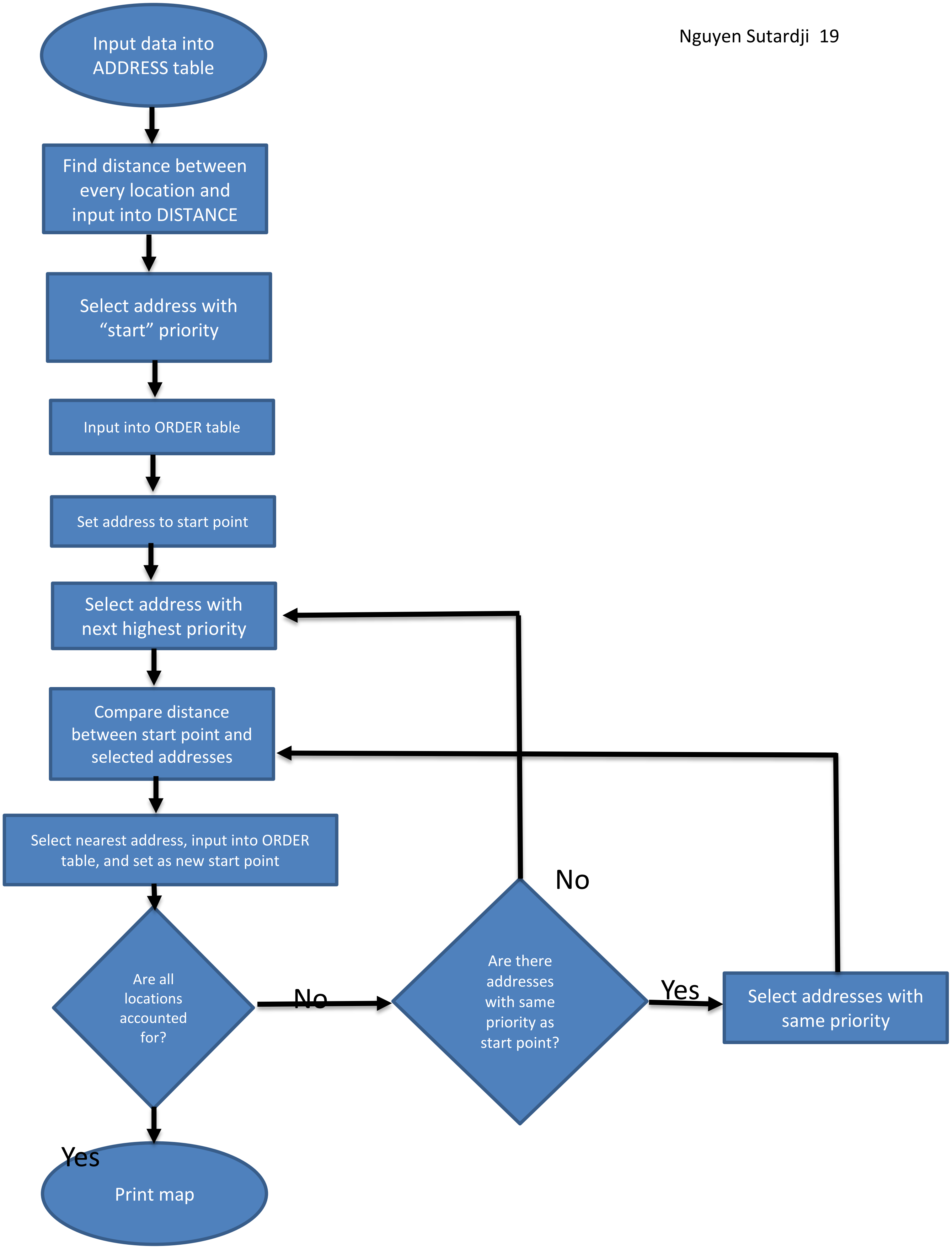
Figure 6 – Priority-Based Algorithm Diagram Example

The following steps for our priority-based algorithm are as follows:

1. Start at node 0
2. Move to the closest available node with the lowest priority
3. Move to closest node with same priority until no more available
4. Repeat steps 2 and 3 until there are completely no more available nodes
5. Return to node 0

Converting Algorithm to Code

Now that our mathematical model has been created, we must convert the algorithm so it can be programmatically sequenced through code. This was done with using Visual Basic and SQL on the server side, and HTML and Javascript on the client side. Before any coding was done, we first made a process flow chart to map out the logic of our code.


