Massachusetts Institute of Technology — Handout 24
6.042J/18.062J: Mathematics for Computer Science — October 6, 1998
Professors Nancy Lynch and Srinivas Devadas

# Problem Set 5

**Due: Start of class on October 15, 1988.**

**Read sections 1.7, 1.8, 5.1, 5.2, and 5.3 in Rosen.**

**Problem 1.** **(15 points)** In 1997 there was a breakthrough on a long-standing problem in computer science called The Traveling Salesman Problem. The idea is that there is a salesman who wants to visit $n$ cities scattered around the country. The problem is to determine the order in which he should visit the cities so as to minimize the total distance that he travels, assuming that he can go in a straight line between any two cities. Surprisingly, this problem is very difficult when the number of cities $n$ is large.

The breakthrough was an algorithm that efficiently computes a nearly-optimal solution, a route for the salesman that has length at most 10% greater than the shortest possible route. The running time of the algorithm is described in the orginal publication as "nearly linear" in the number of cities. More precisely, the algorithm requires about $n \ln^{400} n$ steps.

**(a)** Prove that $n \ln^{400} n$ is $o(n^{1.1})$. This shows that the algorithm truly is "nearly linear" in an asymptotic sense. You may make use of L'Hopital's rule:

$$\text{If } \lim_{n\to\infty} f(n) = \infty \text{ and } \lim_{n\to\infty} g(n) = \infty, \text{ then } \lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{f'(n)}{g'(n)}.$$

**(b)** Find a specific value of $n$ such that $n^{1.1}$ is actually greater than $n \ln^{400} n$.

**Problem 2.** **(10 points)** Order the functions in each set below so that each function is either $O$ or $\Theta$ of the next function. Indicate whether the stronger $\Theta$ relationship holds in each case. No proofs are required.

**Example.** $n^2$, $n$, $n^3/(n+1)$, $3n^2$

$$\begin{aligned} n &= O(n^2) \\ n^2 &= \Theta(3n^2) \\ 3n^2 &= \Theta(n^3/(n+1)) \end{aligned}$$

**(a)** $\ln n, \ln \sqrt{n}, \sqrt{\ln n}, n^{1/n}$

**(b)** $(\sqrt{n})^{2 \ln \ln n}, n, n \ln n, \ln n^{\ln n}, \ln n!$

**(c)** $n!, n^n, (n/3)^n, n^{n + \ln \ln n}, e^{n \ln \ln n}$

**Problem 3. (10 points)** Suppose that $n$ ovals are drawn in a plane so that every pair of ovals intersects at either 0 or 2 points. Let $x_n$ be the maximum number of regions that can be formed by such a configuration of ovals. We can express $x_n$ using the recurrence below.

$$\begin{aligned} x_1 &= 2 \\ x_{n+1} &= x_n + 2n \end{aligned}$$

The region outside of any oval "counts"; for example, one oval defines two regions, because the oval has an inside region and an outside region.

**(a)** Solve this recurrence using the plug and chug method.

**(b)** Describe how to arrange $n$ ovals so as to form $x_n$ regions. No proof of correctness is required.

**Problem 4. (10 points)** Consider the following recurrence.

$$\begin{aligned} y_0 &= 1 \\ y_n &= \sum_{i=1}^{n} y_{n-i} \cdot 2^i \end{aligned}$$

**(a)** Guess the solution to the recurrence.

**(b)** Prove your guess correct using strong induction.

**Problem 5. (10 points)** The following recurrence was invented by a late Italian renaissance scholar named Giardi Invertabinacci, who earned a certain repute in his age for mocking profound ideas, poetry, sculpture, and even architecture by constructing upside-down or backward replicas of the orignal. His scaled model of the famed Tower of Pisa still stands– upside down, but not leaning– outside Palermo. His one venture into mathematics was a recurrence relation, recorded in a letter to a contemporary mathematician and reproduced below in modern notation.[1]

