Fall 2012, Second Test, Introduction to Algorithms

Name:
Section:
Email id:

9th April, 2015

This is an open books, open notes exam.
Calculator is allowed, Internet is not allowed.
Answer all six questions. Each Question is worth 10 points. You have 90 minutes to complete the exam.

Sample Solution.

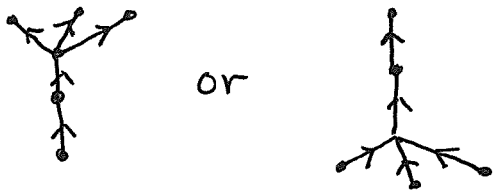
1. **Directed Graph**

(a) Assume that all nodes v are initialized with v.pre=-1 and v.post=-1. In doing a DFS on a directed graph, nodes get pre number when it gets visited first and post number when it gets visited last. If you encounter an edge (u,w) when you are visiting u, give pseudo code to check whether the edge (u,w) is a forward edge or not. [**3 Points**]

pre(u) < pre(w)

post(w) > 0

(b) Construct a directed acyclic graph with 6 nodes which has exactly 6 (3! = 6) topological orderings. [**3 Points**]



(c) You are given a directed graph and a vertex v. Describe an efficient algorithm to find whether every vertex can reach v. (If more than one BFS/DFS is done you will get half the credit). [**4 Points**]

1. Reverse the orientation of each edge.
2. Do a BFS starting at v. if every vertex is having a pre-number then "every vertex can reach v" else "not every vertex can reach v.

2. **Minimum Spanning Tree**

(a) You are given an undirected graph. Suppose all edges in a graph have distinct weights and has more than one spanning tree. Describe an efficient algorithm to find a spanning tree whose cost is between the smallest weight spanning tree and the largest weight spanning tree. [**4 Points**]

Sort 1. For each edge e, test whether e is a bridge. if it is a bridge, set x[e] = 1.

2. Sort the edges in increasing order of weights. Delete the first edge e_1 in that list, such that Bridge(e) ≠ 1 and there is another edge e_1 such that Bridge(e_1) ≠ 1 and $w(e_1) < w(e)$.

3. Find the MST on the resulting list.

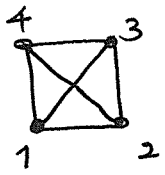
(b) Will the minimum spanning tree algorithm work with negative weight edges - Give a brief explanation. [**2 Points**]

Yes. Kruskal's algorithm selects edges from a list of sorted (based on) the weights of edges. It does not care about negative weight edges.

(c) You are given a complete undirected graph with four vertices with vertices $\{1, 2, 3, 4\}$, and six edges $\{(1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4)\}$. We know that it has 16 span-

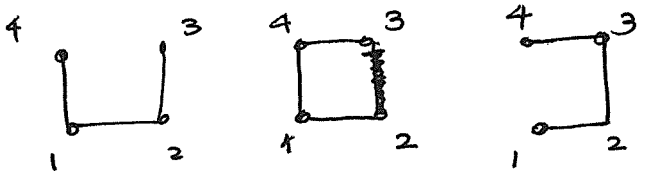
ning trees (From Prufer Sequence). Draw all the spanning trees contain an edge (1,2). [4 **Points**]

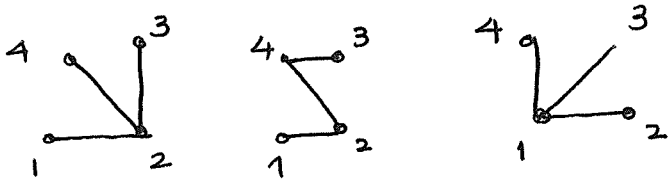
8 trees.

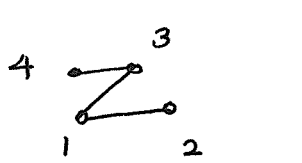


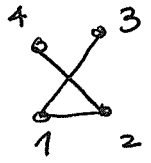
$\left(\frac{16 \cdot 3}{6} = 8 \right)$

16 sp. trees
Each sp. Tree has 3 edges.
Graph has 6 edges.



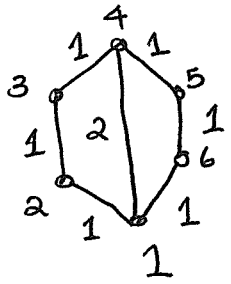




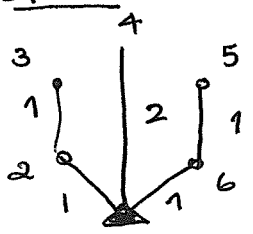


3. **Minimum Spanning Tree /Strongly Connected Component(DFS on a Directed Graph**

(a) Give an example of an undirected graph (with 6 vertices and 7 edges , 6 edges have weight 1 and 1 edge has weight 2) such that the weight of the shortest path tree (using a starting vertex of 1) is not the same as the weight of the minimum spanning tree [**5 points**]



