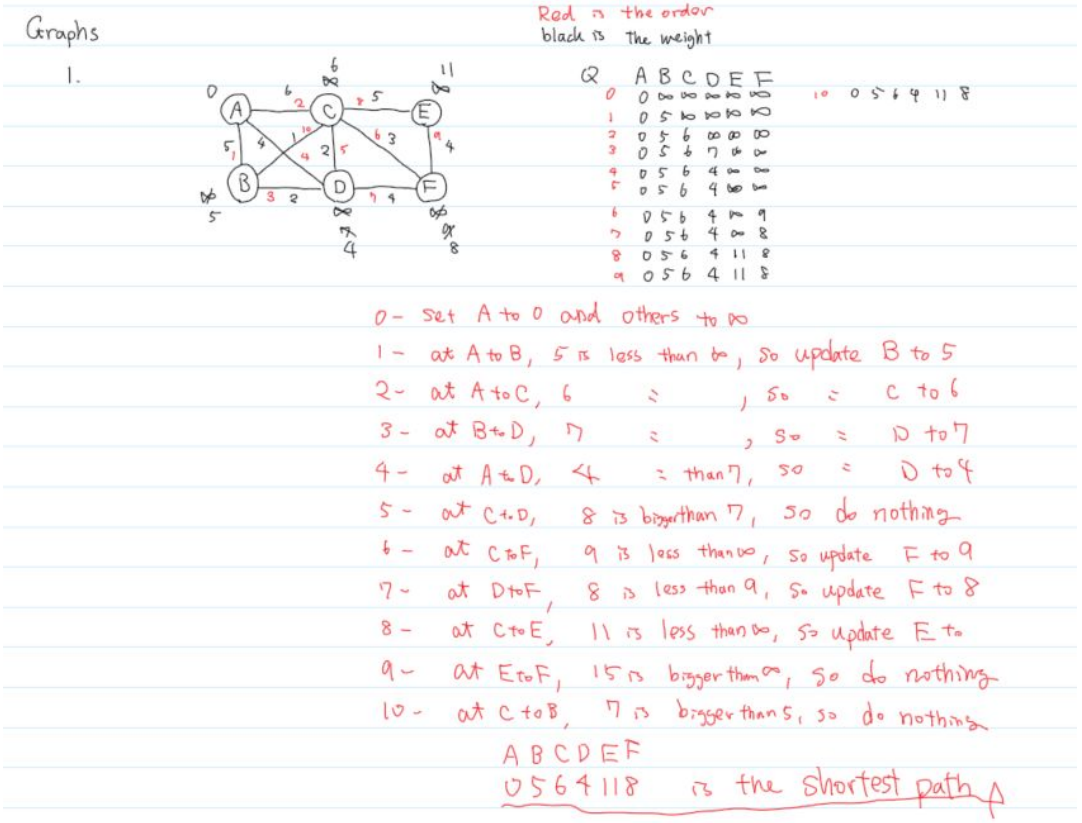
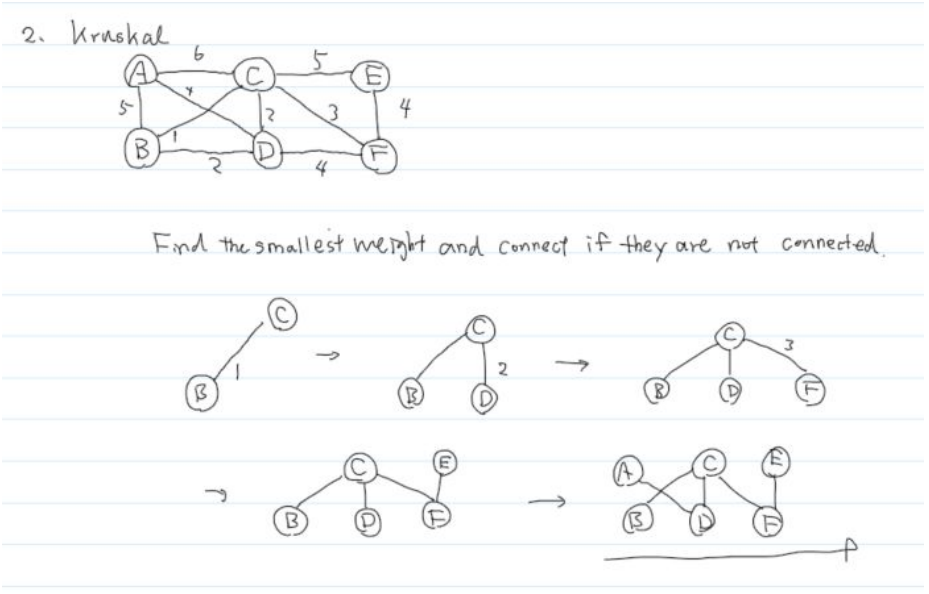
Graphs

