Maximum Donuts

Byung Su Jung, Henry Krider, Jesse Rivera, Jasper Yao

August 13, 2018

Abstract

The goal of this project is to determine the optimal location to open up a donut shop in Seattle. Our approach is to interpret easily obtained public data through the lens of a minisum location problem with some steps to reduce the computation cost via area restriction and simplified product demand. We extract an undirected node graph of Seattle from publicly available data and interpolate population in ZIP Code demarcated regions into node weights of intersections in the city. Once this is done we apply our minisum optimization to find ideally situated locations built with the assumptions of our model. As a test of our model's predictions of real world results we compare prediction vs actual location of donut shops in Seattle.

1 Introduction

In this project, we are looking for the single best place in Seattle to open a donut shop by determining the location where the expected number of regular customers will be maximized. This problem is pertinent to any business looking to open a facility in a city or similarly complex and layered network, in which information like effective distance and customer behavior can be hard to determine or model. Additionally, while data about a city's roads, population, geography, businesses, etc. are largely public and easy to obtain, working with this data and combining it into a functional tool that is useful to any business looking to open a store is a difficult task, as demonstrated by the fact that the vast majority of time spent on this project was in coding. Developing these tools for general use is important for the success of enterprising businesses. Furthermore, the methods we use can be applied to city planning and districting (e.g. choosing which areas to label as residential vs. business districts), as well as building city resources such as fire stations to reach as many people as possible.

Beyond this, being able to approach and solve this kind of network problem is useful to any system that can be modeled as a network of weighted edges and vertices where the goals is to maximize coverage. Examples of this kind of system might include building hospitals in a refugee camp with a network of rough paths, a certain population distribution, and a different distribution of injury, or an online network marketing problem, where effectiveness of an ad decreases with path distance and a maximum effectiveness is desired. Given the prevalence of network modeling problems, the wide applicability of them, and how common the event of opening a store is, the importance of this problem is evident.

This problem belongs to the large category of facility location problems. Facility location problems have been extensively studied in the context of operations research, and entail choosing where to construct facilities to, for example, minimize travel distance or maximize profit. The simplest version of this type of problem was posed by Fermat as early as the 17th century, and arguably led to the development of modern location theory and spatial economics [4]. Our problem can be categorized as a couple different particular subsets of facility location problems: single facility minisum location problems (also referred to as the Fermat-Weber problem), in which a sum of weighted distances is to be minimized, and single facility maximal covering location problem, in which the amount of node coverage is to be maximized. In this paper we treat it as a minisum location problem, but it could be approached as a maximal covering problem with equivalent results.

In beginning to formulate this problem, we assume Seattle to be static, with people staying in their houses except to visit our donut shop via an undirected network made up of intersections (nodes) and roads (edges). We assume that the likelihood of someone to visit our donut shop is based only on the shortest-path distance between them and

the shop, by some function $f(x, y) = 1/d(x, y)$.

In constructing this model, we weight each intersections by the approximate number of people that live near that intersection. Our algorithm is then constructed to maximize the function

$$F(v) \coloneqq \sum_{u \in V} \frac{w(u)}{1 + d(u, v)}$$

over all $v \in V$, where $w(u)$ is the node weight, $d(u, v)$ is the shortest path length computed by Dijkstra's algorithm, and $F(v)$ is the total number of expected customers.

To reduce the running time of this algorithm (which runs in $\mathcal{O}(n^3)$ time), we constrain the area we analyze, finding the best location in downtown Seattle as opposed to the whole city. In testing our algorithm's ranking ability we compared the scores of existing downtown donut shops to the average over all intersections, assuming that these donut shops are well placed. In testing our model's predictions of optimum shop location we compare our prediction with the actual locations of donut shops in Seattle, along with qualitative judgment of whether a location would be a good one to open a donut shop.

Given a 1 km radius section of downtown Seattle, our algorithm chose what we see as a qualitatively good intersection. However, there are a couple reasons to be wary of this result, given that its placement is somewhat non-intuitive and not particularly close to any existing donut shop. The algorithm also does not rank existing donut shops as above average, though this might be mostly attributable to our formulation of this test.

