Algorithm Project Report

Prepared by

Ali Maher, Mehdi Mortazavian

6th of Jul 2024

Introduction

The introduction provides an overview of the Orienteering Problem (OP). It explains that the problem involves finding the shortest route between control points while visiting each point only once. The goal is to maximize the total score obtained from the control points, where each point has a different score. The OP has real-world applications in logistics, transportation planning, and recreational activities.

What mentioned in the project documentation is an orienteering problem. We can say this problem is a combination of "Knapsack" and "TSM" problems.

Input

As mentioned Iin the doc, our inputs are data instance of the problem which first line of it contains:

Tmax *#P*

Where the Tmax is out budget for traversing the nodes through a path and second parameter shows the number of paths we are going to involve. (But in this project, we assume P =1)

And following lines shows our nodes properties as below, which x is x coordinate, y is y coordinate and s = score of visiting one node:

x *y* *s*

Note: we are using Euclidean distance in this project not the Manhattan one.

As this problem is a NP hard, so we should find the optimum solution with our tools. Now let's go through our approach to solve them.

Approach 1: Greedy Algorithm _ A

In this approach, we are going to use the greedy algorithm and our choice property is to pay as less as possible. So, by this I should find the closest node possible at each level.

Approach 2: Greedy Algorithm _ B

It is still greedy but we are going to change our choice property. This time we think for the most score. So, by this I mean we should find the node with the highest profit each time.

Approach 3: Greedy Algorithm _ C

It is still greedy :) This time we think for the most profit. And I define the profit as the difference of each node scores and the cost (distance) of visiting each node.

$$Profit = Score\_N - Cost\_N$$

Approach 4: Greedy Algorithm _ D

This is the modified version of the approach-3. Now we are going to set an error coefficient. By changing this parameter each time, we will get different result. So, we should define this error coefficient wisely to get the best answer.

This error coefficient is defined as below:

$$Score / Distance >= Error\ coefficient$$

Note: In the above four approach from greedy mentioned. But we can still improve them. How? Randomly ;) In decision making level we will reach at a point that the value of two or more points will become equal. And at this time, we can use random functions to choose between them.

Approach 5: Dynamic Programming

First of all, we create the "dp" memory and initialize it. After that we reach to the recurrence part which I used following recursive function:

$$dp[j][t] = max(dp[i][t - int(distance\_matrix[i][j])] + profit, dp[j][t])$$

point j, i index of previous control point ,time t

And finally, we reach to backtracking to extract our path from the above result.

Approach 6: MST _ Kruskal

Here I used a graph algorithm to find MST. But why? I am going to make the edges list from subtracting score and distance of the nodes. After that I have two ways: 1. Multiply the value to -1 and find the MST, which this leads to a spanning tree which that gives us the max profit. 2. In Kruskal after we get the edges list then in order to sort the edges increasingly, we should do it reverse.

Approach 7: MST _ Prim

Here I used another graph algorithm to find MST. As mentioned above…

Note: The MST is not the direct solution for this problem. We should find the MST and start to pruning the edges of the tree, till the we reach the maximum path.

Approach 8: DFS with randomized selection and bad nodes pruning

DFS with randomized selection and bad nodes pruning is an algorithm that explores the search space using depth-first search (DFS), selecting random neighbor nodes and pruning nodes with low scores, aiming to find an optimal solution for the Orienteering Problem.

Summary:

We present some algorithms with their implementations for make the best answer.

More details are commented on the files. So, it would be easier to track.

The project files with doc are also available on my [GitHub](#).

Thanks for your time