CIS 422

SOFTWARE METHODOLOGIES I

PROJECT 1

THE TRAVELING SALESPERSON PROBLEM

Juan Flores September 28, 2021

1. OVERVIEW

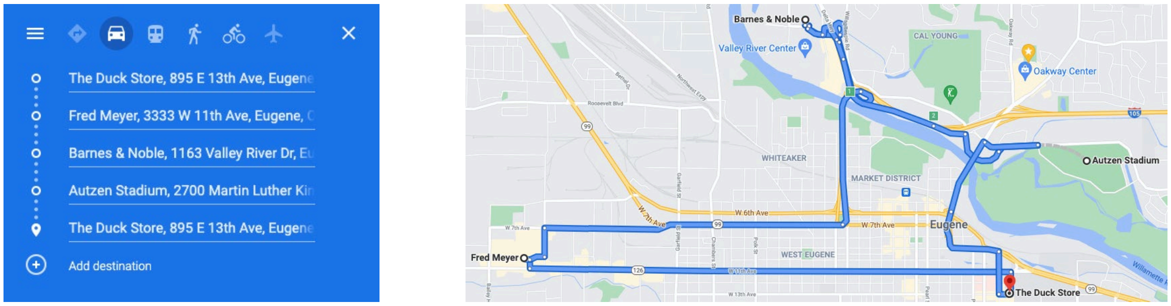
An algorithm is a well-defined and unambiguous sequence of steps to solve a given problem. The concept of algorithm is neither new not it is exclusive to computer science. The first evidence of algorithms dates back to 2500 BC. In computer systems, algorithms are hardware and language-independent descriptions, generally expressed in pseudocode.

In many cases, algorithms are related or based on a given data representation (i.e., data structures.) That happens so often that Niklaus Wirth wrote a book titled "Algorithms + Data Structures = Programs" [1]. Algorithms represent solutions to problems of different nature. Those problems range from abstract mathematical ones to problems with a mathematical basis that find applications in real-life situations. The problem you are to solve in this project is one such instance.

The owner of a commercial distribution company, X, reaches you expressing that the company needs to cut down (i.e., minimize) expenses in the product distribution process. X has several trucks to deliver products to customers. At the beginning of the day, each driver gets a list of customers and the products they need. (Let us assume personnel has loaded the track with the products to be delivered.) The driver must deliver all products during the workday. Some drivers are more skilled than others (perhaps they know the city better) and are more efficient in time and gasoline. We also know that earlier delivery times produce happier clients.

The Traveling Salesperson Problem (TSP) is the following: given a list of addresses, a delivery vehicle driver needs to determine the optimal (most efficient) route to visit them all using the fastest or shortest route that visits all clients and returns to the original point (home).

Drivers have tablets, and it would be nice to add visit points (customer addresses) by tapping on a map or downloading a file. Clicking on a button should produce the desired route. Let us say you have to go to visit all locations shown in Fig. 1 (a). Fig. 1 (b) shows a possible route to follow.



(a) (b)

Fig. 1. (a) List of locations to visit, and (b) one possible route.

There are commercial applications that solve the problem for large companies (i.e., FedEx or Amazon), but that does not prevent you from developing another service to solve the TSP.

2. THE TRAVELING SALESPERSON PROBLEM

Given a set S of points in $\mathbb{R}^2$, one of them denoted as s, the starting point, produce a sequence of them such that the sequence length is the minimum of all possible sequences.

Assuming there is a path from every point to every other point in S, S induces a complete graph with no self-loops. We can reformulate the problem as: find the minimum length path on that graph.

A graph $G = < V, E >$ is composed by a set of vertices, V, and a set of edges, E, connecting pairs of vertices in V. Let us assume $|V| = n$, so $V = \{v_1, \cdots, v_1\}$, then $E = \{(v_i, v_j) \mid 1 \leq i, j \leq n, i \neq j\}$. In a directed graph, edges are pairs, i.e., there is a connection from vertex v_i to vertex v_j, and not necessarily vice versa. A weighted graph $G = < V, E, w >$ is a graph (as described above), where w is a function $w: E \rightarrow \mathbb{R}$, i.e., each edge has an associated weight or cost.