In improving our model, we look at addressing our big problems of cost and ranking ability, as well as general modifications of our current methods such as incorporating more census data into node weights, incorporating city data into edge costs, and adjusting our distance function to better approximate consumer demand. We also look at making certain elements like our methods of reducing node input and our distance function more rigorous.

We conclude that our methods form a good starting point for answering the complex problem of optimally placing a donut shop in a city. While in tests our algorithm leaves some to be desired, it does not unilaterally fail, and with more time it has a lot of room for growth and improvement.

2 Assumptions

We make the following assumptions in our model:

- We assume that people only travel in Seattle on public roads, and that these roads can be modeled as a network of edges and vertices, where edges represent streets, and vertices represent intersections. While people can travel though buildings, walking paths, etc., and a road network could be more accurately modeled as a more complex directed graph, these aspects are too complex for the scope of this project.
- We assume that people generally take the quickest path from one point to another, and by consequence, the effective distance between any two intersections is best approximated by the shortest path distance between them.
- In finding the shortest path distance, we assume that edge cost is equal to the physical length of streets, ignoring traffic, stoplights, and other factors in practical travel time for simplicity.
- We assume that a given person either is or is not a regular customer, and that whether or not a given person is a customer is based solely on the distance between the donut shop and the place they live. This assumes for simplicity that factors like traffic and population flow have minimal effect on the quality of a location. These assumptions are made in order to roughly approximate consumer demand on the preliminary basis that the closer a customer is to a donut shop, the more likely they are to patronize that shop. We do understand, however, that population flow is a significant aspect to address in future iterations of the model.

- We assume that the population distribution of Seattle can be accurately represented by weighting the vertices of our network, where each weight approximates the number of people who live closest to the given intersection. We assume that this number can be effectively approximated by uniformly distributing the population of a ZIP Code over the intersections within it. This is based on the real-world assumption that population distribution does not change abruptly within ZIP Code.
- We also assume that it is possible to place the donut shop at any intersection in Seattle, disregarding various practical considerations of possible shop placement. This assumption is made in order for a pre-existing list of potential locations to not be required to run the algorithm.
- Given the existence of multiple donut shops, we assume that a given customer's preference depends only on the distance to each donut shop (e.g. if there are two donut shops that are equidistant from a vertex, then we will assume that half of the customers at that vertex will be customers of one donut shop and the other half will be customers of the other).
- In testing our algorithm's quality in ranking different locations, we assume that existing donut shops are placed in high quality locations, and so an accurate algorithm will rank these locations as higher than average.
- The only factors that affect how many customers a donut shop will receive are the number of people who live near the donut shop and their proximity to it. According to Mark Cohen, a professor of marketing at Columbia University, "You can't do business in a brick and mortar store without the presence of customers, and proximity to customers is as elemental to a brick-and-mortar store as air, food and water are to each of us as human beings" [6].

3 Mathematical Model and Solution

A general minisum location problem is of the form $\min_{x\in\Omega} \sum_y w(y) \cdot d(x, y)$, where Ω is the feasible region and $w(y)$ are weights assigned to the points y whose distance from x is to be minimized [5]. Since our objective function is intended to model the number of expected customers, we instead consider the equivalent problem $\max_{x\in\Omega} \sum_y \frac{w(y)}{1+d(x,y)}$.

