Algorithms – CS-37000 Homework 6 – January 26, 2006

Instructor: László Babai Ry-164 e-mail: laci@cs.uchicago.edu

ADVICE. Take advantage of the TA's problem sessions.

READING KT, Chapter 5 (Divide and Conquer), sections 5.1 (merge-sort), 5.2 (recurrences), 5.4 (closest pair), 5.5 (convolution and Fast Fourier Transform). "Loop invariants" handout.

HOMEWORK. Please **print your name on each sheet.** Put each solution on a separate sheet. Please try to make your solutions easily readable.

This homework is due on **Tuesday, January 31**. at the **beginning of the class**.

The first four questions ask you to solve the "Practice problems" from the "Loop invariants" handout.

6.1 (2+3+6 points) Loop-invariants, 1.

6.2 (5 points) Loop-invariants, 2.

6.3 (3+5 points) Loop-invariants, 3.

6.4 (4+4+4 points) Loop-invariants, 4.

In the next problem, we revisit the second part of the "Car race" problem (Problem 4.6(b)). The BFS-based algorithm discussed in class requires making decisions of the form "does the point (x, y) belong to R?" ("membership query"). We want to answer these queries in $O(1)$ time. This is easy if we are permitted to set up an $n \times n$ array representing membership in R. We need to be more clever, however, when $|R| \le n$.

Here is a self-contained statement of the problem (you may forget about car race, velocities, etc.).

6.5 (10 points) Let R be a list of m points (x_i, y_i) in the $(n+1)\times(n+1)$ grid ($i = 1, \ldots, m$; $0 \le x_i, y_i \le n$; x_i and y_i are integers.) Assume $m \le n$. Using $O(mn)$ time and space, construct a data structure which will permit membership queries (questions of the form "$(x, y) \in R$?") to be answered in $O(1)$ time.

Finally, a problem on the creative use of the information theory lower bound.

6.6 (a) (10 points) Lucy Ferro designed a comparison-based algorithm that takes an array of n real numbers already arranged in a heap (the heap is implemented as an array) and sorts them. Lucy also brings a performance guarantee: the algorithm will never make more than $f(n)$ comparisons. Lucy is always right (and devilishly clever). Prove: $f(n) \gtrsim n \log_2 n$. You will receive partial credit (5 points) if you prove the weaker relation $f(n) = \Omega(n \log n)$. – WARNING: this question is NOT about the Heapsort algorithm.

It is about Lucy's algorithm which may well be far more intricate and makes use of the full access to the array of data. Your job is to prove that no matter how clever Lucy's algorithm may be, it nevertheless requires $\gtrsim n \log_2 n$ comparisons in the worst case.

(b) (3 points) The "Warning" refers to "worst case." Which word in the problem statement implies that we are talking about the complexity of the *worst case*? (The answer should be just one word.)