Mid-Semester Test

Maximum marks: 50 | September 26, 2008 | Total time: 2 hours

Roll no: ____________________ **Name:** ____

- *Write your answers in the question paper itself. Be brief and precise. Answer all questions.*
- *For any algorithm you design in this test, you should supply an argument about its correctness and also about its running time. Writing only a pseudocode does not suffice.*

1. Suppose that the running time $T(n)$ of an algorithm on an input of size n satisfies

$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + cn \log n$$

for all $n \geqslant 2$, where c is a positive constant. Deduce that $T(n) = \Theta(n \log^2 n)$. **(10)**

Let M denote the maximum of these absolute differences, and m the minimum of them. The problem of determining M (resp. m) is called the maximum-difference (resp. minimum-difference) problem.

(a) Design an $O(n)$-time algorithm to compute M. **(5)**

(b) Design an $O(n \log n)$-time algorithm to compute m. **(5)**

Element uniqueness: Determine whether an array of n integers contains duplicates.

It can be proved (using techniques other than reduction) that *element uniqueness* has a lower bound of $\Omega(n \log n)$ (under reasonable models of computation). Using this result, prove that the algorithm of Part (b) is optimal. **(5)**

3. We often need to compute the convex hull (smallest enclosing convex polygon) of general geometric objects.

(a) Design an $O(n \log n)$-time algorithm to compute the convex hull of n triangles in the plane. **(5)**

non-convex) in the plane. **(5)**

(c) What is the smallest convex polygon enclosing a circle? **(5)**

substring of S and T. Design an $O(mn)$-time dynamic programming algorithm for solving this problem. **(15)**
(**Hint:** Consider the longest common suffix (or its length) $E_{i,j}$ of $S[0 \ldots i]$ and $T[0 \ldots j]$.)
(**Remark:** This problem can be solved in $O(m+n)$ time by using sophisticated data structures like *generalized suffix trees*.)