Solutions to Final

Solutions to Final

May 18th 1995

CS170 David Wolfe

1. (30 points) Check one box for each of the following.

	ALWAYS TRUE	SOMETIMES TRUE	NEVER TRUE
In a connected graph, E=O(V)		X	
In a connected graph, E=Ω(V)	X		
If a directed graph has a depth first forest with no back edges then the graph has n strongly connected components	X		
For a fixed graph G, the Edmonds-Karp algorithm does at least as well as Ford-Fulkerson		X	

	TRUE	FALSE
Log*(m) = O(α(m,n))		X
For data encryption to be secure against eavesdroppers, two parties must meet in a secure environment to agree on secret keys.		X
A heap can be designed which takes O(n) per INSERT and O(1) per DELETE-MIN	X	
A heap can be designed which takes O(1) per INSERT and O(1) per DELETE-MIN		X

In a Fibonacci heap, we could assign $O(1)$ amortized cost per DELETEMIN and per DELETE, as long as we assign $O(\log n)$ amortized cost per INSERT.	X	

You lost 4 per incorrect answer. If you left "log*(m)= $O(\alpha(m,n))$" blank, you lost only 2. (Many students didn't see this question.)

2. (15 points) Alice is deciding between the pseudo-prime test and the randomized primality test.

a. What is the strongest reason you can think of to use a pseudo-prime test?
b. What is the best reason you can think of to use the randomized pirmality test?

a. The pseudo-prime test is good mainly because it is faster.

b. By repeating the randomized test many times, Alice has complete control over the probability of success, no matter what number she is testing.

3. (15 points) Hubert is deciding between using binary heaps or Fibonacci heaps.

a. What is the strongest reason you can think of to use binary heaps?
b. One good reason for Fibonacci heaps?

a. Binary heaps are simpler, they have guaranteed (rather than amortized) running time for each operation and in practice they work faster since n isn't really that big.

b. Fibonacci heaps have a faster amortized running time in the limit as n -> infinity

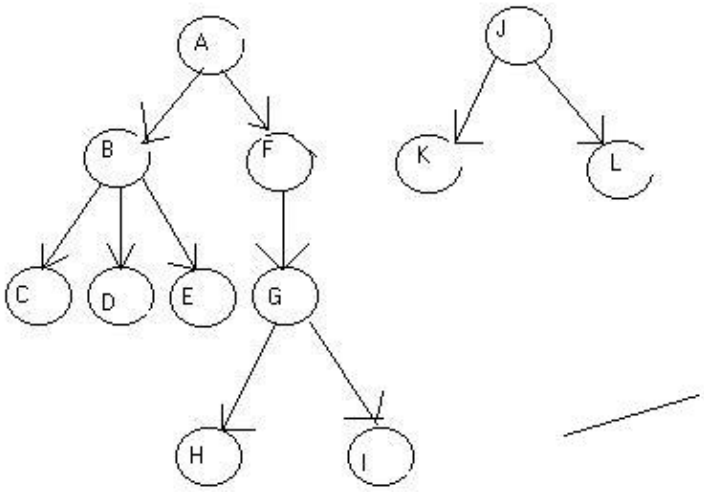
4. (30 points) A depth first search is performed on a graph whose vertices are V = {A, B, C, L}. The vertices listed in order of their preorder-numbers (or in order of their discover times) is:

{ *A, B, C, D, E, F, G, H, I, J, K, L* }

The vertices listed in order of their postorder numbers (or finish-times) is

{ *C, D, E, B, H, I, G, F, A, K, L, J* }

Draw all the trees in the depth-first search forest.



5. (30 points) Let G = (V, E) be a flow network with source s, sink t, and suppose each edge e $\in E$ has capacity c(e) = 1. Assume also, for convenience, that $|E| = \Omega(V)$.

a. Suppose we implement the Ford-Ffulkerson maximum-flow algorithm by using depth-first search to find augmenting paths in the residual graph. In terms of V and E only, what is the worst case running time of this algorithm on G?
b. Suppose the maximum flow for G has been computed, and a new edge e with unit capacity (i.e., $c(e) = 1$) is added to E. Describe how the maximum flow can be efficiently updated. (Note: It is not the value of the flow that must be updated, but the flow itself.) Analyze your algorithm.

See exam 3 solution

6. (30 points) Suppose that we are given n currencies, c_1,,c_n, and an n ×n table R of exchange rates such that one unit of currency c_I buys $R[i , j]$ units of currency c_j. Give an efficient algorithm to determine whether or not there exists a sequence of currencies c_{i1}, c_{i2}, , $c_{i}n$ such that

$$R$$

$$[i_1 , i_2] \cdot R[i_2 , i_3] \cdot \cdots R[i_{k-1} , i_k] \cdot R[i_k , i_1] > 1$$

(This would mean that there is an arbitrage opportunity to turn one unit of c_1 into more than one unit by just exchanging currencies.) Analyze the running time of your algorithm.

