SCSSE, University of Wollongong

CSCI203/803 Data Structures and Algorithms

Spring 2016

Assignment 3 (Due 11:59 PM Fri 30th September)

Objectives

- Design and implement a program based on a greedy algorithm to solve the Minimal Spanning Tree (MST) problem;
- Choose and implement appropriate data structures for the algorithm;
- Analyse the efficiency of different implementations of the algorithm in comparison with the brute force algorithm.

Task

You should write a single program which implements Prim's algorithm with different data structures, as well as the brute force search algorithm, to find the MST of an undirected, connected, weighted graph.

Program

Your program should read a symmetric matrix from a text file that describes weighted edges of an undirected, connected graph and find the MST that is saved in an output file as a sequence of edges of the MST. The program also displays the numbers of vertices and edges in the graph and the time spent to find its MST for each data structure used in Prim's and the brute force search algorithms, which can be saved in another output file as well for your later analysis.

Your program should only measure the time spent by the algorithm with its data structure to find the MST so that it is a proper design to use one method/function to implement an algorithm and its associated data structure. For an algorithm that operates on a weight matrix, it takes the weight matrix stored in a two-dimensional array as its parameter; for an algorithm that operates on adjacency lists, it takes adjacency lists you created from the weight matrix. You may create your own list data structures based on arrays. No STL or Java Collection or other collection framework can be used. In the part of the algorithm implementation, you should not use any library functions including maximal or minimal functions but create your own functions to operate on arrays.

Your program should be readable and well commented.

In addition, you needs to create a small program or script to generate the weight matrix for a graph of various vertices and edges, saved to a text file. When you create graphs, you need consider the density and connectivity of graphs. You may use a program/script by other people, with reference to the source, to generate these matrices.

Data Structures and Algorithms

- Brute force search algorithm
  The algorithm operates on a two-dimensional array that represents the weight matrix.

- Prim's algorithm
  The algorithm operates on one of the following data structures:
  1. two-dimensional array that represents the weight matrix and unordered arrays;
  2. adjacency lists and heaps.

Requirements

<u>CSCI203 students</u> (10 marks or 6 marks)

1. Program
   You have two choices on how to write your code.
   - Implement the brute force search algorithm and Prim's algorithm operating on one data structure of your choice; (3 marks for each)

   or
   - Integrate the brute force search algorithm and Prim's algorithm operating on one data structure of your choice. In this case, you may use code available in public domains. Properly cite the sources of your code in your report and program. (1 mark for each)

2. Report
   a. Cover page
      Student name
      Subject code
      Student number
      Email ID
      (0 mark; your assignment will not be marked if there is not this section)
   b. Introduction (around 100 words)
      Describe an application scenario where the MST is applied and the background and purpose of your experiments. Cite at least one reference. (1 mark)
   c. Methods (around 200 words)
      Describe algorithms you used in your experiments and data you used to produce the results. (1 mark)
   d. Result analysis (no more than 500 words)
      Analyse and discuss your results with reference to time complexity of relevant algorithms (cite the sources). You need to produce results with substantial differences in time spent for each algorithm and data structure. You should use at least one plot to present the results in terms of time spent and graph sizes. (2 marks)
   e. Conclusion (around 50 words)
      Summarise your findings and interpretations.
      (0 mark; -1 mark if there is not this section)
   f. References
      (0 mark; -1 mark if there is not this section; -0.5 marks for each improper reference)

<u>CSCI803 students</u> (10 marks)

1. Program
   Implement the brute force search algorithm and Prim's algorithm operating on two data structures: matrix and heaps. (2 marks for each)

2. Report
   a. Cover page
      Student name
      Subject code
      Student number
      Email ID
      (0 mark; your assignment will not be marked if there is not this section)
   b. Introduction (around 200 words)
      Describe application scenarios where the MST is applied and identify one application as the background and purpose of your experiments. You should interpret well the meanings of the vertices and edge weights, and why it is an MST problem in your identified application. Cite at least three references. (1 mark)
   c. Methods (around 300 words)
      Describe algorithms you used in your experiments and data you used to produce the results. (1 mark)
   d. Result analysis (no more than 600 words)
      Analyse the time complexity of your algorithms and discuss your results. You need to produce results with substantial differences in time spent for each algorithm and data structure. You should use at least one plot to present the results in terms of time spent and graph sizes. (2 marks)
   e. Conclusion (around 50 words)
      Summarise your findings and interpretations.
      (0 mark; -1 mark if there is not this section)
   f. References
      (0 mark; -1 mark if there is not this section; -0.5 marks for each improper reference)

Other requirements

- You must cite references from academic sources. Internet URLs including Wiki pages are not generally acceptable except they are used for links to resources.
- All your programs including scripts will have the following header:

```
/*
 * CSCI203/803 Assignment 3
 *
 * Student name:
 * Subject code:
 * Student number:
 * Email ID:
 *
 * (The following heading goes all places where appropriate if
 *   all or part of code in your program is not written by you.)
 * Author:
 * Sources: (citation of the sources)
 */
```

- Your programs can be compiled and executed with bash scripts called **`a3compile`** and **`a3run`**, respectively, on Ubuntu setup in the lab.
    - The script **`a3run`** should be configured to run for a small graph that can finish within a fraction of a second.
    - The script should be configurable to run for a graph of any size.
    - The script can execute a program/script to generate the matrix before execute your MST program.
    - Failure to compile and run your program to produce results from your scripts will receive 0 mark.
- Write a text file named **`readme`** to explain how your matrix is created, special consideration of your program compilation and execution, or any information related to your program that is necessary to configure and run your program to produce the results in your report. All your results reported in the report must be possibly produced by your submitted programs with easy configuration described in the readme file.
- Report, named **`report.pdf`**, is created in the PDF format.
- Failure to meet any requirements above may result in loss of marks.

Submission instructions

Your completed assignment should contain your **`source code`** including all programs and scripts, **`readme`** and **`report.pdf`** in a zipped file named **`ass3.zip`**. You should not include any graph matrices in the submission. In this assignment, all input files will be generated by your program/scripts.

To submit your assignment, transfer your **`ass3.zip`** file to **`banshee`** and run the command:

```
turnin -c csci203 -a 3 ass3.zip
```

from the directory containing your file.

1. Late submissions will be marked with a 25% deduction for each day.
2. Submissions more than three days late will not be marked, unless an extension has been granted.
3. If you need an extension apply through SOLS, if possible before the assignment deadline.
4. Plagiarism is treated seriously. Use of code not created by you without citation is plagiarism. If we suspect any work is copied, all students involved are likely to receive zero for the entire assignment.