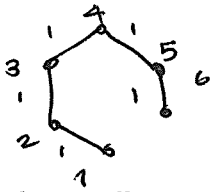
Shortest path tree



Cost 6

Starting vertex

Spanning tree



5
cost

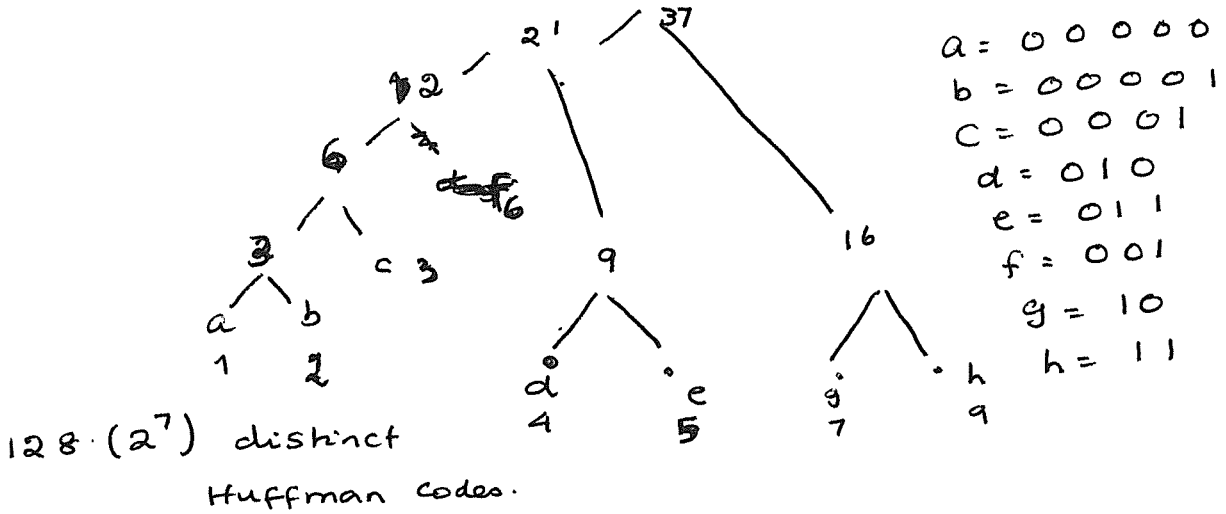
(b) Describe an efficient algorithm in linear time (in size of vertices and edges)(Pseudo Code will suffice) to find, a given directed graph G and an edge $< v, w >$, whether there is a directed cycle containing edge $< v, w >$.[**5 points**]

1. Find the strongly connected component of G.
2. If u and v belong to the same strongly connected component, then u and v lie in a directed cycle.
   [There is a path from v to u. This path along with edge (u, v) will form a cycle.].

   If u and v do not belong to the same strongly connected component, then they do not belong to the same cycle.

4. **Greedy Algorithm and Huffman Tree**

(a) Construct a Huffman Tree with the following characters a,b,c,d,e,f,g,h and their respective frequency of occurrences are 1,2,3,4,5,6,7,9. Write down the codes for a,b,c,d,e,f,g,h from the constructed Huffman tree. How many distinct Huffman Codes for this data are possible (just the number) (For instance, if the character set consists of two symbols [A, B] only, they can be coded as either [0, 1], or [1, 0]. So the number of Huffman codes is 2 in this case.). [**5 Points**]



(b) You are given a set with n of distinct elements. Describe an efficient algorithm to find a subset of $n/2$ elements whose sum is the largest. (Linear time will get you full points.)
[**5 points**]

1. Find the median set of the set S. Let it be y.
   (can be done in linear time).
2. Let X = Φ
3. For each element p in the given set,
   if p <= y, insert p in X.

X is the desired set.

Time complexity is O(n). linear.

5. **Dynamic Programming**

Give an $O(nt)$ algorithm for the following task (A pseudo code will suffice.)
Input : A list of k distinct coins of positive integers $a_1, a_2, \cdots, a_n$; a positive integer t.
Question : What is the largest number of coins that add up to t. (that is sum of the coin values should add up to t and the number of coins should be as large as possible). You are allowed to use as many times as possible a coin of any denomination. The output is a number. First write the recurrence relations before writing a pseudo code. Let $K(i)$ be the maximum number of changes for coin i.

(a) Define $K(0)$ and the initial values of $K(i)$, $0 < i \leq t$

(b) Define $K(y)$ in terms of $K(i)$ $0 \leq i \leq y$

(c) Write a pseudo code.

[10 Points]

a. $K(0) = 0$

$K(i) = -\infty$ for $i = 1$ to t.

b. $K(y) = \max_{1 \leq i \leq n} \left(K(y), \; K(y - a_i) + 1 \text{ if } y - a_i \geq 0 \right)$

c.

```
K(0) = 0
for i = 1 to t
K[i] = -∞
 for i = 1 to t
    for j = 1 to n
      if a[j] >= i
          if K(i - a[j]) + 1 > K(i)
              K(i) = K(i - a[j]) + 1 .

return K(t)
```

6. **Shortest Path and Greedy Algorithm**

(a) You are given a strongly connected directed graph with positive weights and a specified vertex v_0 (v_0 is one of the vertices of the given strongly connected graph). Describe an efficient algorithm to calculate shortest path between all pairs of vertices with condition that the shortest paths should pass through the given vertex v_0. [5 Points]

1. Find the shortest path from v_0 to vertex 0.
1.5 Reverse the arrows of the edges of the graph.
2. Find the shortest path from v_0 to every vertex.
2.5 Let us call the distance computed as $SP[i, v_0]$
3. Shortest path from any vertex i to any vertex j is calculated as $SP[i, v_0] + SP[v_0, j]$

(b) Suppose there are n activities which takes time $t_1, t_2, t_3, \cdots, t_n$ to complete. Describe a greedy algorithm to complete maximum number of activities within a given time T. [5 Points]

1. Sort the activities according to time it takes from lowest to highest.
2. Keep adding activities $t_{\pi(k)}$ one at a time such that $t_{\pi(1)} + t_{\pi(2)} + \cdots t_{\pi(k)} <= T$ and $t_{\pi(1)} + t_{\pi(2)} + \cdots t_{\pi(k+1)} > T$.

return k as the answer.

```
Sort the activities
sum = 0 ; j = 0
  for i = 1 to n
     if Sum + ti <= T
          Sum = sum + ti
          j = j+1

return j
```