Please print your name, major/field, year, Undergrad/Graduate status on your homework. The problems marked “HW” are homework problems due Thursday, January 16, before class. (Note: there are two of these; only one of them was assigned in class.)


2.2 (DO) (a) Out of 12 given coins we know that exactly one is fake but we don’t know whether it is heavier or lighter. Given a balance, use three measurements to find the fake coin and decide whether it is heavier or lighter than the valid coins. (All the valid coins have the same weight. You can put any number of coins in each tray of the balance. Each measurement on the balance results in one of three possible outcomes: left-heavy, right-heavy, or equal.) Indicate why your procedure is correct. (b) Make your procedure non-adaptive: you need to tell in advance which coins to put in each tray in each of the measurements.

2.3 (HW, due January 16)

(a) Prove: if we have 14 coins in Problem 2.2(a) then three measurements don’t suffice. Make your proof very short and elegant (just a couple of lines).

(b) Prove: if we have 13 coins in Problem 2.2(a) then three measurements don’t suffice. Make your proof short and elegant (couple of short paragraphs). (3+6 points) (An elegant solution to (b) will earn you all the 9 points.)

2.4 (DO) Prove: \(\log(n!) \sim n \log n\). Do NOT use Stirling’s formula. Make your solution simple.

2.5 (HW, due Jan 16) The input to a sorting algorithm is a list \(L[1, \ldots, n]\) of \(n\) real numbers. The only way a “comparison-based algorithm” can obtain information about its input is by queries of the form “\(L[i] \leq L[j]\) ?”
Let \( A \) be a comparison-based deterministic algorithm that purports to sort. We turn \( A \) into a randomized algorithm \( A' \) by first randomly permuting the input items and then applying \( A \).

Suppose \( A \) never makes more than \( k = \lfloor 0.98n \log_2 n \rfloor \) comparisons on input lists of length \( n \). Prove that \( A' \) almost never gets it right. More precisely, let \( L \) be a list of \( n \) distinct reals. Let us apply \( A' \) to \( L \). Prove that the probability that \( A' \) returns the correct sorting of \( L \) approaches zero as \( n \to \infty \). \( (8 \text{ points}) \)

2.6 (CHALLENGE) We have \( n \) coins, at least one of which is fake. All the valid coins have the same weight; all the fake coins have the same weight; and the fake coins are lighter than the valid ones. Count the fake coins using \( O(\log^2 n) \) measurements on a balance.

2.7 (DO)

(a) Given a list of \( n \) numbers, we wish to find their Max in the comparison model. Prove: (a1) \( n - 1 \) comparisons suffice and (a2) this is best possible.

(b1) Find the Max and the second-maximal element using \( n + \log_2 n + O(1) \) comparisons.

(b2) (CHALLENGE) Prove that the \( n + \log_2 n + O(1) \) bound is optimal.

(c1) Find the Max and the Min using \( 3n/2 + O(1) \) comparisons.

(c2) (CHALLENGE) Prove that the \( 3n/2 + O(1) \) bound is optimal.

2.8 (Strassen’s matrix multiplication) (DO) Naive multiplication of two \( n \times n \) matrices takes \( n^3 \) multiplications of numbers: we need to calculate all products of the form \( a_{ij}b_{jk} \). In 1969, Volker Strassen found a faster method. Strassen reduced the case of multiplying two \( n \times n \) matrices to 7 instances of multiplying \( (n/2) \times (n/2) \) matrices, with \( O(n^2) \) overhead that includes a constant number of matrix additions. Write down the resulting recurrence (a) ignoring the overhead and (b) adding an \( O(n^2) \) term for the overhead. Solve the resulting recurrence in the same way as we analysed the Karatsuba–Ofman algorithm.

2.9 (DO) Prove: (\( \forall \epsilon > 0 \)(\( \ln n = o(n^\epsilon) \)).

2.10 (DO) Prove that exponential growth beats polynomial growth: (\( \forall C, \epsilon > 0 \)(\( n^C = o(e^{n^\epsilon}) \))

\(^1\)Clarification “and then applying \( A \)” was added on 1-15 at 8:30 pm.