A path in G is a sequence of k nodes $p = (v_{i1}, v_{i2}, \cdots, v_{ik})$, where $v_{ij} \in V$ and $v_{ij} \neq v_{il}$ $1 \leq j, l \leq k$, i.e., the path is acyclic. We can convert the path into a loop assigning $v_{ik+1} = v_{i1}$. The path cost is $c(p) = \sum_{j=1}^{k+1} w(v_{ij,} v_{ij+1})$. The minimum cost path p^* is the path that represents the minimum cost over all possible paths in G.

Computing a minimum cost path p^* is an NP-Complete problem (it has no known efficient algorithm to provide an optimal solution), so we will produce an approximation to p^* in the sense that $c(p) \geq c(p^*)$, but $c(p)$ is still acceptable for practical purposes. Cormen et al. [2] (35.2, p 1111) provide the first approximation. A second possibility is to solve the problem using Genetic Algorithms [3].

The TSP arises in many transportation and logistics applications. Examples of problems that require a solution to the TSP include designing school bus routes (the motivation for Merrill

Flood, one of the pioneers of TSP research in the 1940s), scheduling of service calls at cable firms, delivery of meals to homebound persons, scheduling of stacker cranes in warehouses, automatic drilling of printed circuit boards, and routing of trucks for parcel pickup or delivery, among many others.

3. IMPLEMENTATION

You are to design a web app that solves the TSP. This section mentions some critical axes or directions in which you can make choices that lead to different implementations of your team's final solution to this project.

3.1. APP MODALITY

In current computer technology, there are different ways to produce computer services. You must choose how to offer your customer the services your company provides.

An App (short for application) is a system that delivers some functionality and solution to a problem your customer may face. Apps may provide a service, be it educational, personal, entertainment, or business.

Computerized solutions can be wrapped and delivered in different forms. The most commonly recognized App formats are:

Web App – application software that runs on a web server, in contrast to computer-based software programs run locally on the device's operating system (OS). Web apps are accessed through a web browser over the internet. Web apps run on a client-server architecture. Examples of commonly-used web applications include webmail, online retail sales, and online banking.

Native App – aka mobile App or just App, is a computer program or software application designed to run on a mobile device such as a phone, tablet, or watch. They are called native because they are generally developed for one kind of device (IOS, Android, or Windows.)

Desktop App – a software program that can be run on a standalone computer. This type of software system is also tied to the operating system they are developed on (Windows, Mac, Linux, etc.) Examples of desktop applications are word editors, media players, and gaming.

I suggest developing a web app, but you are free to implement your solution in any form, as long as the instructor and graders can execute it.

3.2. APPROXIMATE SOLUTIONS TO THE TSP

If you are fond of graphs and graph algorithms, you can use the first solution proposed in Section 2. The algorithm APPROX-TSP-TOUR guarantees to produce a cost no more than twice the optimal cost. APPROX-TSP-TOUR solves the problem by creating a minimum spanning tree T of the graph, using Prim's algorithm; it then returns the Hamiltonian Cycle of the preorder walk of T.

A second solution can be produced by evolutionary computation, in particular, using Genetic Algorithms (GA) [4]. A GA encodes a proposed tour (a permutation of the vertices in G) as an individual's gene; it repeatedly applies the genetic operators of Selection, Mutation, and Crossover to evolve the population of possible solutions. The best solution of the last generation is the approximated solution. Nowadays Artificial Intelligence is permeating in all fields of human activity. GA is not part of Machine Learning, which is the most popular part of AI. GA is part of a different area called Meta-heuristic optimization, a set of stochastic algorithms that perform a directed search in the solution space based on the knowledge acquired by previous explorations.

You may perform bibliographic research and find out other ways to produce non-optimal solutions to the TSP. Whatever algorithm you use, you must briefly describe and explain it. If you use a different algorithm, you must document your choice and its rationale.