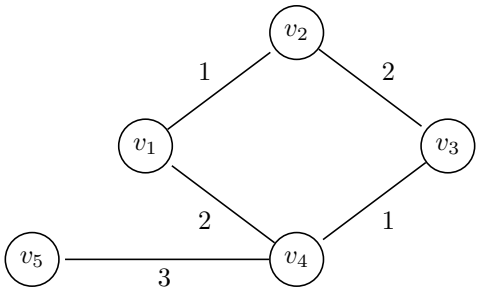
In our case, the feasible region is a connected (undirected) graph $G = (V, E)$ of the street layout of Seattle. Edges represent streets and vertices represent intersections. The edges are weighted to represent physical lengths of streets; to this end we assume that all edge costs are strictly positive. We define our distance function $d : V \times V \to \mathbb{R}_{\geq 0}$ as the shortest distance between two vertices in G. In practice, we compute this using Dijkstra's algorithm. We choose our weight function to be the population at a given vertex. That is, we define $w(v) \coloneqq$ number of potential customers at v. We want our objective function $F(v)$ to model the number of expected customers for a donut shop located at vertex $v \in V$, so it should increase with population and decrease as the population moves further away. We choose our objective function to be the simplest function satisfying these requirements,

$$F(v) \coloneqq \sum_{u\in V} \frac{w(u)}{1 + d(u, v)}.$$

Our strategy is to compute $F(v)$ for each vertex $v \in V$ and then manually select the vertex that maximizes $F(v)$. This is a computationally expensive procedure, however, we make several efforts to address this issue. For starters, we implement a cache to keep track of the distances between pairs of vertices. That is, every time a distance is computed we record this information and store it in a table so that it can be retrieved instantly the next time it is needed. This ensures that we never run Dijkstra's algorithm on any given pair of vertices more than once. In a large graph, it may also be beneficial to first use a clustering algorithm to find a subset of vertices $U \subset V$ with a high population density and then considering the modified problem of finding $\max_{v\in U} F(v)$. Due to time constraints, we were not able to implement a clustering algorithm. We instead manually selected a circle of radius 1 km encompassing downtown Seattle and ran our algorithm on the subset of vertices within the circle.

3.1 Toy Problem

For illustrative purposes, we first demonstrate our strategy on a simple toy problem. Consider the following graph with $w(v_1) = w(v_2) = w(v_3) = w(v_4) = 300$ and $w(v_5) = 600$:



Using Dijkstra's algorithm, we compute the following distances:

	v_1	v_2	v_3	v_4	v_5
v_1	0	1	3	2	5
v_2	1	0	2	3	6
v_3	3	2	0	1	4
v_4	2	3	1	0	3
v_5	5	6	4	3	0

We then compute

$$F(v_1) = \frac{300}{1} + \frac{300}{1+1} + \frac{300}{1+3} + \frac{300}{1+2} + \frac{600}{1+5} = 725$$
$$F(v_2) = \frac{300}{1+1} + \frac{300}{1} + \frac{300}{1+2} + \frac{300}{1+3} + \frac{600}{1+6} = \frac{4975}{7} \approx 710.71$$
$$F(v_3) = \frac{300}{1+3} + \frac{300}{1+2} + \frac{300}{1} + \frac{300}{1+1} + \frac{600}{1+4} = 745$$
$$F(v_4) = \frac{300}{1+2} + \frac{300}{1+3} + \frac{300}{1+1} + \frac{300}{1} + \frac{600}{1+3} = 775$$
$$F(v_5) = \frac{300}{1+5} + \frac{300}{1+6} + \frac{300}{1+4} + \frac{300}{1+3} + \frac{600}{1} = \frac{5795}{7} \approx 827.86$$

and conclude that v_5 would be the optimal location to open a donut shop.

3.2 Competitors

Seattle is a big city that is already home to many donut shops. As such, we cannot expect all donut-eaters to be regular customers of our shop. To address this issue, we modify our objective function by multiplying each term by some proportion $\pi \in [0, 1]$. This proportion can be interpreted as the proportion of donut-eaters that we expect to choose our donut shop over our competitors'.

