CMSC 27200 Project Design Document

Jamar Sullivan, Jason Jia

May 2022

1 What have you thought about so far? Which ideas seemed promising, and which ideas didn't seem so promising?

We have thought about the following ideas so far and considered their potential effectiveness.

Our main ideas that seem promising are:

- Primary Approach: Greedy algorithm with a heuristic, where the algorithm repeatedly finds the next best node based on certain useful metrics. This idea does not necessarily give the optimal solution, but we believe suitable metrics can give a good approximation.
  - We plan to augment it with partial brute force, where we try all possible nodes for the first (or second) node and then perform the greedy algorithm from those nodes (updating the current utility and timestamp).
  - The greedy algorithm can be tweaked easily in response to challenging inputs, and this ability to generate iterative improvement makes it attractive.
- Alternative Approach: We could use an evolutionary/genetic algorithm to create sample, initially random solutions that would converge to a higher utility solution. This would involve defining the chromosome of the agent, and how we decide to give it utility. The fitness function will be built to prefer the agents with higher utility, and consider different crossover and mutation functions.

Our other ideas that didn't seem so promising are:

- Dynamic Programming to fully solve the problem. This idea is appealing in theory, but coming up with the subproblem and recursion will be difficult.

- An algorithm that first tries to complete all attractions, and if unsuccessful, iteratively remove attractions using some heuristic until the route becomes feasible. However, we don't see a feasible implementation from this idea.

2 What is your approach for the algorithm? How will it process the input, and how will it find a good solution? What heuristics or ideas do you plan to leverage, and why?

Our primary approach is to implement a greedy algorithm with partial brute force.

Our pre-processing is as follows:

- We remove nodes with 0 utility.
- We also create a graph with the weights of each edge as the time taken to travel between the two nodes (rounded up to the nearest integer).
- Initialize a list of feasible nodes F containing all nodes.

Our main loop is as follows:

- While there are still feasible nodes remaining:
- We remove infeasible nodes from F_t, defined as nodes where $S_{i,j,t}$, the sum of traveling time $d_{i,j}$ (from Jonathan's current location i), waiting time $w_{j,t}$, time spent at the attraction t_j, and the traveling time $d_{j,O}$ (from the node j to the origin) exceeds the remaining time available (at the current time t).
- We define time $T_{i,j,t} = d_{i,j} + w_{j,t} + t_j$.
- Heuristic: We determine the next node j to travel to, given that Jonathan is at point i at time t, by choosing the node j with the highest score $S(i,j,t) = \alpha \frac{u_j}{T_{i,j,t}} + \frac{1}{|H|} \sum_{h \in H} \frac{u_h}{T_{j,h,(t+T_{i,j,t})}}$, where H is the set of all feasible nodes given that Jonathan is at node j at time $t + T_{i,j,t}$, and has already visited a certain sequence of nodes.

Our justification is as follows:

- We wanted to capture the idea that a node is more likely to be the optimal node if it has high average utility over time, where time includes the time required to travel from Jonathan's current location, wait for the attraction to open (if needed), and finish the attraction.

- We also wanted to capture the idea that a node j is more likely to be the optimal node if choosing this node now still allows Jonathan to choose other nodes with high average utility over time. We do this by averaging the weighted utilities of nodes that are further accessible from node j.
- We plan to try out different values of α, which captures the relative weight placed on "immediate" versus "future" average utility.

3 What tools or services do you plan to use?

We plan to use Python with local machines.