See homework set 9 solutions

7. (40 points) For this problem, all answers should be actual numbers (not just expressions), and you should show your work. No calculator is permitted. You will not lose credit for a little arithmetic error as long as I can follow your work.

Bob wants to send Alice the message M = 5 using RSA. Bob knows Alice's public key is $(e, n) = (7, 33)$.

a. (10 points) What other public keys could Alice have chosen given her choice of n=33?
b. (10 points) What is the encrypted message that Alice receives?
c. (15 points) What is Alice's secret key that she uses for decryption.
d. (5 points) Decrypt Bob's message as Alice would using her secret key.

If, after all calculations, you think you have made an arithmetic error in parts a-c, admit why you think you have, but don't spend time correcting the error.

a. She could have used any value of d ∈ {1, 3, 7, 9, 11, 13, 17, 19}, since these are relatively prime to $\phi(n) = 20$.
b. Since $3 \cdot 7 = 21 \equiv 1 \pmod{n}$, $(n, d) = (33,3)$. If you used the gcd algorithm to compute d, you could values so that $\phi(n) \cdot ? + d \cdot ? = 1$. The second "?" mod $\phi(n)$ is e.

```
GCD Calculation Extended Euclid's algorithm

(20, 7) 20 · −1 + 3 · 7 = 1 ß so d = 3

(7,6) 7 · 1 + 6 · −1 = 1

(6,1) 6 · 0 + 1 · 1 = 1
```

c. All of the following equations are true (mod 33). Bob computes M^e or

$$5^1 \equiv 5$$

$$5^2 \equiv 25 \equiv -8$$

$$5^4 \equiv (-8^2) \equiv 64 \equiv -2$$

$$5^7 \equiv 5^1 \cdot 5^2 \cdot 5^4 \equiv 5 \cdot (-8) \cdot (-2) \equiv 80 \equiv 14$$

So the encoded message is 14.

d. To decode 14, compute $14^d \equiv 14^3 \equiv 49 \cdot 7 \cdot 8 \equiv 16 \cdot 7 \cdot 8 \equiv -2 \cdot 7 \cdot 4 \equiv (-1) \cdot 7 \cdot 4 \equiv -28 \equiv 5$.

8. (20 points) Consider the following recurrence relation:

$$T(n) = 2T(3n/4) + n$$

$$T(n) = 1 \text{ (for } n \leq 2)$$

Prove that $T(n) = \Omega(n^2)$

We'll show $T(n) \geq cn^2$ for some fixed c by induction.

$$T(n) = 2T(3n/4) + n$$

$$\geq$$

$c(3n/4)^2 + n$ (by induction)

$$= 2 \cdot (9/16)cn^2 + n$$

$$= (9/8)cn^2 + n$$

$$\geq$$

cn^2 (No matter what c is.)

The base case is $n \leq 2$.

9. (20 points) Give an efficient algorithm to find the MAXIMUM weight spanning tree of a graph G(V, E). Determine your algorithm's running time. (I will not specify the running time you should be looking for.)

The simplest thing to do is just create a new weighted graph G' with the same edges as G, but with the edge weights negated. A minimum spanning tree on G' is a maximum spanning tree on G. With Prim's algorithm, this takes time $O(E + V \lg V)$.

If it did not occur to you that negative edge weights were ok, you could have made the edges weights $W - w(u, v)$, where W is the maximum weight edge originally.

10. (40 points) Given a list of n numbers $(a_1, , a_n)$, you want to find any number that list greater than the MEDIAN.

a. Give an upper bound on the exact number of comparisons required (i.e. O(n) is not an appropriate answer) by describing an algorithm and analyzing it.
b. Give a lower bound on the absolute number of necessary comparisons.

a. Scan the first $\lceil n/2 \rceil + 1$ elements of the list and output the maximum. The element you output necessarily has at least $\lceil n/2 \rceil$ elements below it and therefore must be larger than the median. This requires only $\lceil n/2 \rceil + 1$ comparisons.

β⌈

n/2⌉+ 1 is also the lower bound. If you use fewer comparisons, you may have only looked at numbers among the bottom half of the list. In that case, no matter what number you output, it will not be larger than the median.

11. (50 points)

a. (25 points) For a weighted graph *G*, prove that the maximum weight edge in a cycle is not in a minimum spanning tree of *G*. You will receive no credit if your proof is unclear.
b. (25 points) Given a connected graph with exactly V + 10 edges, find its minimum weight edge in the cycle. (You may use the result in part (a), even if you get part (a) wrong.)

a. Suppose *e* were the maximum weight edge and yet was in the Minimum spanning tree *T*. If *e* is removed from *T* it divides the graph into two compents T_1 and T_2. Another edge, *e*', in the cycle with *e* must cross the cut (T_1, T_2), and has smaller weight than *e*. Adding *e*' (instead of *e*) would create a minimum spanning with smaller total weight.
b. Run depth-first search to find any cycle in *G*, and remove the maximum weight edge in the cycle. Repeat the process a total of eleven times until your down to a connected graph with |V| - 1 edges which is the MST.