Input data into ADDRESS table
Find distance between every location and input into DISTANCE
Select address with "start" priority
Input into ORDER table
Set address to start point
Select address with next highest priority
Compare distance between start point and selected addresses
Select nearest address, input into ORDER table, and set as new start point
Are all locations accounted for?
No
Are there addresses with same priority as start point?
No
Yes
Select addresses with same priority
Yes
Print map

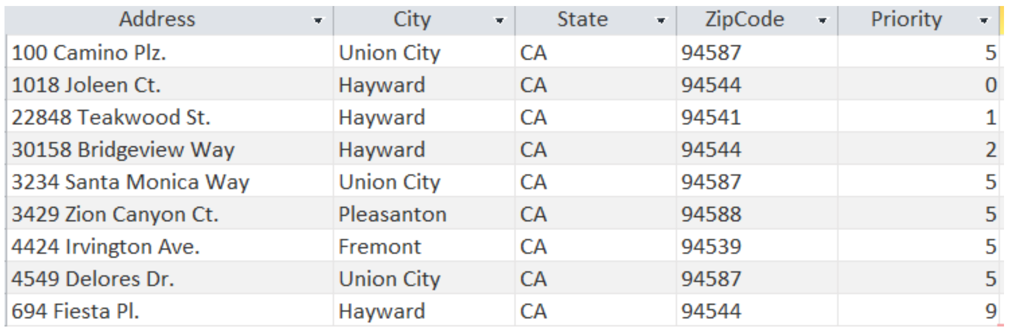
We decided that

Figure 7 - Flow Process Chart of Algorithm in Code Form

the algorithm can be best translated using SQL. This language of database management can look up certain criteria, most specifically the distance and priority criteria. Using "Select," "Insert," and "Delete" statements, we are able to manipulate tables, treating each record as an individual piece of data that can replicate the information given in a node diagram. All of the coding is done on the server side of Visual Studios.

Methodology

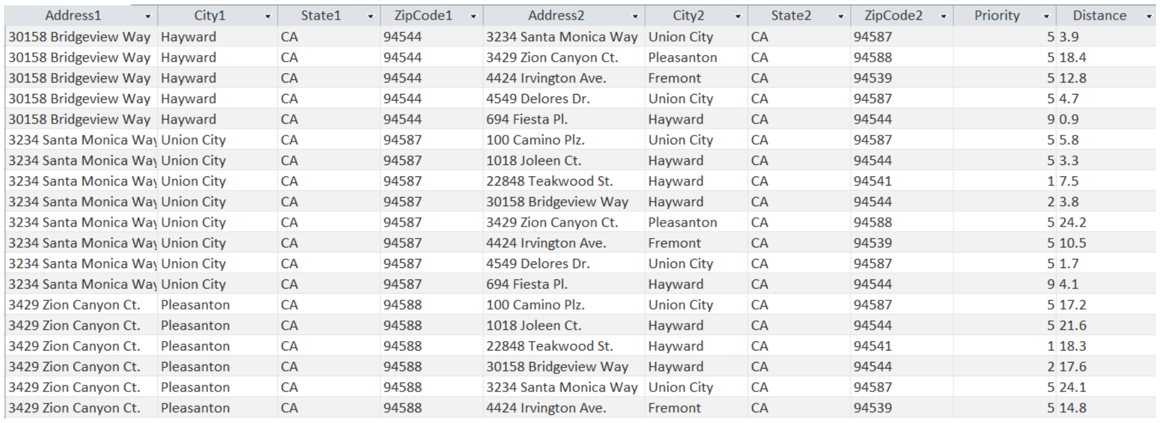
Testing and validating the computer application was fairly simple. We constantly changed the addresses and priority numbers to see if the algorithm was working every single time. Most of the testing was done during the programming portion of the project. A method of debugging code that we learned was to constantly check that the correct values and records were being selected by the "Select" statements by adding a line of code that would print the selected value every time it went through the loop. We also double checked that the tables were populated correctly. When adding data to the ADDRESS table, the following information is needed as shown in Figure 8.



Address	City	State	ZipCode	Priority
100 Camino Plz.	Union City	CA	94587	5
1018 Joleen Ct.	Hayward	CA	94544	0
22848 Teakwood St.	Hayward	CA	94541	1
30158 Bridgeview Way	Hayward	CA	94544	2
3234 Santa Monica Way	Union City	CA	94587	5
3429 Zion Canyon Ct.	Pleasanton	CA	94588	5
4424 Irvington Ave.	Fremont	CA	94539	5
4549 Delores Dr.	Union City	CA	94587	5
694 Fiesta Pl.	Hayward	CA	94544	9

Figure 8 – ADDRESS Table Layout

Once all the destinations along with their associated priority have been inputted, the distances are calculated between each address, populating the DISTANCE table. This is the equivalent of showing a diagram, connecting all of the nodes together to display the distance, or weight between each node.



