COMP20230: Data Structures & Algorithms
Assignment 2 Briefing

Dr. Mark Matthews

Office: A1.13
School of Computer Science
University College Dublin

mark.matthews@ucd.ie

Learning Outcomes

On successful completion of this project the learner will be able to:

- Run and assess the performance of various sorting algorithms
- Apply and run Dijkstra's shortest path algorithm and discuss your results.
- Identify the optimal bridge building solution to connect 7 islands using a Minimum Spanning Tree and Prim's Algorithm.

Assignment 2 - Part 1

Sorting Algorithms:

Run the Python code of the bubble, quick and merge sort algorithms given in labs and compute their running times to sort an input list A with x integer elements (integer values are in the range of 0 to x-1).

Please note that the input list A should be initially sorted in reverse (descending order) before running the sorting algorithm.
Continued..

Assignment 2 - Part 1

You need to implement following tasks:

1. Set up/implement 3 sorting algorithms
2. Create reverse order List inputs for your algorithms
3. Plot running times as a function of x for the three algorithms
4. Discuss the obtained results briefly in terms of time complexity of each algorithm and the initial distribution of the input data.

Continued..

Assignment 2 - Part 1

Sorting Algorithms:

You can justify your observations with the use of pseudocode as needed. As an example, consider that you have a list A with a size of x=10 integers arranged in reverse. List before sorting:
A=[9,8,7,6,5,4,3,2,1,0], x = 10
Then you need to sort A in an ascending order using bubble, quick and merge sort algorithms. Therefore, the output should be:
List A after sorting =[0,1,2,3,4,5,6,7,8,9], x=10
In other words, you just need to run bubble, quick, and merge sort algorithm for A and compute the running time of each algorithm. Repeat this procedure for different values of x, for example: x=10,15,20,30,..100. Plot the obtained running time as a function of x to see which algorithm is faster than others. Please save images of the graphs you output and include in your jupyter notebook. To get your input list arranged in reverse before running sorting algorithms, you can use reversed command in Python, i.e. A= list(reversed(A). To measure running time, use time.perf$_c$*ounter*()

Assignment 2 - Part 2

Shortest Path and Graphs

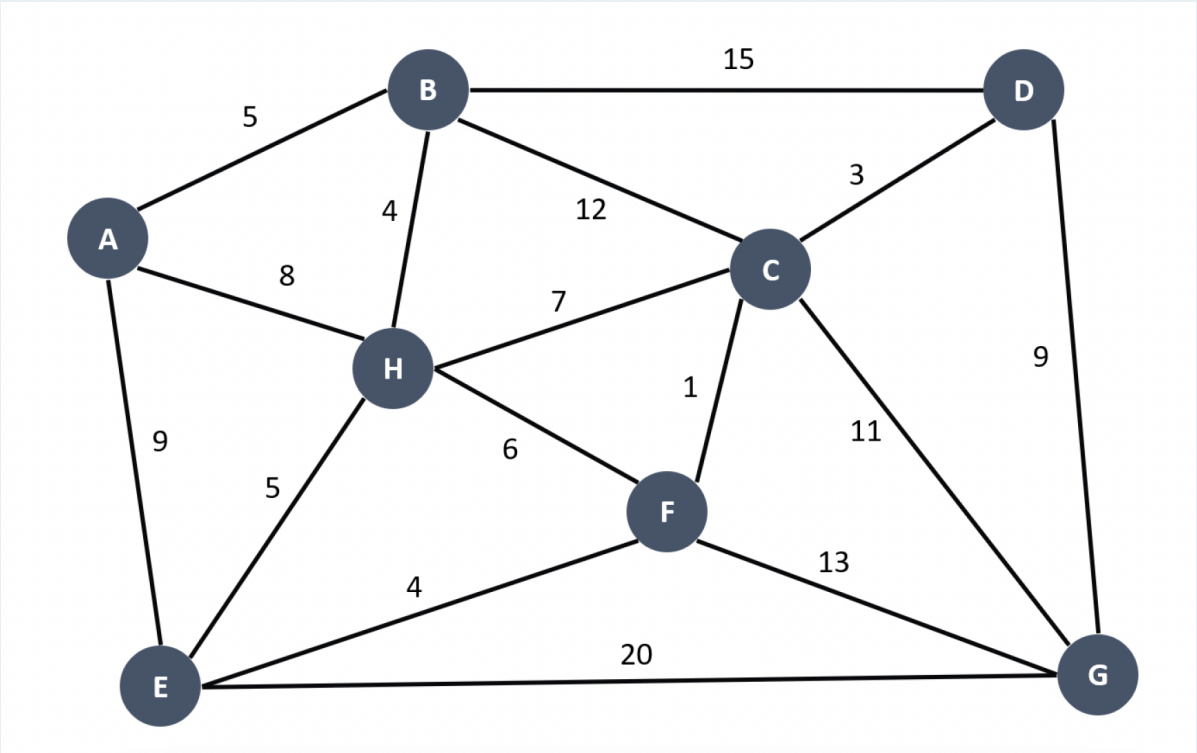
1. For the weighted graph shown in figure 1, apply Dijkstra's algorithm for finding the shortest distance from vertex A to vertex G.
2. You will step through Dijkstraâs algorithm, and write the values that are calculated by the algorithm.
3. Please indicate which nodes have been visited, what is the lowest total distance currently known for each node, and what is the best path known so far to each node.
4. Discuss your results briefly.

