CS 6301.502. Implementation of advanced data structures and algorithms

Fall 2017; Fri, Oct 6.

Long Project LP3: Optimal branchings (directed MST) in directed graphs

Ver 1.0: Initial description (Fri, Oct 6).

Ver 1.1: Added missing information about max score of 100 (Fri, Oct 27).

Due: 11:59 PM, Sun, Oct 29 (1st deadline), Sun, Nov 12 (2nd deadline).

Max score: 100. Max excellence credits: 10.

Initial code base: ArrayIterator, Graph, GraphAlgorithm, XGraph, LP3, Timer

You can also use code from other examples shown by the instructor in this class

* Submit before the first deadline to be eligible for excellence credit.
* For each group, only its last submission is kept and earlier submissions are discarded.
* Your code must be of good quality, and pass all test cases to earn excellence credits.
* Only the first 15 groups satisfying above conditions will be assigned excellence credits.

**Project Description**

Implement the algorithm discussed in class for finding minimum spanning trees in directed graphs (optimal branchings). The version that you need to implement was discussed in class as Tarjan's algorithm for optimal branchings, that was an improvement over the algorithms of Chu and Liu, and, Edmonds. A starter driver (LP3.java) is provided. Do not change the input format used by that program. It reads an integer corresponding to the root vertex, and then an input graph is provided in the format expected by Graph.readDirectedGraph(). You need to write the code for directedMST() in LP3.java. This function should return the weight of the MST it finds. In addition, the list dmst, passed as a parameter to this function, should be filled with the edges of that MST, in the order stated in the file LP3.java. It is recommended that you create separate classes for your code, and add just enough code to LP3.java to execute that code. When submitting the project, LP3.java must be placed in cs6301/gXX/, where gXX is your group name. Do not place LP3.java in some other subfolder. The output of LP3.java is already set up in the starter code. Do not change it.

**Excellence credits**

Your submission can get up to 10 excellence credits, if your code is one of the first 15 groups that submit good quality code before the first deadline, that correctly solves the problem, and passes all test inputs. An important criterion that will be used in judging quality is whether your solution is able to use standard algorithms for graph problems, such as DFS as-is, without rewriting them specially for this algorithm. Other aspects of quality such as OOP techniques, organization of objects and code, transparency of code, etc. will also be used. Programs selected for EC will also have to complete their runs successfully, within a timeout period. The best programs will complete within 10% of the allotted time.

**Input:**

**1**

**5 7**

**1 5 8**

**1 4 7**

**1 3 6**

**4 3 3**

**3 5 6**

**5 3 2**

**5 2 1**

**Output:**

**17**

**Additional lines of output with VERBOSE=1:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**null(5,2)(4,3)(1,4)(3,5)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Timer output**