Analysis of Algorithms

Assignment 7

Protik Dey

Ans: to the Ques: No: 1 (a)

If the activity sequence is like (0, 5), (6, 10), (4, 7), & (10, 20), the algorithm will select (4, 7) and (10, 20) which is not optimal. The optimal solution is (0, 5), (6, 10), (10, 20). Here least amount of active time fails.

Ans: to the Ques: No: 1 (b)

```
greedy-sol (s, f){
    A = {a1}        //last started activity
    K = 1
    for i := 2 to len(s)
        if f(i) ≤ s(k)
            A = A ∪ {am}
            K = m
    return A;
}
```

Proof of correctness: Let's considere any non-empty subproblem S_k and let a_k be the activity in S_k with the last start time. Then a_k is in some optimal solution of S_k.

Proof: Let A_k be an optimal solution of S_k and let a_i be the activity in A_k with last start time. If $a_k = a_i$ then our proof is done.

Let's suppose $a_k \neq a_i$. Let considere $A_k' = (A_k \cup \{a_k\}) \setminus \{a_i\}$

Activities of A_k have a finish time earlier than the start time of a_i and a_k has a start time greater than the finish time of all activities of A_k. So $|A_k'| = |A_k|$

So A_k' is optimal and hence proved that the algorithm is correct.

(Proved)

Ans: to the Ques: No: 2

There are six valid topological sorts:

i) a b c d e f
ii) a b d c e f
iii) a b d e c f
iv) a d e b c f
v) a d b c e f
vi) a d b e c f

—

Ans: to the Ques: No. 3

We can use DFS to find a cycle in graphs. A cycle exists when a back edge is present. We can modify the DFS to determine whether or not an undirected graph contains a cycle.

```
modified_DFS (G){

    for each vertex u ∈ G
        color[u] = white;
        parent[u] = null;
    for each vertex u ∈ G
        if color[u] = white
            cycle_exist(u);

}
```

```
cycle_exist (u){
    color[u] = gray
    iscycle = false
    for each v ∈ adjacent of u
        if (color[v] = white)
            parent[v] = u
            iscycle = cycle_exist (v)
        else if (color[v] = gray &&
                 parent[u] ≠ v)
            return true;

    color[u] = black
    return iscycle
```

Here the basic functionality is same as DFS. So the running time is $O(n+m)$ where n = number of vertices, m = number of edges

—

Ans: to the Ques. No: 4

Let T_1 and T_2 be two distinct MST of G. Since $T_1 \neq T_2$, there exists at least one edge e such that $e \in T_1$, $e \notin T_2$ or $e \notin T_1$, $e \in T_2$.

Let's assume that $e \in T_1$ but $e \notin T_2$. So adding e to T_2 will create a cycle, C. Let e' be an edge in the cycle C.

$cost(e) \neq cost(e')$ as edge costs are distinct. Let's consider e is the heaviest edge of the cycle. For any given cycle, an MST will not use the heaviest edge. So e does not belong to T_1. So it contradicts our assumption that both T_1 and T_2 are MST.

So a connected graph G has a unique MST if all the edge costs are unique.

(Proved)

—