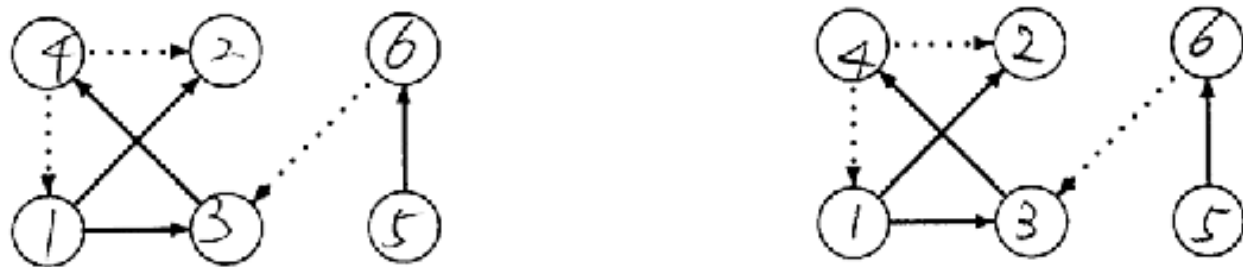


Cs170 - Exam 2
March 13, 1996
Prof. David Wolfe

1. (40 points)

For each of the following questions, short-answer questions, no explanation is necessary. No partial credit is possible, so recheck your answer. (Note: I avoid trick questions.)

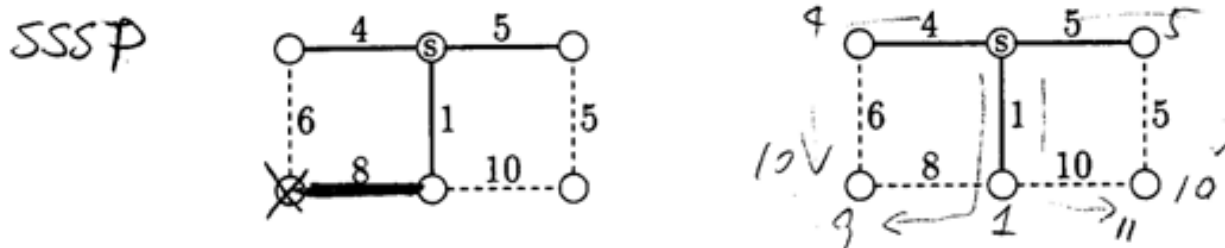
(a) (25%) Depth-first search is run on the following graph, G , obtaining the DFS tree shown by the solid edges. (The dotted edges indicate non-tree edges in G .) Indicate the preorder number (or, as Manber calls it, the DFS-number) for each vertex by writing the numbers



(The second copy is for your convenience.)

[figure 1.a]

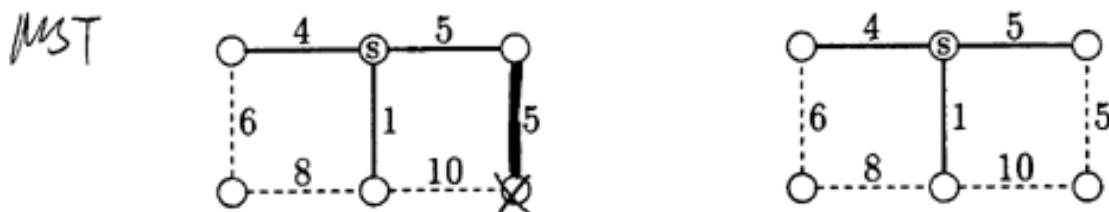
(b) (25%) Dijkstra's algorithm is being run to find shortest paths from source s in the following undirected graph. The graph is shown in dashed edges, and the tree so far is solid. Darken the one dashed edge which would next be added to the tree:



(The second copy is for your convenience.)

[figure 1.b]

(c) (25%) Prim's algorithm is being run to find the minimum spanning tree on the same graph, growing the tree from vertex s . Darken the one dashed edge which will next be added to the tree:



(The second copy is for your convenience.)

[figure 1.c]

(d) (25%) The following idea is used for an algorithm which checks if the product of two polynomials is correct:

Input: Three polynomials of degree at most d , $p_1(x)$, $p_2(x)$ and $p_3(x)$.

Problem: Determine if $p_1(x)p_2(x)=p_3(x)$ for all real values of x .

Algorithm sketch: plug in a random value for x_0 from $\{1, \dots, 2d\}$, and evaluate the three polynomials at x_0 . Output “equal” if and only if $p_1(x_0)p_2(x_0)=p_3(x_0)$.

Is the algorithm Monte-Carlo, Las-Vegas or neither? _____

2. (20 points)

Give a linear-time algorithm to find all the longest paths from a single source s in a directed acyclic unweighted graph. (Hint: I can think of two algorithms. One uses topological sort, the other uses a shortest path algorithm.)

3. (20 points)

A ski rental agency has m pairs of skis, where the height of the j^{th} pair of skis is s_j . There are $n < m$ skiers who wish to rent skis, where the height of the i^{th} skier is h_i . Each skier wants a pair of skis whose height matches her own height as closely as possible. The goal is to assign the skis to the skiers so that the sum of the absolute differences of the heights of each skier and her skis is minimized.

In the following example, the algorithm should output the optimum value 5:

Input		Optimum matching	
Skiers	Skis	Match	Cost
$h_1=60$	$s_1=59$	h_1 with s_1	$ 60-59 =1$
$h_2=67$	$s_2=62$	h_2 with s_3	$ 67-70 =3$
$h_3=70$	$s_3=70$	h_3 with s_4	$ 70-71 =1$
	$s_4=71$		
Output: Total cost of optimal			5

(a) (0% - DO NO ANSWER) Prove that in the optimal assignment of skier to skis, the taller skier always gets the taller pair of skis. In other words, prove that if skier A is taller than skier B then in the optimal assignment, skier A's skis are taller than skier B's skis.

[You don't have to do this part, assume its been done]

(b) (50%) Assume the height of the skis and skiers are sorted as in the example. Define

$C_i^j = \begin{cases} \text{The minimum assignment cost if the first } i \text{ skiers are assigned skis} \\ \text{among the first } j \text{ pairs (and no other skiers are assigned skis).} \end{cases}$

Note that C_i^j only makes sense if $i \leq j$. In the example,

$$C_2^3 = (|h_1 - s_1| + |h_2 - s_3|) = 1 + 3 = 4$$

$$C_3^3 = (|h_1 - s_1| + |h_2 - s_2|) + |h_3 - s_3| = 1 + 5 + 0 = 6$$

Give a recurrence for C_i^j in terms of C_{i-1}^{j-1} , C_i^{j-1} and/or C_{i-1}^j .

(c) (50%) How long does it take to evaluate C_m^n by dynamic programming? The running time should be in terms of m and n . Briefly explain your reasoning.

[figure 3]

4. (20 points)

A tournament is directed graph with exactly one dge between every pair of vertices. In other words, to get a tournament, take a complete undirected graph and direct each edge. Show that every tournament has a Hamilton path. (recall that a Hamilton path is a path which includes each vertex exactly once. It will not be a cycle.)

For example, for the tournament on four vertices shown on the left, I have drawn a possible Hamilton path on the right:



[figure 4]

Big hint: one way to being a proof is:

“Let v be any vertex in tournament G . Partition the vertices of G into three sets, $\{v\}$, S and T , where S is the set of vertices in G which point to v , and T is the set of vertices which v points to.