AssignmentOne

SOFE 3720/ CSCI 4610: Introduction to Artificial Intelligence/ Artificial Intelligence

Winter 2019

Dr. Sukhwant Kaur Sagar

Submission Deadline: Tuesday, Feb. 12, 2019, 11:59 PM

Submission Guidelines

One submission per group and no more than three students per group, please.

Student Number	Banner id	Student Name
1.		
2.		
3.		

Note:

a. **Email submissions will not be accepted.**
b. **Only one copy of the solution to this problem needs to be submitted per team.**
c. **Make sure that all of you contribute equally; in your write up, explain how you divided the work.**

This assignment is developed for a cross-listed (SOFE + CSCI) Intro to Artificial Intelligence course taught as part of a semester-long study for city of Oshawa. This is a city in Ontario, Canada, on the Lake Ontario shoreline. In this assignment, we will be using the characteristics of streets and elevation changes to try to find optimal routes for walking through city of Oshawa. The basic underlying structure will use A* to find the "shortest" path, but it will be up to you to decide how to define shortest, and to make sure your heuristics work with the definition that you make. In particular, developing a cost function and appropriate (admissible) heuristic given that cost function require a bit of care.

Data Requirements: You will need to generate two data files: an OpenStreetMap XML of the routes, and an elevation map [Samples are uploaded in BB]

A) For the routes, go to openstreetmap.org and search for the location you are interested in. Click on the "Export" button and you will get an XML file that can be used. The provided starter code will extract the "ways" (streets as well as pedestrian routes). The XML can contain all information to build the map of the area, including many things which are not streets, such as the coastline, cliffs, parking lots, etc. It will be up to you to parse this XML so that your search uses only the relevant pieces. You will probably find it useful to read up on how OpenStreetMap is defined as well as to use some code to sift through the XML to see what types of features are present here. Note that all nodes should have a latitude and longitude associated with them.

B) For the elevation, you are required to have elevation data. The HGT file (elevations as simple binary integer data) provided as a sample is from the SRTM data set. You are required to select yours as per the part of the city you choose. This could be obtained from earthexplorer.usgs.gov and is available for most of the world. You have to create a login in order to access the data. Look under the Data Sets tab for Digital Elevation -> SRTM -> SRTM Void Filled. Currently this appears to have the same data only in other formats - if you download the BIL format and unzip it, there will be a .bil which contains the same form of data but in little-endian format! . This file contains a high-density grid of elevation data. Specifically, it contains a 2-byte integer representing height in meters for each cell of city grid. The grid should cover the area between latitude and longitude, and the data is row by row from north to south. Thus, you should fairly easily be able to determine the (approximate) elevation of any particular node in the OSM database.

You may develop a graphical solution to this problem, as it will probably assist in your debugging process. To assist you in this, some startup code is provided that reads in each of the two data sets and creates a GUI window with some lines and circles.

Your final solution should be able to take (in some form) two OSM nodes and return the shortest path between them along with the expected time to walk along the path. You must use the elevation change and distance between nodes as part of your calculations, but beyond that you

may choose your path costs in any way you believe is reasonable. Note that when calculating distances, one degree of latitude is larger than one degree of longitude - the given code has some useful information in this regard.

Submission/grading

You will submit your code (both in report as well as link to GitHub repository) and a write-up showing some graphical outputs. The write-up will contain discussion and defense of your cost calculation and associated heuristic function. You should include results (Screenshots or GUI) from some sample runs and your observations on how well these match your personal experience.

Grades will be assigned as follows:

- Correctness of A* search, including heuristic: 40%
- Correctness of data handling: 10%
- Implementation and defense of path cost function: 25%
- Efficiency of solution: 10%
- Writeup: 15%

**********GOOD LUCK**********