15-451 Algorithms, Fall 2012

Homework # 1 **Due: September 11, 2012**

Please hand in each problem on a separate sheet and put your **name** and **recitation** (time or letter) at the top of each page. You will be handing each problem into a separate box in lecture, and we will then give homeworks back in recitation.

Remember: written homeworks are to be done *individually*. Group work is only for the oral-presentation assignments.

Problems:

(25 pts) 1. **Recurrences.** Solve the following recurrences, giving your answer in Θ notation. For each of them, assume the base case $T(x) = 1$ for $x \leq 10$. Show your work.

(a) $T(n) = T(n-2) + n^3$.

(b) $T(n) = T(n/2) + \lg(n)$.

(c) $T(n) = 7T(n-2)$.

(d) $T(n) = \sqrt{n}\, T(\sqrt{n}) + n$. (E.g., we might get this from a divide-and-conquer procedure that uses linear time to break the problem into $\sqrt{n}$ pieces of size $\sqrt{n}$ each. Hint: write out the recursion tree.)

(25 pts) 2. **Recurrences, part 2.** Consider the following algorithm to sort an array of size n:

If $(n \leq 2)$ then sort with at most 1 comparison. Else:

1. Recursively sort the first $\lceil 2n/3 \rceil$ elements of the array.
2. Recursively sort the last $\lceil 2n/3 \rceil$ elements of the array.
3. Recursively sort the first $\lceil 2n/3 \rceil$ elements of the array.

For this question, you will be analyzing its runtime. Unlike other examples in this course, in this problem we will specifically tackle the problem of how to deal with the pesky $\lceil$ceilings$\rceil$.[1]

(a) Given that the runtime of this algorithm follows the recurrence $T(n) = 3T(2n/3)$ for $n > 2$ (with $T(2) = 1$), what is the running time of this algorithm in $\Theta()$ notation? Don't worry about rounding yet.

(b) Now, we tackle the issue with rounding. Suppose we have a sequence of numbers x_k and y_k for $k = 0, 1, 2, \ldots$ defined recursively as follows:

i. $x_0 = y_0$

ii. $x_k = (2/3)x_{k-1}$ for all $k > 0$

iii. $y_k = \lceil (2/3)y_{k-1} \rceil$ for all $k > 0$

Prove that there exists a constant c such that $y_k - x_k \leq c$ for all $k \geq 0$. (Note: This constant should be independent of x_0 and y_0.)

[1] In most examples we see in this course, we can just increase the size of the input so that the division stays integral all the way down. But you can't just assume n is a power of 3/2, since that is not an integer.

(c) Use (b) to analyze the actual recurrence of $T(n) = 3T(\lceil 2n/3 \rceil)$ for $n > 2$, giving your answer in $\Theta()$ notation.

Hint: to do this, analyze the depth of the algorithm's recursion tree. Specifically, give an upper bound u and lower bound ℓ (with $u - \ell \leq c'$ for some constant c') on the number of iterations of performing "$n \to \lceil 2n/3 \rceil$" that are needed to bring n down to 2. Once you have computed these quantities u and ℓ, the solution to the recurrence should be immediate.

(10 pts) 3. **Probability and expectation.** An *inversion* in an array $A = [a_1, a_2, \ldots, a_n]$ is a pair (a_i, a_j) such that $i < j$ but $a_i > a_j$. For example, in the array $[4, 2, 5, 3]$ there are three inversions. A sorted array has no inversions, and more generally, the number of inversions is a measure of how "well-sorted" an array is.

(a) What is the *expected* number of inversions in a random array of n elements? By "random array" we mean a random permutation of n distinct elements $a_1, \ldots, a_n$. Show your work. Hint: use linearity of expectation.

(b) It turns out that the number of comparisons made by the Insertion-Sort sorting algorithm is between I and $n + I - 1$, where I is the number of inversions in the array. Given this fact, what does your answer to part (a) say about the average-case running time of Insertion Sort (in Θ notation)?

