CSE 208A Final Exam
Randomized Algorithms, Approximation, and Relaxation
Due June 14, 11:59 PM

Answer ALL problems below. For each of the algorithm problems, design as asymptotically efficient an algorithm as possible. Give a correctness argument (explanation, if it is relatively simple, or proof if not) and time analysis. You may use any well-known algorithm or data structure, or algorithm from the text or from class, as a sub-routine without needing to provide details.

Randomized approximation algorithms: NOTE CHANGE FROM TUESDAY. (Thanks, Christian) In relational databases, a report is a set S of vectors $\vec{x}$ indexed by some fields I, $\vec{x} = (x_i | i \in I)$. Let S (indexed by I) and T (indexed by J) be two reports of size s and t, and let $i \in I$, $j \in J$. The *join* of S and T on fields i, j is (equivalent to) the set $Join(S,T,i,j) = \{(\vec{x}, \vec{y}) | \vec{x} \in S, \vec{y} \in T, x_i = y_j\}$. In query optimization, you frequently want to estimate the size of a join without explicitly computing it. Assume you can perform arithmetic and linear operations on x_i in unit time given $\vec{x}$. Give a probabilistic algorithm that with probability at least .9 that given S, T, i, j and $1/2 > \epsilon > 0$ returns a value J with the following property: $(1-\epsilon)|Join(S,T,i,j)| - \epsilon(s+t) \le J \le (1+\epsilon)|Join(S,T,i,j)| + \epsilon(s+t)$. For $\epsilon \ge (s+t)^{-1/4}$ your algorithm should run in time $O(s+t)$ and use $o(s+t)$ external memory. (10 pts. Hint: this is similar to the homework problem on estimating the union, but be careful about independence of random variables).

Convex programming: Consider the problem of, given n points $p_1, ...p_n$ in R^d, finding the smallest sphere that contains all the points, i.e., a vector $c \in R^d$ and a radius R so that the Euclidean distance between c and p_i is at most R for each i. You want to minimize R

a. (2 points) . Give an efficient algorithm to give an approximation within a factor 2.

b. (8 points). Give a polytime approximation scheme for approximating the minimum within a factor of $1+\epsilon$. Your algorithm needn't be especially efficient.

On-line algorithms: Consider the bin packing problem when restricted as follows: Bins are all size 3, and items are either size exactly 1 or 2. Call this problem $1, 2$ bin-packing.

(2 points) Give an explicit formula for the size of the mimimum packing of t items of size 2 and u items of size 1.

(4 points) An on-line algorithm for bin-packing must process the packages in (unsorted) order, and assign each to a bin as it is processed. Prove that

no deterministic on-line algorithm for $1, 2$ bin packing has approximation ratio less than $3/2$.

(4 points) Give and analyze a deterministic on-line $1, 2$ bin-packing algorithm with approximation ratio $3/2$.

(3 points) Remember that an algorithm has *competitiveratio* R if there is a constant B with $Cost(Algorithm(x)) \leq R * Cost(Opt(x)) + B$ for every x. Give a deterministic on-line $1, 2$ bin packing algorithm with competitive ratio less than $3/2$.

(3 points) A probabilistic algorithm has approximation ratio R if $E(Cost(Algorithm(x))) \leq RCost(Opt(x))$ for every instance x. Prove that no probabilistic on-line 1-2 bin-packing algorithm has approximation ratio less than $4/3$.

(4 points) Give and analyze a probabilistic on-line 1-2 bin-packing algorithm with approximation ratio less than 1.5, i.e., better than any deterministic algorithm.