

# Ph.D. Comprehensive Exam: Math and Algorithms

Fall 1999

The exam includes eight problems;  
the length of the exam is three hours.

### Problem 1

Give the best-case complexity and worst-case complexity ( $\Theta$ -notation) of the following recursive algorithms.

DIVIDER( $n$ )

**if**  $n$  is even

**then return** DIVIDER( $n/2$ )

**else return**  $n$

MULTIPLIER( $n$ )

**if**  $n$  is divisible by 1999

**then return**  $n$

**else return**  $n \cdot \text{MULTIPLIER}(n - 1)$

ADDER( $n$ )

**if**  $n > 0$

**then return** ADDER( $n - 1$ ) + ADDER( $n - 1$ )

**else return** 0

**Problem 2**

Suppose that we have four algorithms, called  $A_0$ ,  $A_1$ ,  $A_2$ , and  $A_3$ , whose respective running times are  $n$ ,  $n^2$ ,  $\lg n$ , and  $2^n$ . If we use a certain old computer, then the maximal sizes of problems solvable in an hour by these algorithms are  $s_0$ ,  $s_1$ ,  $s_2$ , and  $s_3$ .

Suppose that we have replaced the old computer with a new one, which is  $k$  times faster. Now the maximal size of problems solvable in an hour by  $A_0$  is  $k \cdot s_0$ . What are the maximal problem sizes for the other three algorithms, if we run them on the new computer?

**Problem 3**

Consider the following recurrence:

$$\begin{aligned} C_0 &= 0 \\ C_n &= 3 \cdot C_{n-1} + \frac{3^n}{n \cdot (n+1)} \quad (\text{where } n \geq 1) \end{aligned}$$

Give a formula for  $C_n$  in terms of  $n$ , without the use of recurrence. For the *full* credit, you need to provide an exact formula, rather than a  $\Theta$ -notation; however, you can get a *partial* credit for determining the order of growth ( $\Theta$ -notation) of  $C_n$ .

**Problem 4**

King Arthur once invited a number of knights to his castle, where they stayed for several days. Each evening, the king and his guests dined at the Round Table; in total, there were *twenty-three* people at the table. According to the king's decree, they took different seat on different evenings, and no two people sat next to each other more than once. What is the maximal number of seating arrangements that satisfy the king's condition?

### Problem 5

Suppose that you are using a programming language that allows only integer numbers and supports three operations on them: addition, subtraction, and multiplication; the running time of each operation is constant, that is,  $\Theta(1)$ . The language does *not* have operations for division and exponentiation.

Write an efficient algorithm `DIVIDE`( $n, m$ ) that computes  $\lfloor n/m \rfloor$ , where  $n$  and  $m$  are positive integers, and give the time complexity of your algorithm. Note that computing  $\lfloor n/m \rfloor$  in  $\Theta(\lfloor n/m \rfloor)$  time is *too slow*, and you need to design a faster algorithm.

**Problem 6**

Consider the *Two-Coloring Problem*, which requires to assign colors to the vertices of a given undirected graph. Each vertex must be either white or black, and adjacent vertices must have different colors. Determine whether this problem is NP-hard and justify your answer.

**Problem 7**

Design two different approximation algorithms for the *Traveling Salesperson Problem*, and determine the time complexity of your algorithms.

Recall that the Traveling Salesperson Problem requires to input a weighted graph, and find a minimum-weight cycle that includes all vertices of the graph. The purpose of approximation algorithms is to find a cycle whose weight is “almost” minimal. You may assume that the graph is undirected, every two vertices are connected by a finite-weight edge, and all weights are positive.

### Problem 8

We consider a modified version of the *Satisfiability Problem*. The input in this problem consists of two parts: (1) a set of boolean variables and (2) a collection of clauses constructed from these variables. Each clause is a *disjunction* of one or more literals, and each literal is either a variable or a negated variable. Note that we do *not* allow empty clauses, that is, each clause must include at least one literal. Let  $n$  be the total number of the input clauses. The task is to determine whether there is a truth assignment that satisfies at least  $\lceil \frac{2}{3}n \rceil$  clauses. Prove that this problem is NP-complete.