Algorithms and Theoretical Computer Science
Ph.D. Qualifying Examination
Fall 2003

Name:	
Net ID:	Alias:

- Please print your name, NetID, and an alias of your choice in the boxes above. Write your alias, but *not* your name or netID, on each page of your answers. Using an alias will allow us to grade your exam anonymously.
- The exam consists of eight written questions, four in the morning and four in the afternoon. You will have three hours for each group of four questions. Please start your answers to each numbered question on a new sheet of paper.
- All else being equal, it is better to solve some of the problems completely than to get partial credit on every problem.

#	Score	Grader
1		
2		
3		
4		
5		
6		
7		
8		
$\sum$		

1. (a) Describe and analyze an algorithm to sort an array $A[1..n]$ by calling a subroutine SqrtSort(k), which sorts the subarray $A[k+1..k+\lceil\sqrt{n}\rceil]$ in place, given an arbitrary integer k between 0 and $n-\lceil\sqrt{n}\rceil$ as input. Your algorithm is not allowed to inspect or modify the array except by calling SqrtSort. How many times does your algorithm call SqrtSort in the worst case?

   (b) Prove that your algorithm from part (a) is asymptotically optimal; that is, prove that no algorithm can sort using asymptotically fewer calls to SqrtSort than your algorithm.

   (c) Now suppose SqrtSort is implemented recursively, by calling your sorting algorithm from part (a). For example, at the second level of recursion, the algorithm is sorting arrays roughly of size $n^{1/4}$. What is the worst-case running time of the resulting sorting algorithm? For full credit, do *not* assume that $\sqrt{n}$ is always an integer.

2. (a) Describe and analyze an algorithm to compute the union of n circular disks in the plane, where the boundary of every disk passes through a common point.

   (b) Describe and analyze an algorithm to compute the union of n circular disks in the plane, where the center of every disk lies on a common line.

   For full credit, your algorithms must be optimal, and their descriptions must be complete and self-contained (although one can refer to the other).

3. (a) Give a definition (clean, clear, complete) of a *2-head finite-state automaton (2-head FSA)*, its operation, and the language it accepts.

   (b) Prove or disprove: The language accepted by each 2-head FSA is *regular.*

   (c) Prove or disprove: The language accepted by each 2-head FSA is *context-free.*

   (d) Prove or disprove: The language accepted by each 2-head FSA is *context-sensitive.*

   (e) Prove or disprove: It is decidable whether a 2-head FSA accepts an input string (both the FSA and the string are input parameters).

   (f) Prove or disprove: It is decidable whether a 2-head FSA accepts an empty language.

4. Given a set of points P in the plane, the convex hull of P is the smallest convex polygon that contains P. A point of P is called a corner of $\mathcal{CH}(P)$ if it does not lie in the interior of $\mathcal{CH}(P)$, or in the interior of the edges of $\mathcal{CH}(P)$.

   (a) Let V be a subset of the integer grid

   $$U = \{(i,j) : 1 \le i \le n, 1 \le j \le n\}.$$

   Prove that $\mathcal{CH}(V)$ has at most $O(n^{2/3})$ corners.

   (b) A point $(a,b) \in U$ is primitive if $\gcd(a,b) = 1$. Prove that the number of primitive points in U is $\Omega(n^2)$. *Hint:* consider the Euler number $\phi(n)$, which is the cardinality of the set of numbers smaller than n that has gcd 1 with n. Namely,

   $$\phi(n) = |\{a : 0 \le a < n, \gcd(a,n) = 1\}|.$$

   Prove that $\sum_{i=1}^{n} \phi(i) n/i = \Omega(n^2)$.

   (c) Show a subset $V \subseteq U$, such that $\mathcal{CH}(V)$ has $\Omega(n^{2/3})$ corners.