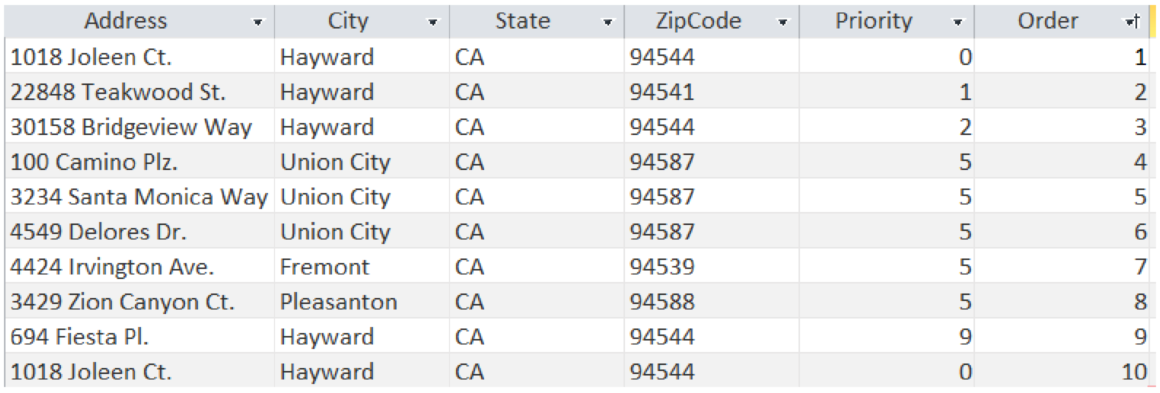
Address1	City1	State1	ZipCode1	Address2	City2	State2	ZipCode2	Priority	Distance
30158 Bridgeview Way	Hayward	CA	94544	3234 Santa Monica Way	Union City	CA	94587	5	3.9
30158 Bridgeview Way	Hayward	CA	94544	3429 Zion Canyon Ct.	Pleasanton	CA	94588	5	18.4
30158 Bridgeview Way	Hayward	CA	94544	4424 Irvington Ave.	Fremont	CA	94539	5	12.8
30158 Bridgeview Way	Hayward	CA	94544	4549 Delores Dr.	Union City	CA	94587	5	4.7
30158 Bridgeview Way	Hayward	CA	94544	694 Fiesta Pl.	Hayward	CA	94544	9	0.9
3234 Santa Monica Way	Union City	CA	94587	100 Camino Plz.	Union City	CA	94587	5	5.8
3234 Santa Monica Way	Union City	CA	94587	1018 Joleen Ct.	Hayward	CA	94544	5	3.3
3234 Santa Monica Way	Union City	CA	94587	22848 Teakwood St.	Hayward	CA	94541	1	7.5
3234 Santa Monica Way	Union City	CA	94587	30158 Bridgeview Way	Hayward	CA	94544	2	3.8
3234 Santa Monica Way	Union City	CA	94587	3429 Zion Canyon Ct.	Pleasanton	CA	94588	5	24.2
3234 Santa Monica Way	Union City	CA	94587	4424 Irvington Ave.	Fremont	CA	94539	5	10.5
3234 Santa Monica Way	Union City	CA	94587	4549 Delores Dr.	Union City	CA	94587	5	1.7
3234 Santa Monica Way	Union City	CA	94587	694 Fiesta Pl.	Hayward	CA	94544	9	4.1
3429 Zion Canyon Ct.	Pleasanton	CA	94588	100 Camino Plz.	Union City	CA	94587	5	17.2
3429 Zion Canyon Ct.	Pleasanton	CA	94588	1018 Joleen Ct.	Hayward	CA	94544	5	21.6
3429 Zion Canyon Ct.	Pleasanton	CA	94588	22848 Teakwood St.	Hayward	CA	94541	1	18.3
3429 Zion Canyon Ct.	Pleasanton	CA	94588	30158 Bridgeview Way	Hayward	CA	94544	2	17.6
3429 Zion Canyon Ct.	Pleasanton	CA	94588	3234 Santa Monica Way	Union City	CA	94587	5	24.1
3429 Zion Canyon Ct.	Pleasanton	CA	94588	4424 Irvington Ave.	Fremont	CA	94539	5	14.8

