SPRING 1993
COMPUTER SCIENCES DEPARTMENT
UNIVERSITY OF WISCONSIN—MADISON
PH. D. QUALIFYING EXAMINATION

Foundations of Computing
Depth Examination

Monday, February 8, 1993
3:00 – 7:00 PM
113 Psychology

GENERAL INSTRUCTIONS:

1. Answer each question in a separate book.
2. Indicate on the cover of *each* book the area of the exam, your code number, and the question answered in that book. On *one* of your books list the numbers of *all* the questions answered. *Do not write your name on any answer book.*
3. Return all answer books in the folder provided. Additional answer books are available if needed.

SPECIFIC INSTRUCTIONS:

Answer any *four* of the following *six* questions.

POLICY ON MISPRINTS AND AMBIGUITIES:

The Exam Committee tires to proofread the exam as carefully as possible. Nevertheless, the exam sometimes contains misprints and ambiguities. If you are convinced a problem has been stated incorrectly, mention this to the proctor. If necessary, the proctor can contact a representative of the area to resolve problems during the *first hour* of the exam. In any case, you should indicate your interpretation of the problem in your written answer. Your interpretation should be such that the problem is nontrivial.

Theory of Computing
Depth Exam
Spring 1993

Directions: There are six questions, of which you must do FOUR.

1. The *longest common subsequence* problem is the following. Given two strings $A = a_1 a_2 \ldots a_n$ and $B = b_1 b_2 \ldots b_m$, find the longest string $x_1 x_2 \ldots x_k$ which is a subsequence of both A and B. (If S is a subsequence of T, then the letters of S need not appear consecutively in T; for example, the word THEORY has HER as a subsequence.) Give an efficient algorithm to solve the longest common subsequence problem, analyzing its complexity as a function of n and m.

2. Consider the following problem. Given a set of deterministic finite state automata, $A_1, \ldots, A_m$, is there a string that is accepted by all of them? Prove that this problem is in P when m is a constant. Prove that it is PSPACE-complete in the general case, where m is not a constant.

   [Hint: to prove the hardness part, construct the automata so that, given a computation (sequence of configurations) of a space bounded machine, each automaton checks one position in each configuration.]

3. Define a relation $<$ on $\{0,1\}^n$ as follows: $x < y$ iff y results from x by changing exactly one 0 to a 1. If f is a Boolean function, we say that f has a *mind change* from x to y if $x < y$ and $f(x) \neq f(y)$. The *mind change complexity* of a Boolean function is the minimum number of mind changes over all sequences

$$0^n = x_0 < x_1 < x_2 < \ldots < x_n = 1^n.$$

   a) Show that there is no polynomial-time algorithm to compute the mind change complexity of a Boolean function (presented using, say, a Boolean circuit) unless P=NP.

   b) Note that the "obvious" method to find the mind change complexity of f evaluates the function $(n-2)!$ times. Find a better algorithm, which uses $O(n2^n)$ evaluations.

4. A polygon (given as a sequence of vertices in the plane) is said to be *star-shaped* if there is a point P in the interior of the polygon that can "see" all its vertices. (More formally, the line PV must lie inside the polygon for every vertex V.) The set of all such points is called the *kernel* of the polygon. Give a linear-time algorithm to compute the kernel of a star-shaped polygon (which you can assume is presented as a list of vertices in "clockwise" order).

5. The MAX 3SAT problem is: given a Boolean formula in conjunctive normal form, where each clause has at most 3 literals, what is the maximum number of simultaneously satisfiable clauses?
   a) Prove on a random assignment to the variables, one can expect that at least 1/2 of the clauses are satisfied.
   b) Find an approximation algorithm for the MAX 3SAT problem, whose output is within a factor of 2 of the true solution.

6. Let S be a set of n points in the plane. The *diameter* of S is the diameter of the smallest circle that contains all points of S. Give an algorithm for determining the diameter of S with as small a running time as possible.