CS 466/666, Fall 2007 Dec. 6–Dec. 17, 4×24 hours
A. Lubiw

TAKE HOME FINAL EXAM

In answering this exam you are encouraged to try the problems yourself first, but you are allowed to use any books/papers, including those available electronically. Acknowledge your sources and write your solutions in your own words. You MAY NOT consult with other people. This includes talking, electronic communication, etc. And in particular, you may not talk with other students in the course about the exam unless both of you have already handed it in. You may ask the TAs or me for clarifications of the problems, but no hints will be given. Please check the course newsgroup for possible corrections.

0. Write out the promise "I promise not to consult with other people about this exam during the 4 days I have it". Sign your promise.

1. [10 marks] Suppose you want to support Search and Insert on a set of elements from a totally ordered universe. A set will be stored as a collection of sorted arrays. Specifically, to store n elements, let $b_k, \ldots, b_0$ be the binary representation of n, and use a collection of arrays, $A_k, \ldots, A_0$ where A_i is empty if b_i is 0, and A_i has 2^i elements otherwise. It doesn't matter which elements go in which A_i's.

   (a) [3 marks] Give a Search algorithm and analyze its worst case run time.

   (b) [7 marks] Give an Insert algorithm and analyze its worst case and amortized run times.

2. [18 marks] The following two problems are about unit squares in the plane. The squares have horizontal and vertical sides.

   A. [6 marks] Packing. The problem here is to choose a disjoint subset of squares of maximum cardinality. In class we gave a simple greedy algorithm that gave at least $\frac{1}{4}$ times the optimum. Consider the following algorithm: choose a square of minimum x-coordinate; discard this square and any it intersects; repeat. Prove that this algorithm chooses a set of size at least $\frac{1}{2}$ times the optimum.

   B. [12 marks] Colouring. The problem here is to colour the squares with a minimum number of colours, such that two squares that intersect have different colours. The goal is to give a polynomial time 4-approximation algorithm.

      i. [1 mark] Formulate the problem as a graph colouring problem.

      ii. [2 marks] Show that greedy colouring of any graph produces a colouring of at most $\delta + 1$ colours, where δ is the maximum degree. Greedy colouring means: for $i = 1, \ldots, n$ give vertex i the minimum colour number that you can.

      iii. [1 mark] If a graph has a clique of size k, what can you say about the size of a minimum colouring?

      iv. [5 marks] For the particular graphs arising from squares, show that a vertex of degree d in the graph has a fairly large clique in its neighbourhood.

      v. [3 marks] Put the above together to obtain a polynomial time 4-approximation algorithm.

3. [10 marks] Consider the problem of packing n line segments of lengths $l_1, \ldots, l_n$ into m tracks to minimize the maximum length of a track. For example, segments of lengths 2, 4, 2, 1, 3 can be packed into 3 tracks as follows: track 1 has 2, 1; track 2 has 4; track 3 has 2, 3. This solution has maximum track length 5, but there is a better solution.

   A simple on-line algorithm places interval i in the track that has minimum length so far. Note that the example above followed this strategy. Prove that this algorithm has approximation ratio $2 - \frac{1}{m}$. Hint: Focus on the last segment in the maximum length track.

4. [18 marks] Consider the k-server problem with points $p_1, \ldots, p_n \in R$. The k servers are points that can move along the real line. You must deal on-line with a sequence of requests, each request asking for a server to be at some point p_i. The cost is the total distance travelled by the servers. In class we looked at the local greedy algorithm which always moves the closest server to the request point. (In case multiple servers are equally close, only move one of them.)

   Consider the following algorithm: if a server is already at p_i do nothing; otherwise if all servers are to the left [or all to the right] of p_i then move the closest server to p_i; otherwise, take the closest server to the right and the closest server to the left and move them at uniform speed towards p_i, stopping the motion when one server reaches p_i. Observe that this algorithm may leave a server at a point that is not one of the p_j's.

   The goal of this problem is to show that this algorithm is k-competitive.

   (a) [3 marks] For points $p_1 = 1$, $p_2 = 3$, $p_3 = 4$, and $k = 2$ with servers initially at p_1 and p_3, give a request sequence to show that the local greedy algoirthm has unbounded competitive ratio, and then show and analyze what the above algorithm does on that request sequence.

   (b) [3 marks] Show that an optimum off-line algorithm can be modified without changing its cost so that for each request, no server moves except the one that moves to meet that request.

   (c) [4 marks] This is about a general amortized analysis technique for competitive ratios. Suppose that ALG is a deterministic algorithm for an on-line problem. Suppose that for each request, OPT moves and then ALG moves. Let Φ be a non-negative potential function that depends on the configuration of ALG and OPT, such that $\Phi = 0$ in the initial configuration. Suppose also that: (1) if OPT incurs cost s_i for the ith request, Φ increases by at most ks_i; (2) if ALG incurs cost t_i for the ith request, Φ decreases by at least t_i. Prove that ALG is k-competitive. Hint: the amortized cost to ALG for the ith request is $t_i + \Delta\Phi$.

   (d) [8 marks] Returning to our specific algorithm ALG, define a potention function as follows. For any bijection between the k servers of ALG and the k servers of OPT, let the cost be the sum of the distances between corresponding servers. Define Φ as: k times the minimum cost of a bijection between the servers of ALG and the servers of OPT plus the sum of the distance between all $\binom{k}{2}$ pairs of ALG's servers. Using this potential function, prove that ALG is k-competitive.