CS2040C Semester 2 2021/2022

Data Structures and Algorithms

Tutorial+Lab 09

Graph Traversal

For Week 11

Document is last modified on: March 23, 2022

1 Introduction and Objective

Now that we have stored our graphs in one (or more – by now you should realize that you can do this) graph data structure(s), we want to run various (graph) algorithms on it.

In this tutorial, we will focus on two graph traversal algorithms: Depth-First Search (DFS) and Breadth-First Search (BFS) and concentrate on what they can do on top of just traversing the underlying graph.

We will heavily use `https://visualgo.net/en/dfsbfs` in this tutorial.

The tutorial part will be shorter to give more time for PE debrief (this session will be *after* PE on Tuesday, 29 March 2022, 6.30-8.30pm).

2 Tutorial 09 Questions

Review the Harder Topics

Q1. Tutor will spend some time (depending on the requests) to review any remaining harder topics about graph traversal that may not be clear even after Week 09+10 classes. In recent years, these are the usually harder topics for students, in decreasing order of difficulty:

1. `https://visualgo.net/en/dfsbfs?slide=7-10` to 7-11 (toposort, revisited below)
2. `https://visualgo.net/en/dfsbfs?slide=7-6` to 7-9 (finding connected components; check your understanding about the $O(V \times (V + E))$ versus just $O(V + E)$ analysis again)
3. `https://visualgo.net/en/dfsbfs?slide=8` (all other more advanced graph traversal topics that are not the main focus of CS2040/C are optional and such questions will be answered **offline**, after/outside class and will not be part of CS2040/C/S final assessment)

Deeper Stuffs about Topological Sort

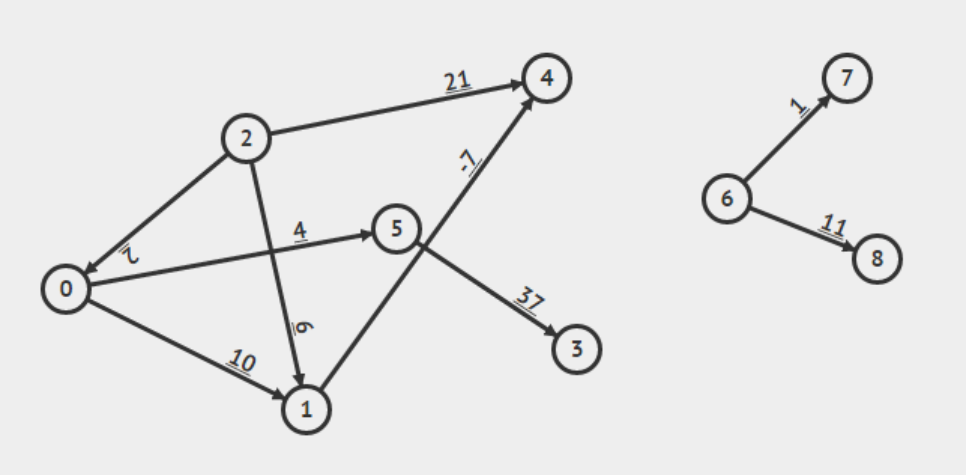

Figure 1: A Sample DAG (ignore edge weights for this question)

Q2. The modified DFS or modified BFS (Kahn's) topological sort algorithm given in class (please review `https://visualgo.net/en/dfsbfs`, 'topological sort', either the DFS or BFS version) only gives *one* valid topological ordering. How can we find **all** possible valid topological orderings for a given DAG? For example, there are **1008** possible valid topological orderings of the DAG in Figure 1. Starting point: What kind of DAG has the smallest/largest number of possible valid topological ordering, respectively?

Q3. The modified BFS (Kahn's) topological sort algorithm is actually quite interesting (read the details at `https://en.wikipedia.org/wiki/Topological_sorting#Kahn's_algorithm`). Can we change the underlying data structure (from a queue that is used in the modified BFS @ VisuAlgo) into another data structure? What if we replace the queue with a stack (`std::stack`)? What if we replace the queue with a priority queue (`std::priority_queue` or `std::set` – hopefully you now remember that this is also a valid ADT Priority Queue implementation)? What if we replace the queue with a hash table (`std::unordered_set`)?

Graph Modeling Exercise

Q4). If time permitting (if the clock is still around 30m into the tutorial because there is not much issues with in Q1), then the tutor will open Q4). and do 'live algorithm-level solve' for that (CS2040/C/S-level) problem.

Hands-on 9

TA will run the second half of this session with a few to do list:

- PS5 Debrief (A Quick One)
- PE Debrief

Problem Set 6

We will end the tutorial with **high-level** discussion of PS6.
As we are just starting, we will focuses more on understanding the tasks of PS6 A+B and discussing lower level subtasks of each task.