

Homework Assignment 5

November 14, 2002

Due Date: December 5

Core Problems

1. (10 pts) Consider the problem of determining whether an undirected graph with n vertices has a path of length at least 2. Assume that the algorithm is restricted to ask questions of the form “Is there an edge between i and j ?”. Give the best adversary argument you can on the number of such questions required to determine if the graph has a path of length 2 or more.
2. (10 pts) Give the best lower bound you can for the problem of finding the median among the $2n + 1$ elements in the two sorted arrays $A[1] < A[2] < \dots < A[n] < A[n + 1]$ and $B[1] < B[2] < \dots < B[n]$ using a comparison-based algorithm that can only learn about the array elements by posing questions of the form is $A[i] < B[j]$ for $1 \leq i \leq n + 1$ and $1 \leq j \leq n$.
3. (10 pts) Consider the problem of merging two sorted arrays a_1, a_2, \dots, a_n and b_1, b_2, \dots, b_n . Prove the best lower bound you can on the number of comparisons required.

4. (15 pts) Consider the problem of determining whether an undirected graph G with 2^k vertices is connected (i.e. whether there is an undirected path between each pair of vertices). Assume that the algorithm is restricted to ask questions of the form “Is there an edge between i and j ?”. Give the best adversary argument you can on the number of such questions required to determine if G is connected. You should go beyond the solution for Practice Problem 2.

Hint: Extend the adversary argument given in the practice problem solutions. Think recursively.

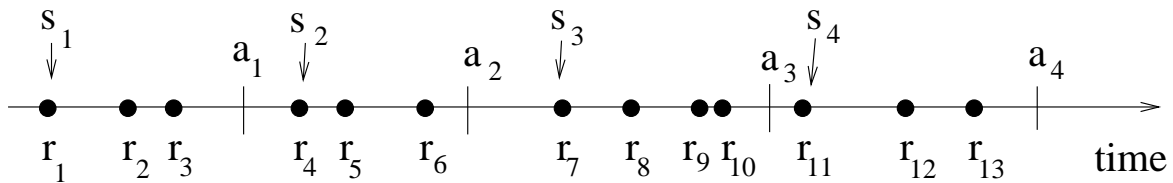
5. (10 pts) Consider the “searching for a hole in the fence” problem that was considered in class. You are initially standing at the origin of the real line. The hole is at some unknown (positive or negative) integer coordinate h . You can move left or right at a cost equal to the distance moved, and the game continues until you reach h . The goal is to minimize the distance traveled. In class we gave a deterministic on-line algorithm with a competitive ratio of 9 (which is optimal for deterministic strategies). In this problem you are to describe and analyze a randomized algorithm whose competitive ration is at most 7.
6. (10 pts) Consider the following on-line problem. You have a video-on-demand service but are limited by your equipment to play only one movie (for one person) at a time. In an on-line fashion you receive requests where each request is for a 2 hour movie and a slack period $s \geq 0$ which indicates how long the customer is willing to wait for the movie to start. Furthermore, once you start a movie for a customer you must finish showing it before servicing another customer. Your goal is to decide which requests to service (and when to service them) as to maximize the number of jobs serviced. Give a 2-competitive on-line algorithm. Be sure to clearly describe the algorithm and prove it has a competitive ratio of 2.

Advanced Problems (Required only for CS 539T students)

7. (15 pts) Using an adversary argument to show that *any* comparison-based algorithm to find the second smallest of n elements must make at least $n + \lceil \lg n \rceil - 2$ comparisons. (Although I'm not asking you to find it, there is a comparison-based algorithm that makes at most $n + \lceil \lg n \rceil - 2$ comparisons.)

Hint: Partition the input elements into groups in which each group has a minimum element that is known to be less than (or equal to) all elements in its group. Then decide how to answer comparisons in the following three cases: comparing two non-minimum elements, comparing a minimum element to a non-minimum element, and comparing two minimum elements.

8. (10 pts) Consider the following on-line problem. There is a sequence of packets received at times r_1, \dots, r_n where $r_i < r_{i+1}$. The goal is to position ACKS (acknowledgments) within the sequence where each ACK will acknowledge all packets that have arrived since the last ACK. We now define the cost of a solution. Let $A = \langle a_1, \dots, a_k \rangle$ be the times of the ACKS where $1 \leq k \leq n$, $a_i < a_{i+1}$ and $a_k \geq r_n$ (i.e. the last ACK follows the last packet received). For $2 \leq i \leq k$ let s_i be the time of the next packet received after time a_{i-1} and for notational convenience let $s_1 = r_1$. Thus s_i is the time when the earliest packet in the sequence of packets acknowledged by the ACK at time a_i is received. Here is a figure illustrating these definitions.



Then the cost of $A = \langle a_1, \dots, a_k \rangle$ is defined as

$$\text{cost}(A) = ck + \sum_{i=1}^k (a_i - s_i)$$

where c is a constant that corresponds to the cost of an ACK. The goal of the on-line algorithm is to minimize cost A . Note that the algorithm selects k , the number of ACKS to use.

Give a 2-competitive algorithm for this problem.