Let $C \subset V$ denote the set of competing donut shops. We assume that if multiple donut shops exist, then customers' preference depends only on the distance to each donut shop. The proportion π should therefore depend on the location of our donut shop, v, the location of the donut-eaters, u, and the locations of our competitors, C. We want this function $\pi(u, v, C)$ to increase as v gets closer to u, and decrease as u gets closer to vertices in C. We ended up choosing our new objective function to be

$$F(v) \coloneqq \sum_{u \in V} \left[\pi(u, v, C) \cdot \frac{w(u)}{1 + d(u, v)} \right],$$

where

$$\pi(u, v, C) = \frac{1/d(u, v)}{1/d(u, v) + \sum_{c \in C} 1/d(u, c)}$$

is the proportion of original customers from vertex u we expect to retain at a donut shop at v after accounting for competitors. We define $\pi(u, v, C) = 1$ if $u = v$ and $\pi(u, v, C) = 0$ if $u \in C$. Also note that $\pi(u, v, C) = 1$ if C is empty.

The following are several desirable properties of $\pi(u, v, C)$ that one would expect from a function that models the proportion of retained customers. We chose $\pi(u, v, C)$ to be the simplest function we could find satisfying these properties. In each of the following propositions we will assume that $C \subsetneq V$ is arbitrary, and that $u, v \in V$ are any vertices satisfying $u \neq v$ and $u \notin C$ (this avoids needing to address trivial edge cases in each proof).

Proposition 1. $\pi(u, v, C) \in [0, 1]$.

Proof. It is obvious that $\pi(u, v, C)$ is nonnegative. To see that it is less than or equal to 1, simply notice that the numerator can never be larger than the denominator. □

Proposition 2. $\pi(u, v, C) \to 1$ *as* $d(u, c) \to \infty$ *for all* $c \in C$. *That is, as our competitors move farther away we will eventually expect to retain all our customers.*

Proof. In the limit as all $d(u, c) \to \infty$ we have $1/d(u, c) \to 0$ for all $c \in C$, thus

$$\pi(u, v, C) = \frac{1/d(u, v)}{1/d(u, v) + \sum_{c \in C} 1/d(u, c)} \to \frac{1/d(u, v)}{1/d(u, v)} = 1.$$

□

Proposition 3. *Suppose* $\operatorname{diam}(V) = \sup_{x,y \in V} d(x, y) < \infty$ *(realistically,* $\operatorname{diam}(V)$ *is bounded by the circumference of the Earth, and in our case it is bounded by the diameter of Seattle). Then* $\pi(u, v, C) \to 0$ *as* $|C| \to \infty$. *That is, as more competing donut shops open up we will eventually lose all of our customers.*

Proof. First note that for all $c \in C$ we have $1/d(u, c) \geq 1/\operatorname{diam}(V)$, thus $\sum_{c \in C} 1/d(u, c) \geq |C|/\operatorname{diam}(V)$. Letting $|C| \to \infty$ we obtain

$$0 \leq \pi(u, v, C) = \frac{1/d(u, v)}{1/d(u, v) + \sum_{c \in C} 1/d(u, c)} \leq \frac{1/d(u, v)}{1/d(u, v) + |C|/\operatorname{diam}(V)} \to 0.$$

□

Proposition 4. *Let* $c_0 \in C$. *Then* $\pi(u, v, C) \to \pi(u, v, C \setminus \{c_0\})$ *as* $d(u, c_0) \to \infty$. *That is, if a competing donut shop is sufficiently far away from* u *then we need not consider it a competitor for the potential customers at* u.

Proof. In the limit as $d(u, c_0) \to \infty$ we have $1/d(u, c_0) \to 0$ so that

$$\pi(u, v, C) = \frac{1/d(u, v)}{1/d(u, v) + \sum_{c \in C} 1/d(u, c)} \to \frac{1/d(u, v)}{1/d(u, v) + \sum_{c \in C \setminus \{c_0\}} 1/d(u, c)} = \pi(u, v, C \setminus \{c_0\}).$$

□

Proposition 5. $\pi(u, v, C) + \sum_{c \in C} \pi(u, c, (C \cup \{v\}) \setminus \{c\}) = 1$. *That is, the proportions of donut customers going to each donut shop should sum to 1.*