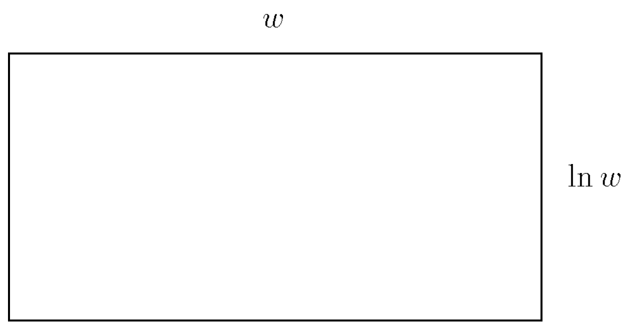
$$\begin{aligned} x_1 &= 1 \\ x_2 &= 1 \\ x_{n+1} &= \frac{x_n x_{n-1}}{x_n + x_{n-1}} \end{aligned}$$

[1] Okay, your TA just made all this up. But good story, eh?

**(a)** Guess the solution to the recurrence. Your solution may be expressed in terms of Fibonacci numbers.

**(b)** Use strong induction to prove your guess correct.

**Problem 6.** **(20 points)** Recently, Theory Pig found a small piece of blue crayon behind the filing cabinet in Professor Lynch's office. Theory Pig was delighted to find that with this wondrous new toy he could draw pretty blue rectangles with a width of $w$ centimeters and a height of $\ln w$ centimeters for any $w > 1$ that he took a fancy to.




Our pal proudly presented this remarkable discovery to 6.042 TA Rodrigo Leroux. Rodrigo was unimpressed: "Yeah, but what if I want a rectangle with an area of $n$ square centimeters? Then what should be the width $w$?" Fidgeting nervously with his crayon, Theory Pig replied that he believed the width should be $\Theta(n/\ln n)$ asymptotically, but that unfortunately he could not rigorously prove this fact. Rodrigo replied, "If you can't play with your toys properly, then they'll be taken away!" And then he confiscated Theory Pig's crayon!

Return joy to our little pal's world by supplying a proof of his claim.

**Problem 7.** **(10 points)** We have devised a new version of `MergeSort`, which we call `MergeSortX19`, for the intellectually challenged inhabitants of planet Xoros 19.

Here is how the new algorithm works. The input is a list of $n$ distinct numbers. If the list contains a single number, then there is nothing to do. If the list contains two numbers, then we sort them with a single comparison. If the list contains more than two numbers, then we perform the following sequence of steps.

- We make a list containing the first $\frac{2}{3}n$ numbers and sort it recursively.
- We make a list containing the last $\frac{2}{3}n$ numbers and sort it recursively.

- We make a list containing the first $\frac{1}{3}n$ numbers and the last $\frac{1}{3}n$ numbers and sort it recursively.
- We merge the first and second lists, throwing out duplicates.
- We merge this combined list with the third list, again throwing out duplicates.

The final, merged list is the output. We consider this algorithm ideal for the slow-witted creatures of Xoros 19, because multiple copies of each number are maintained. Therefore, even if the sorter occasionally forgets about a number, the algorithm can still output a complete, sorted list.

**(a)** Write a recurrence for the maximum number of comparisons needed to sort a list of $n$ numbers using `MergeSortX19`. Assume that merging a list with $l_1$ numbers and a list with $l_2$ numbers and throwing out duplicates requires at most $2(l_1 + l_2)$ comparisons.

**(b)** Give a $\Theta$ bound on the maximum number of comparisons used by `MergeSortX19` to sort a list of $n$ numbers.

**Problem 8. (15 points)** Use the Akra-Bazzi method to find a $\Theta$ bound on the solution to each of the following recurrences.

**(a)** $T(n) = 4T(\lfloor \frac{n}{3} \rfloor) + n^2$

**(b)** $T(n) = T(\lfloor \frac{3n}{5} \rfloor) + T(\lfloor \frac{4n}{5} \rfloor) + n^2$

**(c)** $T(n) = T(\lfloor \frac{n}{3} \rfloor) + 2T(\lfloor \frac{n}{4} \rfloor) + n\sqrt{n}$