1. Run Dijkstra's algorithm to find the shortest paths from node A to all other nodes.



2. Run an algorithm of your choice (Krusal) and find a minimum spanning tree



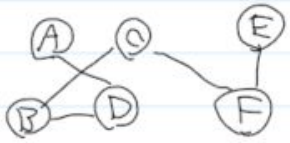
3. Is minimum spanning tree of this graph unique? Justify your answer if the answer is yes, provide a proof; if the answer is no, provide a counter-example and explain why this is the case

3.

The minimum spanning tree of this graph is not unique.

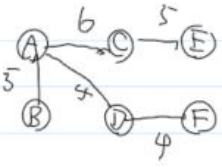
The minimum spanning tree is the tree that the sum of all edges is the smallest, so the shape can be different if the adjacent edges are the same. for example, in the graph, you can find another minimum spanning tree.

So the MST is not unique.



4. Consider the average distance from A to all other nodes, first by following edges on the shortest path tree (a), let's call it d_{SPT}^{avg}, and then following edges on the minimum spanning tree found in (b), let's call it d_{MST}^{avg}. Which one is greater?

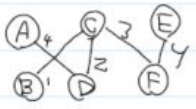
4. d_{SPT}^{avg} =



Sum = $d_{AB} + d_{AC} + d_{AD} + d_{AF} + d_{AE}$
= 5 + 6 + 4 + 8 + 11
= 34

average = $\frac{\text{Sum}}{\#\text{node} - 1} = \frac{34}{5} = 6.8$

d_{MST}^{avg} =



Sum = $d_{AB} + d_{AC} + d_{AD} + d_{AF} + d_{AE}$
= 7 + 6 + 4 + 9 + 13
= 39

average = $\frac{\text{Sum}}{\#n-1} = \frac{39}{5} = 7.8$

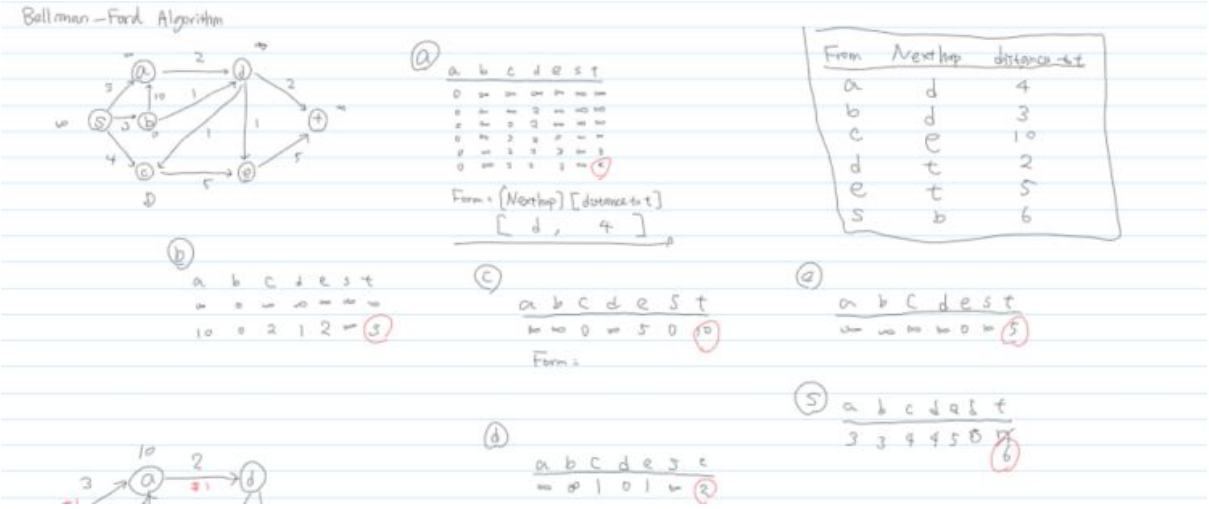
from above calculation, d_{MST}^{avg} is bigger.

