4. Job Scheduling + MST

Tuesday, February 23, 2021 8:14 AM

Job	Profit	Slot
A1	100	5
A2	90	4
A3	80	3
A4	70	2
A5	60	1
A6	50	5
A7	40	4

1. Jobs = 0
2. Allocate j to slot 1 for all pending elements until the slot is filled

A1--> slot 0/1

A2:
X points to the largest filled slot --> slot 0/1
If S[X]>4(A2), then decrement x
Keep a count of the amount of time x is decremented

Job	Profit	Slot
A1	100	3
A2	90	5
A3	80	4
A4	70	2
A5	60	2
A6	50	1
A7	40	2

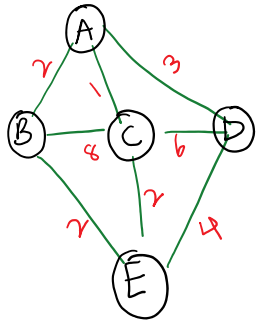
Complexity: O(n)
With dynamic programming: O(n^2)

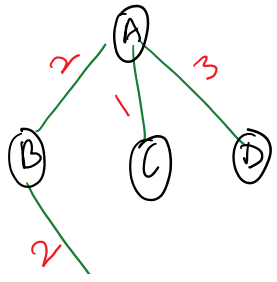
Minimum Spanning Tree with Prim's algorithm (Greedy algorithm)

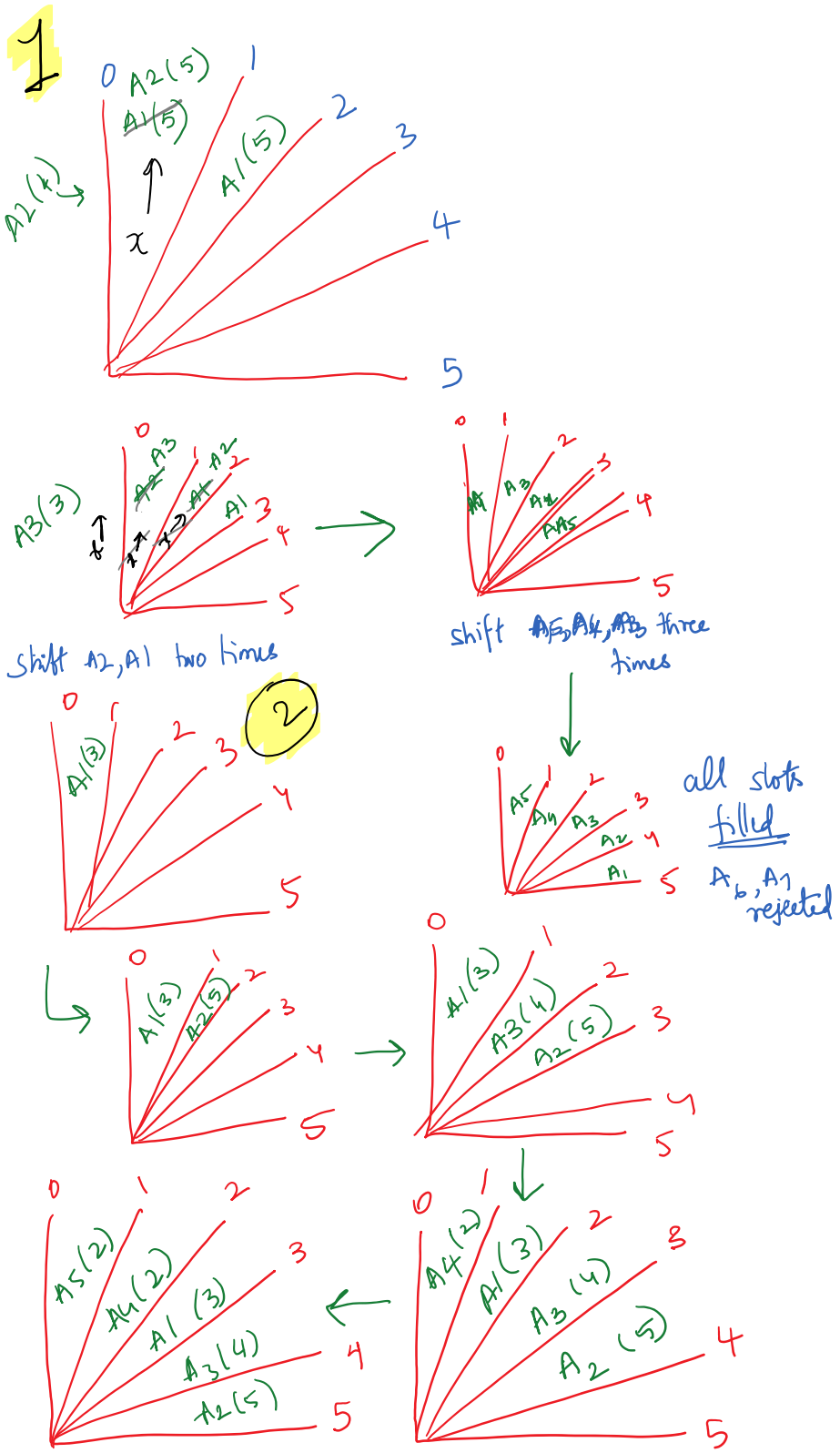
Given a graph G = {v,e} (undirected)
Objective: Find a tree that connects all the vertices in a graph without forming a cycle with a minimal cost of edges