Continued..

Assignment 2 - Part 2

Part 2: Shortest Path and Graphs

Note: please use pen and paper to draw the graph after each step. You can use the drawing style given in Lecture 16: Graphs (2). You can fit all the plots in one paper if that suits you.



Continued..

Assignment 2 - Part 2

Part 2: Shortest Path and Graphs

1. Verify your answer of task (a.) using networkX Python package. Write Python code to create an empty weighted graph, G
2. Add vertices (nodes) with their weights given in figure 1 to G
3. Draw G. For the position (pos) parameter, you can adopt either the method which will compute a random layout for you, or a user-defined dictionary with node as keys and values as a coordinate list or tuple (please see Lab notebook for more details).
4. To compute all shortest paths and distances between vertex A and all other reachable vertices in the graph using Dijkstra algorithm. For each vertex, print a list that contains its shortest path with the minimum distance.
5. Draw a graph of shortest path tree (i.e. the same graph G but after removing all unwanted edges, the edges are not included in shortest path tree).

Assignment 2 - Part 3

Part 3:

There are eight small islands in a lake, and the state wants to build seven bridges to connect them so that each island can be reached from any other one via one or more bridges. The cost of constructing a bridge is proportional to its length. The distances between pairs of islands are given in the table below.

Island no.	1	2	3	4	5	6	7	8
1		240	210	340	280	200	345	120
2			265	175	215	180	185	155
3				260	115	350	435	195
4					160	330	295	230
5						360	400	170
6							175	205
7								305
8								

Continued..

Assignment 2 - Part 3

Part 3:

1. Convert the this problem into a minimum spanning tree (MST) problem, and solve it using Prims algorithm (starting with island 1 as the tree root node) to find which bridges to build so that the total construction cost is minimized.
2. For each island (node) in the priority path show the distance value and the minimum cost edge connecting it to the current tree.
3. Show also the edges in the tree after each edge is added.
4. Discuss the achieved results briefly.
   Note: you can follow the drawing style given in Lecture 17: Graphs (3) to answer this question. You can fit all the plots in one paper.

Assignment 2 - Part 3

Part 3:

1. Verify your answer of task (a.) using networkX Python package. Write a Python code to:
2. create an empty weighted graph, G
3. add vertices (nodes) with their weights given in table to G
4. draw G for the position (pos) parameter, you can adopt either the nx spring layout(G) method which will compute a random layout for you, or a user-defined dictionary with node as keys and values as a coordinate list or tuple (please see Lab notebooks for more details).
5. to compute the MST using Prims algorithm. As an output, print a list that contains the minimum spanning tree.
6. draw the MST graph (i.e. the same graph G but after removing all unwanted edges, the edges are not included in MST).

Your report should:

- Have proper structure with introduction, main body covering the various areas you address, and conclusions
- Have satisfactory technical coverage, balancing breadth of coverage with depth
- Have soundness of argument
- Have good clarity of expression and level of readability
- Clearly reference sources of information, and avoid plagiarism

Note Assignments submitted up to a week late will be docked 1 grade point, up to 2 weeks late will be docked 2 grade points. After that, assignments will be graded as pass/fail.

Assessment

Submission after week 12

- This assignment can be completed individually, or in groups of 2 or 3. Grading will be adjusted to account for the number of contributors.
- This assignment is worth 45% of the overall mark for COMP20230
- It should be uploaded to Brightspace by 5pm by Friday 29th April 2022.
- Submission: Code/Notebooks with embedded analysis of your solution.