3.3. USER INTERFACE

One idea of a user interface for your web app that solves the TSP is to implement a direct manipulation mechanism that operates directly on a map. You can tap on locations on the map to set the customers' addresses or read a file with that information and display it on the map. The computation of a solution's cost requires the distances from every customer to every other customer. Once the solution is computed, the map must display the results using the same visual means.

Google Maps is a web mapping platform offered by Google; it provides all the services our application requires. You need to create a Google Developer Profile [5] to use Google Maps API.

Although I strongly suggest using the Google Maps API, feel free to use any other maps library. If you use another service, library, or API, you must document the reasons for your decision.

4. BUSINESS MODEL

Let us assume you researched existing solutions to the problem, consider their strengths and weaknesses, and produce a solution that outperforms existing solutions in execution time, solution quality (cost), or usability. Having a system that can compete against the existing products, you may want to commercialize your solution.

In that case, you need to define a business model to sell your software or the services it provides to the public (companies of all sizes and needs) [6]. Before you formulate a business model, you need to answer the following questions: What is a product business model? What types of business models are there? What is a revenue model? What are the components of a business model?

In this part, you do not need to submit a formal business model but discuss the different marketing possibilities and how you would go about commercializing your product. You do not need to include this part in your submission, but you must have it in your project presentation.

5. TECHNICAL REQUIREMENTS

This section presents a minimum set of technical requirements. Your projects should generate many more than just these.

i. The delivered system must be complete. Even if you deliver a subset of the required functionality, the provided system must execute without significant bugs. The graders do not have time to debug your software.
ii. Systems should use standard libraries to the extent possible. Follow the guidelines provided to document your code and mention what each library provides.
iii. Provide the required instructions to install, execute, and use your system.

6. TEAMS, MEETINGS, AND DELIVERY

You will be working in groups of five students. Early in the class, we will study team organization; you will use those concepts to organize your team, divide work, etc. Before each meeting with the customer (see below), every student will turn in a peer evaluation form. Keep that in mind when accepting your part of the work. Also, each team must submit a brief report of the team organization. More about this will be detailed in class.

In a real-life project, you need to meet with your customer or a designated committee for the following reasons, among others:

i. Initial meeting – that is where the customer specifies the need for a new system.
ii. Questions and concerns – developers may have questions about the processes included in the system or the system environment (its ecosystem).
iii. Quality specifications – these are specifications required by the customer or proposed by the developer.
iv. Specifications of standards – standards represent general knowledge of the application field.
v. Changes to specifications – procured by the customer. These better happen before you start any implementation or detailed design. You must sign a contract before the project begins.
vi. Progress reports – either previously agreed and scheduled, requested by the customer, or proposed by the developer team.

Taking this scenario to this course project:

- I will be your customer.
- Each team will be assigned two meetings before the final presentation.

Project delivery is the final presentation of the project to the customer. For larger projects, there may be several deliveries at the end of each planned project stage.

Preliminary meetings will occur during my office hours, and the final delivery will take place during class. Those meetings appear in the class schedule, but their dates may change slightly, depending on your advance. In this project, the final delivery will occur during class and in the form of a brief presentation.

7. REFERENCES

[1] Wirth, Niklaus (1976). Algorithms + Data Structures = Programs. Prentice-Hall. ISBN 978-0-13-022418-7.

[2] Cormen, Thomas H., et al. Introduction to algorithms. Third Edition. MIT Press, 2009.

[3] Juneja, Sahib Singh, et al. "Travelling salesman problem optimization using genetic algorithm." 2019 Amity International Conference on Artificial Intelligence (AICAI). IEEE, 2019.

[4] P. Larrañaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, and S. Dizdarevic, "Genetic algorithms for the travelling salesman problem: a review of representations and operators," *Artificial Intelligence Review*, vol. 13, no. 2, pp. 129–170, 1999.

[5] https://developers.google.com/maps/documentation/maps-static/get-api-key

[6] https://www.altexsoft.com/blog/business/software-business-models-examples-revenue-streams-and-characteristics-for-products-services-and-platforms/