Figure 9 – DISTANCE Table Layout

After the next table has been set up, the algorithm will run in a loop, checking for the lowest priority available, running the MST algorithm, and repeating until the route has been generated. Once the order has been determined, a Javascript function will take the addresses and generate the route. Manual calculations were done in order to validate the authenticity of the application. Although it presented a route as an end product, it didn't always mean it was the most optimal according to the model we were attempting to follow. Mapping out to the process flow chart was the most helpful method of making sure we had all the steps done in the correct order. Creating the chart ended up being the more difficult part when it came down to creating the code process programmatically. Our experience in SQL allowed us to code the algorithm process quite easily.

Results

After inputting the data and testing the algorithm, the results were delivered in two main forms: a table labeled “ORDER” and a route mapped out on Google Maps. It was fairly easy to test the algorithm when there was only one address associated to each priority number, but we had to make sure it was following the MST model when determining the order of destinations with the same priority number. As mentioned in the methodology, we had to validate that within the addresses with a priority of 5, the algorithm selected the correct order by manually cross referencing the distances between all of those addresses.



Address	City	State	ZipCode	Priority	Order
1018 Joleen Ct.	Hayward	CA	94544	0	1
22848 Teakwood St.	Hayward	CA	94541	1	2
30158 Bridgeview Way	Hayward	CA	94544	2	3
100 Camino Plz.	Union City	CA	94587	5	4
3234 Santa Monica Way	Union City	CA	94587	5	5
4549 Delores Dr.	Union City	CA	94587	5	6
4424 Irvington Ave.	Fremont	CA	94539	5	7
3429 Zion Canyon Ct.	Pleasanton	CA	94588	5	8
694 Fiesta Pl.	Hayward	CA	94544	9	9
1018 Joleen Ct.	Hayward	CA	94544	0	10

Figure 10 – ORDER Table Layout

When the final table has been properly populated, the addresses are brought into the Google Maps API in numerical order. The important thing to note is that Google Maps will always calculate the route based on the order the addresses are inputted into the system, so validating that they were actually being inserted into Google Maps in the correct order was crucial. The final product came out as shown in Figure 11.