(40 pts) 4. **Matrix games.**

The Algo Rhythms music company is expanding its business in the United States, and is going to open a new store in Atlanta, Baltimore, or Chicago (Al, the CEO, wants to keep the location secret for now). Because of the associated spike in ticket sales, the Logger Rhythms rock band is thinking about doing a concert in one of those cities right after the grand opening. Since sales have been down this year, they can't wait until after the location is announced to plan their concert.[2]

In order to precisely describe what is going on, Bob, the lead guitarist, writes down a matrix with the rows corresponding to different cities to hold their concert, the columns corresponding to different possible locations for the store, and filling in the matrix with the total amount of anticipated ticket sales. Specifically, the matrix is as follows:

	Store Location: A	B	C
Concert Location: A	5	1	1
B	1	5	1
C	1	1	5

← anticipated sales (100s of tickets)

(a) Suppose that Bob has a sister named Alice who works at Algo Rhythms, and overheard some execs talking about the new location. From what she heard, Alice believes that there is a 35% chance the store will be in Atlanta, a 20% chance it will be in Baltimore, and a 45% chance it will be in Chicago. In that case, what is the best place for Bob to choose for his concert, and what is the expected amount in sales for that location?

[2] Any relation to actual companies or bands of these names is purely coincidental.

(b) Suppose instead Bob suspects that Al is secretly trying to make his band go bankrupt. Because of this, Bob decides to randomize in making his decision.

i. With what probabilities should Bob choose a location so that no matter what city Al selects, Bob's expected ticket sales are as good as possible? (I.e., Bob wants

$$\min_{\text{Al's choices}} \mathbf{E}_{\text{Bob's choices}}[\text{ticket sales}]$$

to be as high as possible).

ii. What is the expected sales under this distribution?

(c) Suppose that because of a recent music festival, the anticipated ticket sales result in the following improved matrix (the only change is the upper-left-corner):

		Store Location		
		A	B	C
Concert Location	A	7	1	1
	B	1	5	1
	C	1	1	5

← anticipated sales (100s of tickets)

i. *Now* what probabilities should Bob use for his locations so that the worst-case expected sales ($\min_{\text{Al's choices}} \mathbf{E}_{\text{Bob's choices}}[\text{ticket sales}]$) are as high as possible? Hint: by symmetry, you can assume Bob's probabilities on B and C are equal. So, for some value p, Bob has probability p on B, p on C, and $1 - 2p$ on A.

ii. And what *is* the value of Bob's expected sales when using those probabilities?

iii. Was there anything unexpected about the probabilities you came up with in part (c)i? [There is no right or wrong answer here - but was your intuition that A should have higher or lower probability than B and C?]

(d) Considering the matrix in part (c) above, suppose now *Al* is going to probabilistically decide where to open the new store.

i. What probabilities should Al use so that no matter which city Bob picks, Bob's expected sales are as *low* as possible? (Formally,

$$\max_{\text{Bob's choices}} \mathbf{E}_{\text{Al's choices}}[\text{ticket sales}]$$

is as low as possible?)

ii. And what is the value of Bob's expected sales in this case?

The above is an example of something called a "matrix game" or a "2-player zero-sum game" between players Al and Bob. If you did this correctly, the values you computed in parts c(ii) and d(ii) should be identical. The fact that they are identical is a case of von Neumann's minimax theorem (which we will discuss later in the course), and this value is called the *value* of the game. In class, when we talk about giving "upper bounds" and "lower bounds" for a problem like sorting, we are really talking about upper and lower bounds on the value of the associated game. A randomized upper bound of $O(n \log n)$ means we have an algorithm that guarantees expected cost $O(n \log n)$ no matter what input it is given. A lower bound of $\Omega(n \log n)$ means that no algorithm can do better, which we typically will show by giving an adversarial probability distribution on inputs (an "algorithm for Al") that is bad for all algorithms.