Algorithms CMSC-27000/37000 Final Exam. March 19, 2003

Instructor: László Babai

Show all your work. **Do not use book, notes, or scrap paper.** When describing an algorithm in pseudocode, **explain the meaning of your variables** (in English). This final exam contributes 25% to your course grade.

1. (3 points) Sort the following four functions according to their asymptotic rate of growth: $\binom{n}{5}$, n^5, $n!$, $\pi(n^6)$. ($\pi(x)$ is the number of primes $\le x$.) State the strongest asymptotic comparisons between consecutive members of your sorted list. Prove the asymptotic relations claimed.

2. (6 points) Describe the (sequential) MERGE-SORT algorithm. State the recurrent inequality satisfied by the cost functions (number of comparisons, bookkeeping steps). Asymptotically evaluate the cost function. Prove your answers.

3. (4 points) Describe a linear-time algorithm to determine whether or not a given digraph $G = (V, E)$ is strongly connected. G is given by an array of adjacency lists. Describe all algorithms you use as "subroutines."

4. This question is about the RSA crypotsystem.

   (a) (4 points) Describe the numbers you need to publish and those you need to keep private in order to receive RSA-encrypted messages.

   (b) (2 points) The RSA plaintext is an integer in the range $\{0, 1, \ldots, ?\}$. Fill in for the question mark.

   (c) (2 points) Describe the RSA encryption and decryption functions.

   (d) (4 points) Prove that RSA encryption can be computed in polynomial time (given the appropriate keys). Give a clear statement of the computational problem that needs to be solved. Describe the required algorithm in pseudocode. Name the method.

   (e) (G only, 4 points) Prove the correctness of RSA decryption (that the original plaintext is recovered). *Hint.* Fermat's little Theorem.

5. (a) (6 points) Out of n items, an algorithm finds the median using $T(n)$ comparisons, where

$$T(n) \leq T(n/5) + T(7n/10) + O(n).$$

Prove: $T(n) = O(n)$.

   (b) (G only, 6 points) Describe the algorithm referred to in the preceding question. State the more general problem it solves recursively. State the algorithm in English. Clarity counts. Prove the recurrence given.

6. (a) (3 points, G only) Give a formal definition of NP.

   (b) (3 points) Define what it means that a language $L_1 \subseteq \Sigma_1^*$ is Karp-reducible to a language $L_2 \subseteq \Sigma_2^*$. Clearly state the domain of your reduction function f.

   (c) (2 point) Let f be a Karp-reduction function. True or false: $f \in$ P. Reason your answer.

   (d) (3 points) What does it mean that the language L is NP-complete?

   (e) (6 points) Give a precise definition of three NP-complete problems (input, question). The problems should not be close relatives. (Ask the instructor if you are not sure your candidate problems are distant enough.)

7. (a) (2 point) Describe the rule satisfied by the keys in a binary search tree.

   (b) (3 points) Write a pseudocode for the in-order traversal of a binary search tree.

   (c) (3 points) Prove that it takes $\gtrsim n \log n$ comparisons to arrange n keys (real numbers) in a binary search tree.

   (d) (3 point) Describe in pseudocode the FIND operation in a binary search tree. What is the cost of this operation?

   (e) (4 points) What is the balancing condition in AVL trees?

   (f) (G only, 4 points) Prove, based on your answer to the preceding question, that the depth of an AVL tree is $O(\log n)$. State the exact constant hidden in the big-oh notation.

8. (8 points) Describe Batcher's Odd-Even Merge network as a parallel algorithm in pseudocode. (MERGE, not SORT!) Do not forget to indicate parallelism. Evaluate the parallel time (depth of the network).

9. (a) (3 points) Define the concept of a loop-invariant. Be as formal as reasonable. Make sure you give a clear definition of what kind of statement can be a candidate loop invariant.

   (b) (2+2+2 points) Decide which of the following statements are loop-invariants for Dijkstra's algorithm. Reason your answers. (b1) All black vertices are accessible. (b2) All accessible vertices are black. (b3) All accessible vertices will eventually become black.

10. (2+2+2 points) State the input and the output to each of the following three algorithms:

    (a) Dijkstra's; (b) Prim's; (c) Floyd's.

    Make sure you state the conditions the input must satisfy. Organize your answer in three columns to permit direct (line-by-line) comparison.

11. (a) (4 points) What is the meaning of the cost function after the i-th round of Dijkstra's algorithm? (This is the "brain" of the dynamic programming aspect of Dijkstra's algorithm.)

    (b) (G only, 4 points) Let Q_2 be the answer to the preceding question. Q_2 is not a loop invariant but there is a very simple loop invariant Q_1 such that $Q_1 \& Q_2$ is a loop invariant; this fact is the key to the proof of correctness of Dijkstra's algorithm. State Q_1.

12. (4 points) Give an exact definition of the type of data maintained and the queries served by a UNION-FIND datastructure. Warning: you lose points if you include comments not relevant to the concept of this data structure, such as specifics about an application or implementation details.

13. (G only, 8 points) *Matrix chain multiplication.* We define the cost of multiplying an $a \times b$ matrix and a $b \times c$ matrix as abc (which is the actual number of multiplications required). Suppose we are given a sequence of $n+1$ positive integers, $k_0, k_1, \ldots, k_n$, representing the dimensions of a sequence of n matrices, $A_1, \ldots, A_n$, where the dimensions of A_i are

$k_{i-1} \times k_i$. (The matrices are not given.) We wish to organize the computation of the product matrix $A_1 \dots A_n$ by multiplying two matrices at a time. This requires fully parenthesising the long product. (E. g., we get a different cost for $((A_1 A_2)(A_3 A_4))$ than for $(((A_1 A_2) A_3) A_4)$.) Find the optimal (min-cost) arrangement of parentheses using $O(n^3)$ operations (arithmetic and comparisons). Describe your algorithm in **pseudocode.** *Hint:* dynamic programming. Make an $n \times n$ array of problems $P_{i,j}$. Half the credit goes for the clear definition of $P_{i,j}$ (the "brain" of the dynamic programming algorithm).

14. (G only)

    (a) (3 points) Describe DFS by pseudocode. Your input is a digraph. No source vertex is specificed; the algorithm must reach every vertex.

    (b) (2 point) Define how DFS classifies the edges of a digraph into 4 categories.

15. (G only, 6 points) Assuming SAT is NP-complete, prove that 3-SAT is NP-complete.

16. (G only, 4 points) State the decision problem FACTOR (instance, question). Prove: FACTOR $\in$ NP $\cap$ coNP. You may use a major result published in 2002 regarding primality testing. State the result and indicate where you use it.