UCSD CSE 101 MIDTERM 2, Winter 2008

Andrew B. Kahng / Evan Ettinger CSE 101
Feb 29, 2008

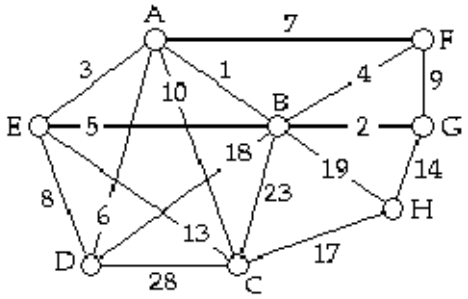
Name: _____

Student ID: _____

Read all of the following information before starting the exam:

- This test has 3 problems and is worth 50 points. It is your responsibility to make sure that you have all of the pages!
- You have 50 minutes to work on the exam. If you finish early, please be quiet when you leave. *Out of courtesy to those still working on the exam, no one will be allowed to turn in their exam during the last 10 minutes.*
- Show all work, clearly and in order, if you want to get full credit. I reserve the right to take off points if I cannot see how you arrived at your answer (even if your final answer is correct).
- Circle or otherwise indicate your final answers.
- Be clear and to the point. You will lose points for rambling and for **incorrect or irrelevant** statements.
- Make sure you have all 6 pages. Use the scratch page and/or the reverse sides of pages as necessary - and ask if you need extra sheets of paper. Good luck!

1. (*10 points*) **MST.** The following problems concern the graph below:



a. (*3 pts*) When executing Kruskal's algorithm, what is the sixth edge that is chosen?

Ans: Edge AC with length 10.

b. (*3 pts*) What is the total weight of the minimum spanning tree?

Ans: 40.

c. (*4 pts*) Does this graph have a unique minimum spanning tree? Explain briefly.

Ans: Yes it does. There will be only one MST for this graph since all the edge weights are distinct.

2. (*20 points*) **Divide-and-Conquer.** You are given a *sorted* (from smallest to largest) array of *distinct* integers $A[1], \ldots, A[n]$, which can be positive, negative, or zero. You want to decide whether or not there is an index $i \in \{1, 2, \ldots, n\}$ such that $A[i] = i$.

For example, $A = [-3, 1, 3, 5, 11]$ has $A[3] = 3$, while $A = [-10, -5, -1, 6, 10, 22]$ does not have any index where $A[i] = i$.

Design the fastest algorithm that you can for solving this problem (10 pts). Prove that your algorithm is correct (5pts), and provide a formal analysis of its running time (5pts).

[Hint: No credit will be given for a linear-time algorithm. Instead, model your algorithm on binary search. You can assume that n is a power of 2.]

Ans: The idea here is to examine the element at $A[\lceil n/2 \rceil]$ recursively.

DQElementEqIndex($A[1 \ldots n]$**,** $offset$**):**

1. if $A[\lceil n/2 \rceil]$ equals $offset + \lceil n/2 \rceil$, then return TRUE
2. if $|A| \leq 1$, return FALSE
3. if $A[\lceil n/2 \rceil] < offset + \lceil n/2 \rceil$, return DQElementEqIndex($A[(\lceil n/2 \rceil + 1) \ldots n]$, $offset + \lceil n/2 \rceil$)
4. else, return DQElementEqIndex($A[1 \ldots (\lceil n/2 \rceil - 1)]$, $offset$)

The initial call to this algorithm will be DQElementEqIndex($A[1 \ldots n]$, 0). The correctness of this algorithm relies on the observation that if $A[\lceil n/2 \rceil] < offset + \lceil n/2 \rceil$, then none of the $A[i]$ can be equal to i for all $i < \lceil n/2 \rceil$. This is because all the items are distinct and sorted. The same argument holds for the case when $A[\lceil n/2 \rceil] > \lceil n/2 \rceil + offset$.

The running time is $T(n) = T(n/2) + O(1)$, which by the Master Theorem is $O(\log n)$.

3. (*20 points*) **Dynamic Programming.** Consider a network of n cities. Between each pair (i,j) of cities, there is a road having a bridge whose clearance height is $h(i,j) \geq 0$ with $h(i,j) = h(j,i)$. A trucking firm wants to determine a route to each city from their distribution center at city S that maximizes the minimum bridge height along the route, subject to the restriction that at most k cities are visited enroute (excluding the distribution center). Given all height values $h(i,j)$ and a number k, your algorithm should return an array that has the largest possible minimum bridge height on routes to each city from S, subject to the restriction that no route can pass through more than k other cities.

a. (*12 pts***)** Write a recurrence to solve this problem (6 pts). Write down what your recurrence relation means in English and argue briefly why it's correct (6 pts). Don't forget the base cases!

Ans: Let D(i,k) = "The maximum minimum bridge height from S to city i along routes of length at most k."

The base case is $D(i,1) = h(S,i)$ for all i.

Consider calculating $D(i,k)$. We want the larger of the two options for it (1) the maximum minimum height for paths of length at most $k-1$ (i.e. D(i,k-1)), or (2) the maximum minimum height for paths that use at most $k-1$ edges to some city j and then road (j,i) as the k^{th} edge. Term (2) can be written $\max_j \min\{D(j,k-1), h(j,i)\}$, where the inner min calculates the minimum bridge height along the path through city j, and we maximize over choices of j.

The recurrence is as follows $D(i,k) = \max\{D(i,k-1), \max_j \min\{D(j,k-1), h(j,i)\}\}$.

b. (*8 pts*) Convert your recurrence into a dynamic programming algorithm (4 pts). What is the space complexity of your algorithm (i.e. how big is the table) (2 pts)? What is the time complexity (in terms of n, k) (2 pts)?

Here is the algorithm:

MinBridge(h, k):

1. Initialize $D(i, 1) = h(S, i)$ for all i.
2. For i = 2 to k, For j = 1 to n
   (a) $D(j, i) = \max\{D(j, i-1), \max_l \min\{D(l, i-1), h(l, j)\}\}$.
3. return $D(j, k)$ for all cities j

The space complexity is dominated by storing the 2 tables $h(i, j)$ and $D(i, k)$ which are of size $O(n^2)$ and $O(nk)$ respectively. Since k can be no larger than n, the space complexity is $O(n^2)$.

The time complexity is dominated by filling in the kn entries of $D(i, k)$. Each entry takes $O(n)$ time to calculate, so the entire running time of the algorithm is $O(n^2k)$.

Scrap Page
(please do not remove this page from the test packet)