Proof. Let $D = C \cup \{v\}$. Then $\pi(u, v, C) = \frac{1/d(u,v)}{\sum_{c \in D} 1/d(u,c)}$. Our goal is to show that $\sum_{x \in D} \pi(u, x, D \setminus \{x\}) = 1$. Indeed, we have

$$\sum_{x \in D} \pi(u, x, D \setminus \{x\}) = \sum_{x \in D} \frac{1/d(u, x)}{\sum_{c \in D} 1/d(u, c)} = \frac{\sum_{x \in D} 1/d(u, x)}{\sum_{c \in D} 1/d(u, c)} = 1.$$

□

4 Data Acquisition

In order to use real data for the project, we extracted intersection and road data from OpenStreetMap by using the Osmnx package in python. We found the length of each road in Osmnx as well.

Once we acquired the intersection and road data, we needed population data for each intersection. In order to find the population data, we used ZIP Code as a connection between intersection and population. We found the right ZIP Code for each intersection from JSON using Google API Geocode. We also found population for each ZIP Code in United States using U.S. Census data. With every intersection with ZIP Code assigned, we calculated the number of intersections in each ZIP Code, and divided the ZIP Code's population by the number of intersections in order to obtain a uniform distribution. Then, we assigned population to each intersection.

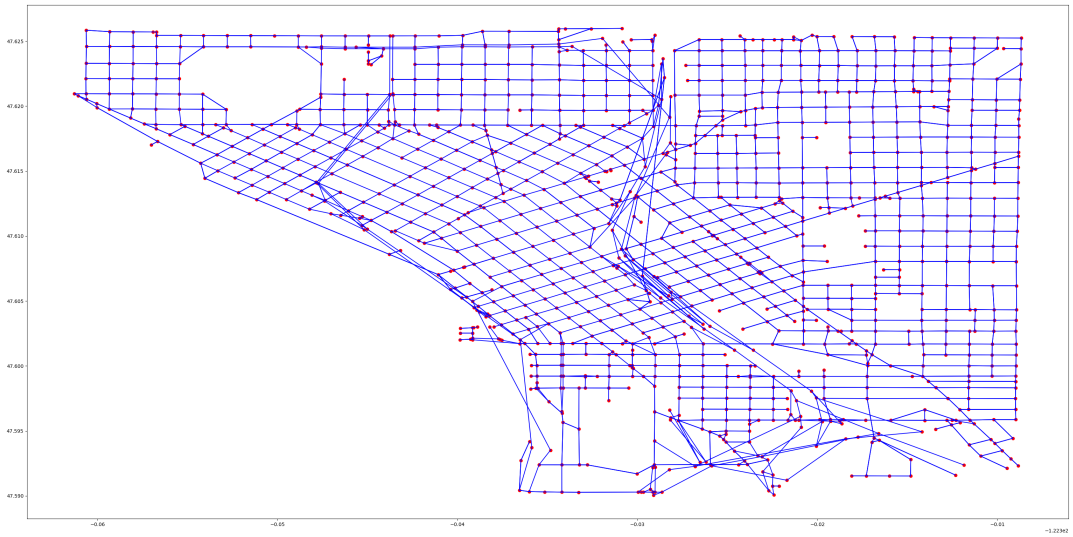


Figure 1: A graph representing the street layout of downtown Seattle.

5 Testing

We have three basic measures of whether our methods are effective or not:

- Given the ability to choose any node in a network, does our algorithm choose an optimal location that looks qualitatively like a good location for a donut shop?
- Given a list of existing donut shop locations (which we assume to be better than average locations), does our algorithm rank these locations as better than the average?
- Does our algorithm run in a reasonable amount of time?

While the third of these is not a constraint unique to our problem, it is an important one given the cost of our algorithm and size of the road network of Seattle, and the one that acted as the hard constraint in our project.

