Example Answers of Data Structures and Algorithms

(Software School, Fundan University, winter, 2009 - 2010)

Questions (100 points)

1. Amortized cost

PUSH 2
POP 0
BUCKUP 0

2. Topological sort

Step 1: Call depth first search algorithm.
Step 2: Sort by finishing time.
Run time: $O(nlgn)$
Yes, it can be used to solve the single-source shortest path problem for a DAG with negative edge path costs because DAG is a directed acyclic graph.

3. Find the minimum spanning tree for every connected, undirected graph

Count the number of edges that appear in the minimum spanning trees, denoted by x. The minimum number of edges that need to be removed from an undirected graph that the resulting graph is acyclic is $E - x$.

4. Two key ingredients

Optimal substructure: Dynamic programming builds an optimal solution to the problem from optimal solutions to subproblems. The solutions to the subproblems used within the optimal solution to the problem must themselves be optimal by using a "cut-and-paste" technique. Subproblems should be independent.
Overlapping subproblems: Recursive algorithm revisits the same problem over and over again. In contrast, a problem for which a divide-and-conquer approach is suitable usually generates brand-new problems at each step of the recursion.

For greedy algorithm, we should prove that at any stage of the recursion, one of the optimal choices is the greedy choice. Thus, it is always safe to make the greedy choice.
Greedy-choice property: A globally optimal solution can be arrived at by making a locally optimal choice (a greedy choice at each step yields a globally optimal solution).

5. Write pseudocode for Prim's algorithm

MST-PRIM(G, w, r)
1. **for** each $u \in V[G]$
2. **do** $key[u] \leftarrow \infty$
3. $\pi[u] \leftarrow$ NIL
4. $key[r] \leftarrow \emptyset$
5. $Q \leftarrow V[G]$
6. **while** $Q \neq 0$

```
7.      do u ← EXTRACT-MIN(Q)
8.        for each v ∈ Adj[u]
9.          do if v ∈ Q and w(u, v) < key(v)
10.                then π[v] ← u
11.                     key[v] ← w(u, v)
```

MAKE-SET(x)

1. $p[x] \leftarrow x$
2. $rank[x] \leftarrow 0$

FIND-SET(x)

1. **if** $x \neq p[x]$
2. **then** $x \leftarrow$ FIND-SET($p[x]$)
3. **return** $p[x]$

UNION(x, y)

1. LINK(FIND-SET(x), FIND-SET(y))

LINK(x, y)

1. **if** $rank[x] > rank[y]$
2. **then** $p[y] \leftarrow x$
3. **else** $p[x] \leftarrow y$
4. **if** $rank[x] = rank[y]$
5. **then** $rank[y] \leftarrow rank[y] + 1$

6. Answer questions

a. This algorithm will not work for cyclic graphs

b. The algorithm is equal to DFS.

c. $O(V + E)$

7. Find all-pairs shortest paths

$$D = \begin{bmatrix} 0 & 3 & 7 & 2 & 9 \\ 1 & 0 & 8 & -1 & 6 \\ 0 & 3 & 0 & -4 & 5 \\ 4 & 7 & 11 & 0 & 9 \\ -5 & -2 & 2 & -3 & 0 \end{bmatrix} \quad \pi = \begin{bmatrix} \varnothing & 1 & 1 & 2 & 2 \\ 5 & \varnothing & 5 & 2 & 2 \\ 5 & 1 & \varnothing & 3 & 4 \\ 5 & 1 & 5 & \varnothing & 4 \\ 5 & 1 & 5 & 2 & \varnothing \end{bmatrix}$$

8. Fill the blanks

(*a*) $q > 0$

(*b*) $q \leftarrow \pi\,[q]$

(*c*) $q \leftarrow q + 1$

(*d*) $q = m$

(*e*) $q \leftarrow \pi\,[q]$

9. Minimum cost flow

Path: $s \to 2 \to 1 \to t$, flow: 5, cost: 4, total: 20.

Path: $s \to 1 \to t$, flow: 2, cost: 5, total: 10.

Path: $s \to 2 \to 3 \to t$, flow: 3, cost: 6, total: 18.

Total cost: 48.