5. For a graph G with positive edge weight function $w : E(G) \to \Re^+$, the *shortest path metric* d_G is defined as follows: for vertices u, v in G, $d_G(u, v)$ is the weigthed length of the shortest weighted path between u and v in G. In the *k-median problem,* you are asked to find a set C of k vertices, such that $\mu_G(C) = \sum_{v \in V(G)} d_G(v, C)$ is minimized, where $d_G(v, C) = \min_{c \in C} d_G(v, c)$.

   (a) Let T be a tree with n vertices and with a positive edge weight function w. Assume the maximum degree of T is bounded by Δ. Describe a polynomial time algorithm for this problem (the exponent may depend on k and Δ). How fast is your algorithm?

   (b) Assume, that you are given a randomized polynomial time algorithm such that for some constant $M > 1$, when given a weighted graph G, it outputs a weigthed tree T on the vertices of G, of degree bounded by some Δ and such that for any $u, v \in V(G)$, we have $d_G(u, v) \le d_T(u, v)$ and $E[d_T(u, v)] \le M d_G(u, v)$ (the expectation here is in fact important, as one can show that there is no such tree without it).

   Describe a polynomial time algorithm that computes an approximation to the optimal k-median of G. Namely, it computes a set X of k vertices in G, such that $E[\mu_G(\tilde{C})] \le M \mu_G(C^*)$, where C^* is an optimal solution to the k-median problem in G. $\mu_G(C)$.

   (c) Describe a polynomial time algorithm, such that given parameters ε, δ, a graph G and k, it outputs a solution to the k-median problem of cost $(1 + \varepsilon) M \mu_G(C^*)$ with probability larger then $1 - \delta$. What is the running time of your algorithm ?

6. A Turing machine M^A with oracle A is a multi-tape (deterministic or nondeterministic) machine that has a special tape called the *query tape*, and three special states query state $q_?$, and answer states q_{yes} and q_{no}. The computation of such a machine proceeds like an ordinary Turing machine except for transitions for the query state; from the query state the machine moves to either q_{yes} or q_{no} depending on whether the string on the query tape belongs to the language A or not. The answer states allow the machine to use this answer in its further computation. The time complexity of such a machine is defined like for ordinary machines (each query step counts as one step only). The complexity class P^A is the class of languages that can be recognized by a deterministic machine with oracle access to A in polynomial time. NP^A is defined analogously.

   (a) Show that there is a language A such that $\mathrm{P}^A = \mathrm{NP}^A$.

   (b) Show that there is a language A such that $\mathrm{P}^A \ne \mathrm{NP}^A$. *Hint:* Consider $L_A = \{0^n \mid \exists x \in A,\ |x| = n\}$. L_A is clearly in NP^A. Construct by diagonalization a language A such that $L_A \notin \mathrm{P}^A$.

7. We consider keys taken from a universe set U of size u and assume that each key is represented by $\lceil \log u \rceil$ bits, and can be stored in a single memory cell. Furthermore, all sort of "reasonable" operations can be performed on these keys in time $O(1)$: addition, substraction, shift, mask, comparison, etc., even multiplication. You have at your disposal a hash-table data structure that supports update operations (insert and delete) in amortized time $O(1)$ and search operations in worst-case time $O(1)$ for keys taken from U.

   The goal of this problem is to design a data structure that uses linear storage (in the number of keys currently stored) and supports the following operations: (i) update, (ii) search, (iii) successor and predecessor, in amortized time $O(\log \log u)$ for keys taken from U. Let n be the number of keys stored in the data structure. Proceed in two steps as follows:

(a) Design a data structure of size $O(n \log u)$ using a single hash table (the operation bounds may be larger than required at this point). *Hint:* Recall that each key is a string of $\lceil \log u \rceil$ bits.

(b) Then modify (augment?) it to save the $\log u$ factor (and achieve the required query times). *Hint:* Don't store all the keys directly in the hash table.

Your description and analysis should be as clear, complete and correct as possible. You may simply describe the complete data structure, the (a), (b) break down is provided as a help.

8. (a) Prove that for every language $L \subseteq 0^*$, the language L^* is regular.

   (b) Prove that there is a language $L \subseteq \{0,1\}^*$ such that the language L^* is not regular.