To the second of these, we took a 1 km radius circle encompassing downtown Seattle and compared the scores of the donuts shops within the circle to the mean and median of all the nodes in the circle. When we perform this measurement, we find that the scores of existing donut shops is slightly above average, a score of 127.51 compared

to 112.844. However, this result might not be attributable to our algorithm, but rather to the fact that three of the seven shops sit on the edges of our graph, and therefore likely suffer due to our area constraint and not necessarily because of the true quality of their placement. This points to a flaw in our algorithm - not in our objective function, but in the way we constrain our input. If we were to constrain the amount of possible placements, but not the nodes that contribute to total customers, we might find a more satisfactory answer to this test. Unfortunately, due to time constraints we are not able to make this change. Additionally, as we'll expand on in our results section, our algorithm does not fail the first test we proposed, and so might still make a good starting point for approaching this problem.

6 Results

Our scores represent the steady state expected value of the number of customers that we expect to visit a donut shop per unit time. Given that we were looking to predict the best location, rather than forecast a realistic customer number, we believe it is sufficient for the scores to vary relative to each other to make sound location predictions.
Taking the 1 km radius block of downtown Seattle, our algorithm chose the intersection of Cherry street and Boren ave, which sits on the eastern edge of the cluster. Looking around this intersection, it makes sense as a location for a donut shop. It is sandwiched between the Frye Art Museum, Seattle University and O'Dea High School, and is in an area that's semi-residential with some smaller businesses and organizations. In other words, prime donut country. This location scored approximately 152 in the competitive model and around 195 in the noncompetitive one. Notabley it is situated similarly with respect to other donut shops in the downtown, namely around the periphery of the urban core.

However, it also seems a little odd that our algorithm placed the best location on the edge of the map and not somewhere in the middle. A clue to what might be happening is in our Toy Problem example, where a more isolated node has a higher score by virtue of a much higher population. Because we distribute population based on ZIP Code, the eastern edge of our graph might be dipping into another ZIP Code with higher population density. Looking at how our algorithm distributes population on this map, as well as a gradient map of scores could go a long way in figuring out more thoroughly how our algorithm is working in practice. It is also interesting to note that we ran this simulation model both while accounting for competition and without. In both cases the results were the same.

Additionally, when we ran our algorithm to consider competition from other shops, our optimum location doesn't change, its score just decreases. This is not an entirely surprising result given that our shop location was not extremely close to any others. Since our algorithm allocates a fraction of customers inversely proportional to distance from a location, we expected that competitors exerted minimal spheres of influence. In our model therefore, proximity to competitors is not very significant as evidenced by high density of donut shops downtown regardless. However, it might be useful in the future and when working with larger areas to be less generous with our model of competition, which assumes that all shops are equivalent in quality, popularity, price, etc. This likely wouldn't be the case in reality, and assuming worse odds might give us a safer location.

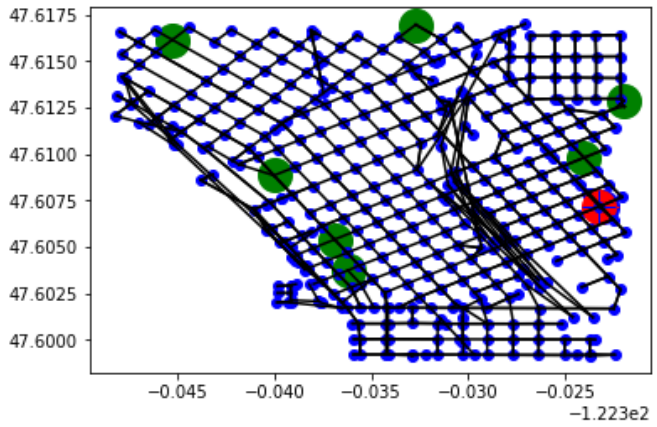


Figure 2: Graph showing our optimal location in red, and competing donut shops in green.