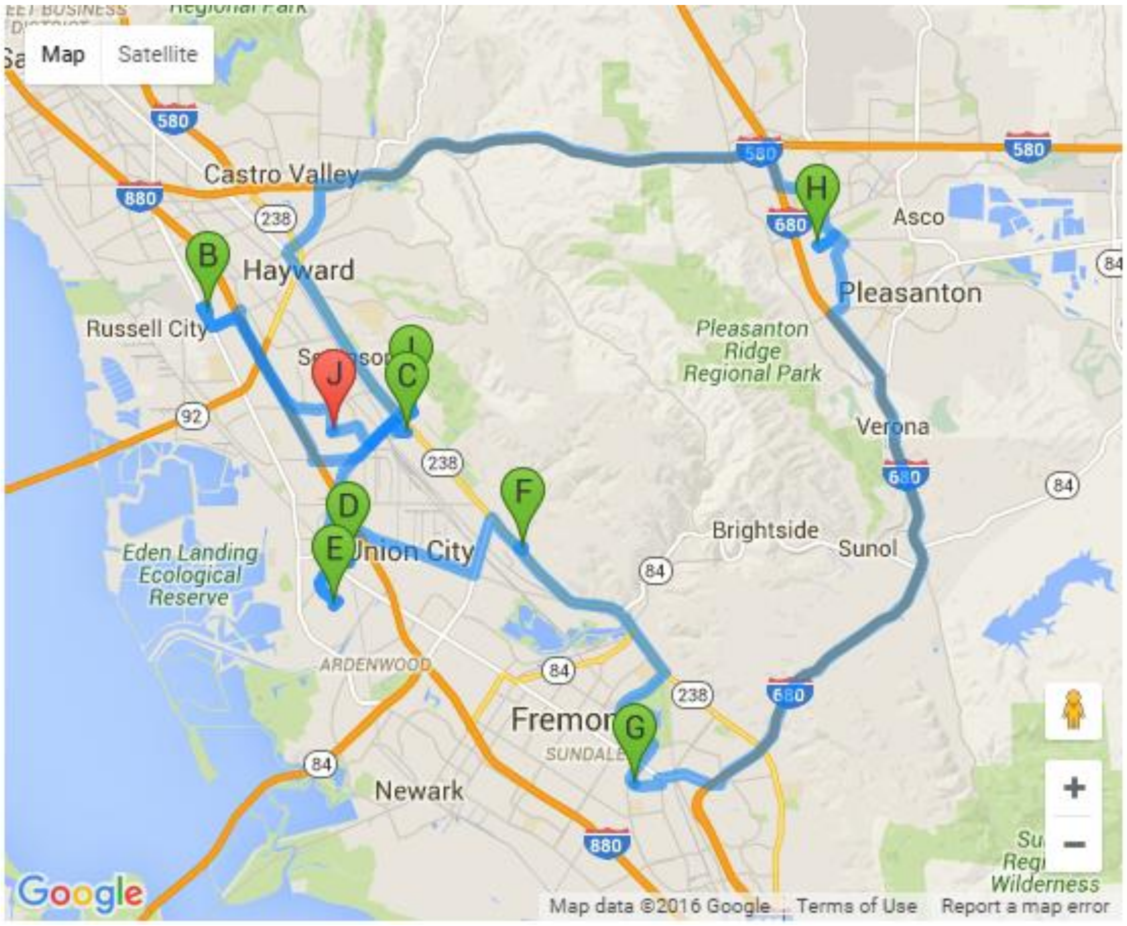


Figure 11 – Route Generated Using Priority-Based Algorithm

This specific route takes a total driving distance of 65.2 miles to complete the tour, which according to our model, is the most optimal route. We ran the same algorithm with the same addresses, but kept all the priorities except the start/end location at 5, and got that the total driving distance would be 51.8 miles.

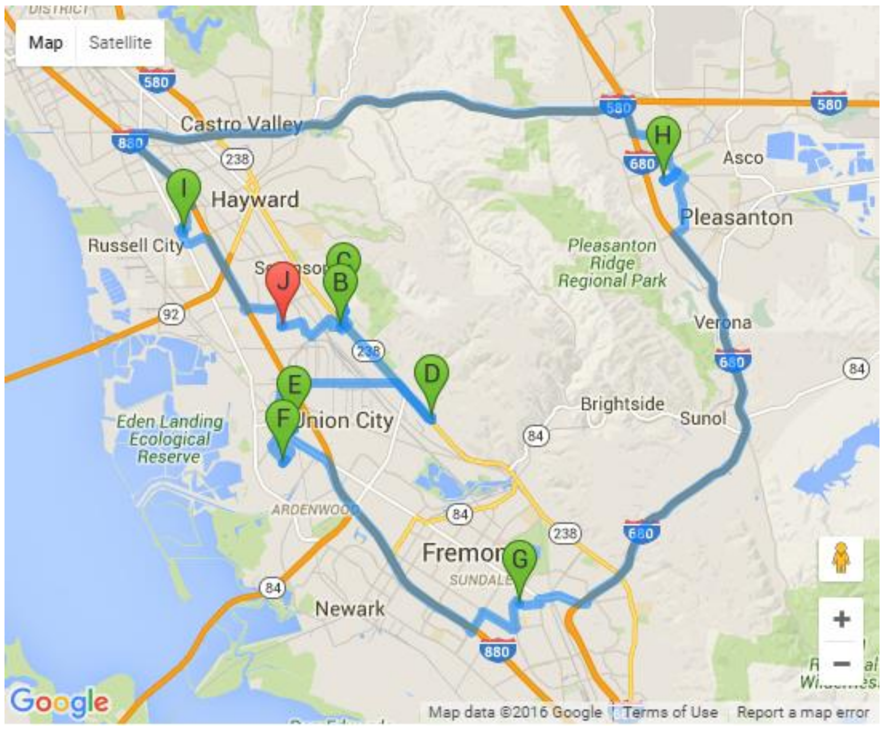


Figure 12 – Route-Generated Using TSP Algorithm

An economic justification can be made based on the situation of what this route is being used for. An example of how using our priority-based algorithm can be economically justified is if this route was made to deliver food. If prioritization is based on the availability of customers being home, then it would make sense to follow our model. Dropping food off at a customer's front door because they aren't home can pose a lot of issues. If the food gets stolen, or if it's raining and it gets the food packaging wet, then the customer would complain, and possibly never use your services again. If traveling an extra couple miles means keeping many of your customers happy, then it would make sense to spend a little more on travelling costs. Losing a potential regular customer can cost more than just the revenue from that one person. It can ruin your reputation and lose hundreds or thousands of potential regulars, and can even put your company out of business. However, if the revenue or profit coming in from catering to specific priorities and time windows does not outweigh the traveling costs, then using our model wouldn't be efficient. In the end, it all depends on the situation and the mission of the business.