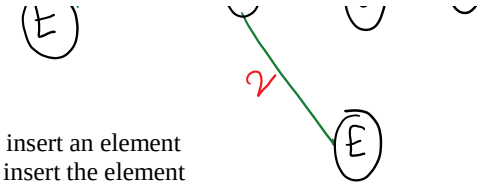
n nodes --> n-1 edges

Adjacency matrix: O(n^2)
Adjacency list: O(n)









Every time you insert an element into the graph, insert the element into a minheap to decrease complexity to O(nlogn)

i	Near(i)	Distance
1	1 ->0	
2	1	28
3	6	inf
4	6	inf
5	**6**	**25**
6	6 -> 0	
7	6	inf

If i==near(i), cost(i)=0

Check 2,3,4,5,7 nearer to 1 and 6?

If(dist(i,1)<dist(i,6))
Add 1
Else
Add 6

If both are infinity, 6 gets added

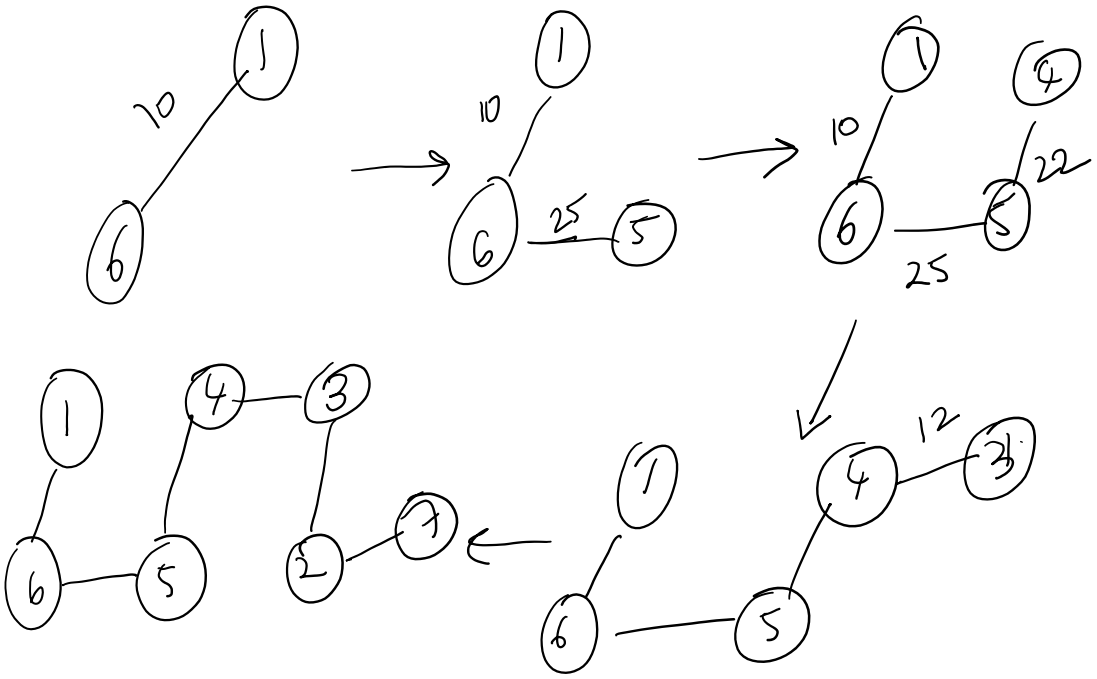
1	0	
2	1	28
3	5	inf
4	**5**	**22**
5	0	
6	0	
7	5	24

Check 2,3,4,7 nearer to 1,5,6?

Since 2 is nearer to 1 in the last step, only compare with 1 and 5

1	0	
2	1	28
3	**4**	**12**
4	0	
5	0	
6	0	
7	4	18

1	0	
2	**3**	**16**
3	0	
4	0	
5	0	
6	0	
7	5	18



1	0	
2	3	16
3	0	
4	0	
5	0	
6	0	
7	**2**	**14**