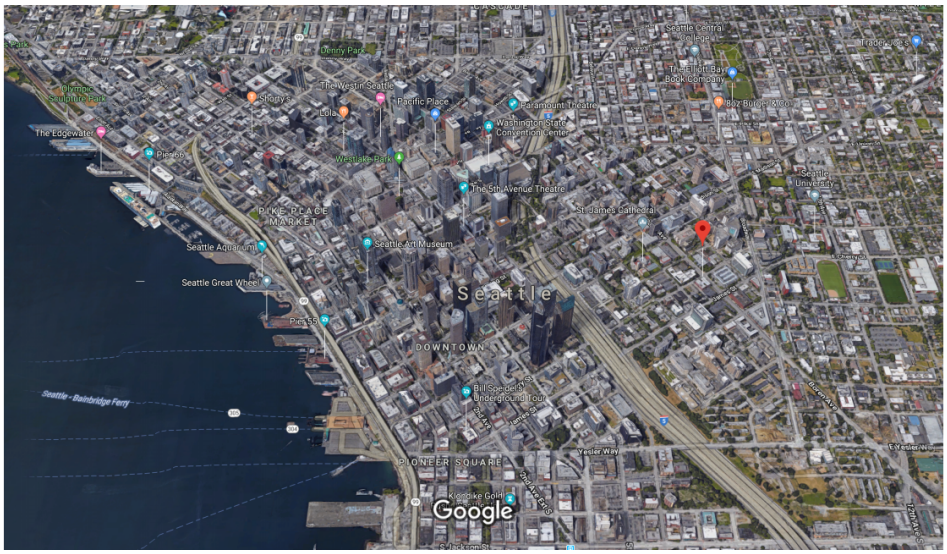


Figure 3: Google map position of optimal location.

7 Improvements

What did our model notably fall short in?

One of the biggest challenges of our algorithm is working with its associated cost constraints. Given a network of n nodes, our algorithm runs in $\mathcal{O}(n^3)$ time complexity, so we are quite limited in what formulation we can input into the algorithm. We used an area constraint centered on a high density neighborhood in order to decrease the number of nodes we analyze, but there are a number of ways in which we might decrease the number of nodes we input with varying results. For example, we can easily rank a list of viable locations, we can decrease the granularity of our network, we can restrict the area to other various arbitrary regions, and we can use various clustering algorithms to find sub-networks of a certain number of nodes. In improving our solution we might either make the initial problem more specific (e.g. finding the best location in Capitol Hill for a donut shop, or finding the best neighborhood in Seattle for a donut shop), or make our choice of which solution to our input constraints we use more rigorous (and perhaps variable based on specific desires and constraints on inputs and outputs).

Another clear area of improvement is addressing our tests and the changes they point to. Our formulation of the ranking test, meant to calibrate our algorithm based on existing donut shop locations, does not produce unambiguous results. Improving the rigor of the test and making adjustments to our algorithm from established research presents a way forward. These adjustments may be simple changes such as tweaking node and edge weights and refining our functions of distance and competition, as described further down. However, our assumption that existing donut shops are well placed may not be an entirely accurate when accounting for other economic and legal factors such as land costs and zoning. In the future when we do adjust our algorithm we will reconsider the validity of our current analysis.

What are the various other considerations that we didn't cover that could improve the model?

A big consideration that we ignore in this project for the sake of simplicity is population flow, i.e. how people are walking, biking, and driving around Seattle and how that can help us find a more optimal location for a donut shop. After all, we expect a scenario of someone stopping in for a donut as they're running errands or just walking down the street to be quite common, and possibly even more so than someone leaving their home specifically to patronize our donut shop. Finding ways to model or indirectly approximate this would be very important were we to seriously approach the problem of opening a donut shop.

In addition to this there are a number of smaller ways in which we could add to and improve our model, without needing to formulate an entirely new element such as population flow.