Conclusion

There are many different routing algorithms to be looked at and considered when creating our new model, but in the end, we chose to stick to the two algorithms that closely matched our objectives. Operations research is an ongoing study that always finds something new to consider, and with the technology today, we can alter these algorithms to be even more complex than ever. Our algorithm, although simple, caters to the idea of "instant gratification whenever you need it".

Further research was conducted to see if implementing other features was possible. These features included a live traffic update, GPS tracking, switching to a mobile platform, and more. After going through all the additional features, it was concluded that these options required our application to be created through a different platform other than Visual Studios. Using Python and MySQL would be a more viable option due to the stronger capabilities if we were to consider adding these features. Replicating the algorithm wouldn't be too difficult, since it is already coded in SQL, but the biggest challenge is using Python as the main platform.

Companies such as Doorman, Instacart, Uber, and DoorDash pride themselves on serving you when you want to be served, whether it's as soon as possible, or if it's between 1PM and 2PM. Many people today are forced to always get their FedEx packages shipped to their office in anticipation of their package being delivered at a time they aren’t home. Things get lost, stolen or misplaced easily, so allowing the customer to dictate when services should be performed is key to providing customer satisfaction. It’s not just about getting something delivered on the right day, it’s about getting something delivered at the right hour, or even minute.

Bibliography

Aho, Ilaro. "Value-added Business Models: Linking Professionalism and Delivery of Sustainability." *Building Research & Information* (2012): 110-14. *Engineering Village*. Web. 23 Oct. 2015.

Ataka, Shinichiro, and Mitsuo Gen. "Solution Method for Multi-Product Two-Stage Logistics Network with Constraints on Delivery Route." *Electronics and Communications in Japan* 9th ser. 92 (2009): 456-61. *Engineering Village*. Web. 23 Oct. 2015.

Kovacs, Attila A., and Sophia N. Parragh. "The Multi-objective Generalized Consistent Vehicle Routing Problem." *The Multi-objective Generalized Consistent Vehicle Routing Problem*. Department of Business Administration, University of Vienna, 20 June 2015. Web. 16 Nov. 2015.

Lin, C.K.Y. "A Vehicle Routing Problem with Pickup and Delivery Time Windows, and Coordination of Transportable Resources." *Computers & Opeartions Research* (n.d.): n. pag. *Engineering Village*. Web. 25 Oct. 2015.

Luo, Zhixing, and Hu Qin. "On Service Consistency in Multi-period Vehicle Routing." *On Service Consistency in Multi-period Vehicle Routing*. N.p., 12 Jan. 2015. Web. 16 Nov. 2015.

McKay, Kevin M. "Integrated Automatic Vehicle Location Systems." *Ieeexplore.ieee.org*. IEEE AES Systems, n.d. Web. 16 Nov. 2015.

Nayak, Nitin, and Anil Nigam. "Modeling Business Services for Implementing on Global Business Services Delivery Platforms." (2007): n. pag. *Engineering Village*. Web. 23 Oct. 2015.

Osborne, Stephen P., Zoe Radnor, Tony Kinder, and Isabel Vidal. "The SERVICE Framework: A Public-service-dominant Approach to Sustainable Public Services." *Brit J Manage British Journal of Management* 26.3 (2015): 424-38. Web. 16 Nov. 2015.

Santos, Doughlas O., and Eduardo C. Xavier. "Taxi and Ride Sharing: A Dynamic Dial-a-Ride Problem with Money as an Incentive." *Taxi and Ride Sharing: A Dynamic Dial-a-Ride Problem with Money as an Incentive*. Institute of Computing, University of Campinas, 1 May 2015. Web. 16 Nov. 2015.

Sasaki, Yasuo. "Optimal Choices of Fare Collection Systems for Public Transportations: Barrier versus Barrier-free." *Transportation Research Part B: Methodological* 60 (2014): 107-14. Web.

Shoot, Brittany. "How This Startup Is Helping Restaurants Be More Efficient About Delivery." *Entrepreneur*. N.p., 1 Aug. 2015. Web. 2 Nov. 2015. <file://localhost/Users/rsutardji/Desktop/Senior%20Project%20Sources/Ryan/How%20This%20Startup%20Is%20Helping%20Restaurants%20Be%20More%20Efficient%20About%20Delivery.html>.