Does the same answer hold for any graph G=(V, E) and node $A \varepsilon V$, or is it specific to this example?

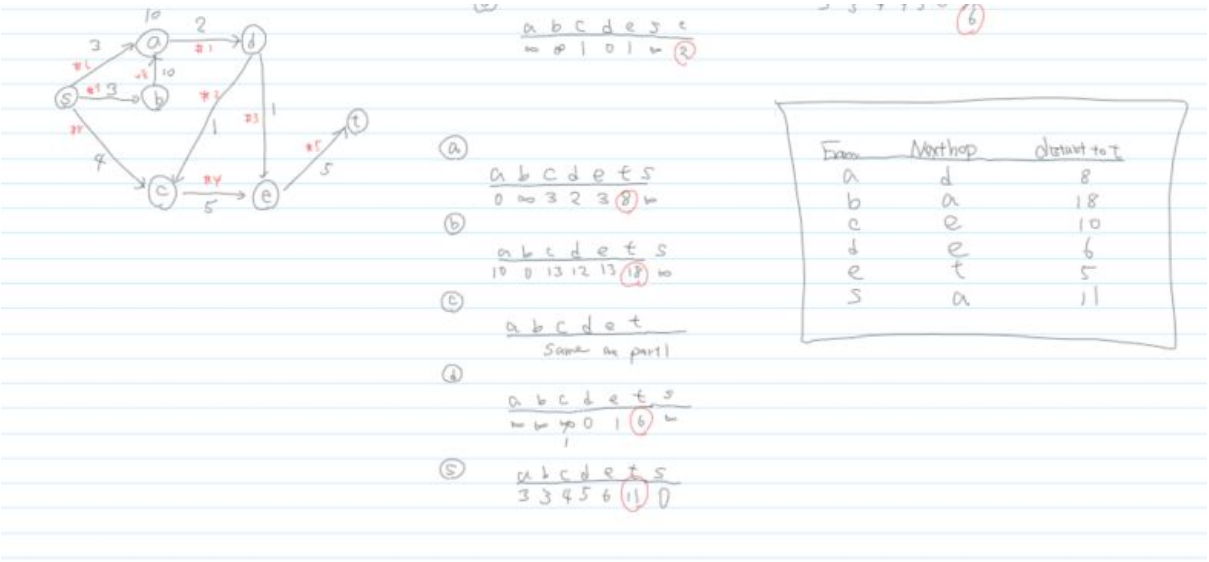
- Yes, $d_{spt}^{avg} \leq d_{mst}^{avg}$ always holds for any graph G=(V, E) and node $A \varepsilon V$. By the definition of the shortest path, the distance between one node to every other node should draw the shortest path tree.

Bellman-Ford Algorithm

1. Find the shortest paths from all nodes to destination t, using the Bellman-Ford algorithm. Show your intermediate steps and the final result in the following form: [next hop][distance to t] for every node.



2. After the algorithm reaches a steady state, somebody cuts off edges d-t and b-d at the same time. Show the computations following those failures and the new [next hop][distance to t] for every node.



Hashing

1. Open addressing hash table is implemented

Open Addressing (table size is 50)			
50%		Insert average reprobe	search average reprobe
	1	0.48	1.4914
	2	0.08	1.0436
	3	0.12	0.9644
	4	0.6	1.2605
	5	0.36	1.1088
	ave	0.328	1.17374

Open Addressing (table size is 50)			
50%		Insert average reprobe	search average reprobe
	1	0.48	1.4914
	2	0.08	1.0436
	3	0.12	0.9644
	4	0.6	1.2605
	5	0.36	1.1088
	ave	0.328	1.17374

2. Chaining hash table is implemented

Chaining Hash (table size is 50)			
50%		Insertion average chain	Search average chain
	1	0.04	0.0025
	2	0.04	0.0025
	3	0.04	0.0025
	4	0	0.0025
	5	0.04	0.0025
	ave	0.032	0.0025

Chaining Hash (table size is 50)			
90%		Insertion average chain	Search average chain
	1	0.111111	0.0025
	2	0.111111	0.0025
	3	0.133333	0.0025
	4	0.088889	0.0025
	5	0.155556	0.0025
	ave	0.12	0.0025

Prim’s algorithm

- Implemented
- The running time of two test graphs is in Typescript_prim
- The solutions are in solution_dens.txt and solution_sparse