First of all, we didn't modify our objective function to take into account the units of measurement that are used in the edge costs in the data. This would amount to essentially multiplying the objective function by some constant, and consequently it would not affect the outcome of our algorithm. It would, however, affect our interpretation of $F(v)$ as the number of expected regular customers. In order to improve our model, we could try out other objective functions of the form

$$F(v) = \sum_{u \in V} \left[\pi(u, v, C) \cdot \frac{w(v)}{1 + \alpha \cdot d(u, v)^k} \right]$$

where α and k are parameters that we could tune by looking at real world data on the amount of customers that existing donut shops receive. The parameter k determines how much of an effect distance has on whether or not someone will be a customer. If someone lives twice as far away from our donut shop are they half as likely to be a regular customer? One fourth as likely? We simply don't know, so answering this question and gathering supporting data is one way in which we could improve our model.

Another improvement could be weighting edge costs differently based on factors that affect travel times. For example, we might expect the slope of a street to correlate to travel time, since residential streets are more likely to be hilly and highways more likely to be flat. This data is publicly available as well, and could be easily incorporated. We might also look into traffic data and attempt to better model the cost of a given path.

We could also modify our vertex weight function $w(v)$ based on consumptions models, such as the likelihood to buy a donut as determined by age, income, education, and similar. These are all publicly available via census data so it would not be to hard to add them in. We would hope to better model consumer behavior and relate that to our node weight and behavior in the future. It is further possible to adopt a full transit network model considering pedestrian and vehicular traffic which could include both public and private traffic and transient reoccurring traffic patterns in the city.

On the business end a possible improvement would be to improve our model of competition. We assumed that the total number of donut shop customers from a node are invariant with respect to the existence of a competitor, and that the ratio of customers to either shop is the ratio of the distance functions between the node and the shops. We believe however, it is likely that the behavior of customers in a competitive market is far more complex and price sensitive. In the future, precedents of intra-market competition modeling could help us better optimize our shop location given the contested donut business in Seattle.

Lastly, as a general improvement we could incorporate different types of city data other that population, roads, and existing donut shop location, such as walk scores, location of all businesses and the industry they belong to, and

property values, both to model a location's desirability and to some extent business costs.

Incorporating these different layers of data and weighing them appropriately could give us a more comprehensive picture and model of the city of Seattle. Undoubtedly this will demand tight integration with our preexisting solver as the complexity of the data increases and create unforseen challenges as new dimensions are added.

8 Conclusions

We conclude that our methods, though imperfect, form a good starting point for answering the complex problem of optimally placing a donut shop in the city of Seattle. While in tests our algorithm is not proven to have strong predictability, it does not unilaterally fail, and its optimum location for a donut shop makes a good amount of intuitive sense, indicating a decent basis for further development. Our methods have a lot of room for growth and improvement, and we regret that we did not have more time to implement them.

Apart from our methods themselves, we found this problem to be programmatically challenging, and we struggled with data compatability, API limitations, and several other problems that only cropped up at the eleventh hour in devlopment. Developing rigorous methods are important for approaching mathematically similar network problems, but it is also important for businesses and cities to have resources so they can perform useful analysis on publicly available data, which their accessibility are not quite as amenable to analysis as they first appear. If we were to take any broad lessons from this project it would be that public data does not necessarily make problems easier. In fact, it is tools that make data useful that are invaluable. It takes significant time and energy to turn public data into actionable intelligence.

References

[1] O. Berman, D. Krass, and Z. Drezner. The gradual covering decay location problem on a network. *European Journal of Operational Research*, 151(3), December 2003.

[2] R. Z. Farahani and M. Hekmatfar, editors. *Facility Location: Concepts, Models, Algorithms and Case Studies.* Springer Science and Business Media, 2009.

[3] M. K. Gavina and J. Rabajante. Mathematical programming models for determining the optimal location of beehives. *Bulletin of Mathematical Biology*, 76(5), March 2014.

[4] G. Giorgi and T. H. Kjeldsen, editors. *Traces and Emergence of Nonlinear Programming.* Springer Basel, 2014.

[5] M. Kon and S. Kushimoto. A single facility minisum location problem under the a-distance. *Journal of the Operations Research Society of Japan*, 40(1), March 1997.

[6] B. Thau. How big data helps chains like Starbucks pick store locations – an (unsung) key to retail success. *Forbes*, April 2014.