Assignment - 2

Greedy Algorithms

⇒ Problem Statement : Implement Greedy algorithms for following problems:
a. Job Scheduling Problem   b. Dijkstra's single source shortest path
c. Prim's Algorithm for creating MST   d. Kruskal's algorithm for creating MST

⇒ Theory : The greedy method is the simplest and straightforward approach. The main function of this approach is that the decision is taken on the basis of currently available information. Examples are :

1) Job Scheduling Problem : Given an array of Jobs where every job has a deadline and associated profit. It is also given that every job takes a single unit of time, How to maximize total profit if only one job can be scheduled at a time.

• Algorithm :
- Sort the jobs based on their deadline.
- Iterate from the end and calculate the available slot between every two consecutive deadline. Include profit, deadline, job ID in max heap.
- While the slots are available and there are jobs left in max heap, include job with max profit in the result.
- Sort the result array based on deadline of jobs.

2) Dijkstra's single source shortest path problem : It is an algorithm for finding the shortest path between nodes of a graph. For a given source node, it finds the shortest path b/w that node & every other node.

• Algorithm :
- We need to maintain path distance of every vertex. That can be stored in an array of size v, v being the number of vertices.
- We also want to be able to get the shortest path, not only know length of shortest path. We map each vertex to vertex that last updated its path.
- Once the algorithm is over, we can backtrack from destination vertex to source vertex to find the path. A min heap can be used to efficiently receive the vertex with least path distance.

3) Prim's Algorithm for creating MST : Prim's Algorithm starts with single node and work its way through several adjacent nodes, exploring all the connected edges along the way. Edge with min weights, not forming cycle in graph gets selected for inclusion in MST.

- Algorithm :

1 - Determine the arbritary start vertex.
2 - Keep repeating steps 3 & 4 unitl fringe vertices (vertices not included in MST) remain.
3 - Select an edge connecting tree vertex and fringe vertex having min wt.
4 - Add the chosen edge to MST if it doesn't form a cycle.
5 - Exit

4) Kruskal's Algorithm for creating MST : Kruskal Algorithm sorts all edges in increasing order of their weights and keeps adding node to the tree only if chosen edge doesn't form a cycle. It also picks the edge with min. cost at first & edge with max cost at last, thereby making a locally optimal choice, intending to find the global optimal solution.

- Algorithm:

- Sort all edges in increasing order of their edge weights
- Pick the smallest edge (min. weight)
- Check if new edge creates a cycle in spanning tree
- If it doesn't form a cycle, then include that edge in MST.
- Repeat above steps until it includes |V-1| edges in MST.

⇒ Conclusion : Implemented various greedy algorithms.