CS 170 Efficient Algorithms and Intractable Problems
Spring 2016 Alessandro Chiesa and Umesh Vazirani

HW 1

Due January 25, 11:59pm

Instructions: You are welcome to form small groups (up to 4 people total) to work through the homework, but you **must** write up all solutions by yourself. List your study partners for homework on the first page, or "none" if you had no partners.

Begin each problem on a new page. Clearly label where each problem and subproblem begin. The problems must be submitted in order (all of P1 must be before P2, etc).

For questions asking you to give an algorithm, you must respond in what we will refer to as the *four-part format* for algorithms: high-level description, pseudocode, running time analysis, and proof of correctness.

Read the **Homework FAQ** post on Piazza before doing the homework for more explanation on the four-part format and other clarifications for our homework expectations.

You risk receiving no credit for any homework that does not adhere to these guidelines.

No late homeworks will be accepted. **No exceptions.** Do not ask for extensions. This is not out of a desire to be harsh, but rather out of fairness to all students in this large course. Out of a total of approximately 13 homework assignments, the lowest two scores will be dropped.

This homework is due Monday, January 25, at 11:59pm via Gradescope. Please submit via PDF.

1. (15 pts.) Compare Growth Rates

In each of the following, indicate whether $f = O(g)$, $f = \Omega(g)$, or both (in which case $f = \Theta(g)$). Briefly justify each of your answers.

	$f(n)$	$g(n)$
(a)	$n^{1/2}$	$n^{2/3}$
(b)	$\log_3 n$	$\log_4 n$
(c)	$2^n + n^5$	$2^n + \log n$
(d)	$n \log(n^4)$	$n^2 \log(n^3)$
(e)	$n^{1.01}$	$n \log^2 n$
(f)	$n \log n$	$(\log n)^{\log n}$
(g)	$\sqrt{n}$	$(\log n)^3$
(h)	2^n	2^{n+1}
(i)	4^n	$n!$

2. (15 pts.) Geometric Growth

Prove that the following formula holds, given a positive real number c.

$$\sum_{i=0}^{k} c^i = \begin{cases} \Theta(c^k) & \text{if } c > 1 \\ \Theta(k) & \text{if } c = 1 \\ \Theta(1) & \text{if } c < 1 \end{cases}$$

The moral: in asymptotics, the sum of a geometric series is simply the first term if the series is strictly decreasing, the last term if the series is strictly increasing, or the number of terms if the series is unchanging. This idea underlies the Master theorem for solving recurrences.

3. (25 pts.) Recurrence Relations

Solve the following recurrence relations and give a Θ bound for each of them.

(a) (i) $T(n) = 3T(n/4) + 4n$

(ii) $T(n) = 45T(n/3) + .1n^3$

(iii) $T(n) = T(n-1) + c^n$, where c is a constant.

(iv) $T(n) = 2T(\sqrt{n}) + 3$, and $T(2) = 3$. (Hint: this means the recursion tree stops when the problem size is 2)

(b) (i) Consider the recurrence relation $T(n) = 2T(n/2) + n \log n$. We can't plug it directly into the Master theorem, so solve it by adding the size of each layer.
Hint: split up the $\log(n/(2^i))$ *terms into* $\log n - \log(2^i)$*, and use the formula for arithmetic series.*

(ii) A more general version of Master theorem, like the one on Wikipedia, incorporates this result. The case of the master theorem which applies to this problem is:
If $T(n) = aT(n/b) + f(n)$ *where* $a \geq 1$, $b > 1$, *and* $f(n) = \Theta(n^c \log^k n)$ *where* $c = \log_b a$, *then* $T(n) = \Theta(n^c \log^{k+1} n)$.
Use the general Master theorem to solve the following recurrence relation:
$T(n) = 9T(n/3) + n^2 \log^3 n$.

4. (25 pts.) Integer Multiplication In class, we covered the integer multiplication technique used by Al Khwarizmi and some European countries. With this method, we multiply and divide numbers by 2 at each iteration. However, there's nothing special about the number 2 here – we could perform this technique by multiplying and dividing another number instead. In particular, let's modify this technique to multiply and divide the numbers by 3 at each iteration. Following the format described in the book, we can create two columns, one for each number being multiplied. For each row, we divide the number in the left column by 3, ignoring the remainder, and multiply the number in the right column by 3. To get the final result, we need to add rows of the right column up. Which rows should be added together to get the final result?

Hint: you may have to multiply some rows by a constant.

(a) Show how to multiply the numbers 11 and 13 using your modified multiplication scheme.

(b) Formally describe and prove the correctness of your modified algorithm. Use the four-part format for algorithms (High-level description, pseudocode, running time analysis, and proof of correctness). Your proof of correctness may be brief.

5. (20 pts.) More Integer Multiplication The following algebraic identities can be used to design a divide-and-conquer algorithm for multiplying n-bit numbers. Assume we are multiplying two numbers a and b. Let a_2, a_1, and a_0 be the top $n/3$, middle $n/3$, and bottom $n/3$ bits of a respectively. Define b_2, b_1, and b_0 in the same way. We can write the product $a \cdot b$ as

$$\begin{aligned} a \cdot b &= \left(2^{2n/3} a_2 + 2^{n/3} a_1 + a_0\right)\left(2^{2n/3} b_2 + 2^{n/3} b_1 + b_0\right) \\ &= 2^{4n/3} \cdot a_2 b_2 + 2^n \cdot (a_1 b_2 + a_2 b_1) + 2^{2n/3} \cdot ((a_2 b_0 + a_0 b_2) + a_1 b_1) + 2^{n/3} \cdot (a_1 b_0 + a_0 b_1) + a_0 b_0 \end{aligned}$$

Furthermore, we know the terms in the above expression can be written as follows:

$$\begin{aligned} (a_1 b_2 + a_2 b_1) &= (a_1 + a_2) \cdot (b_1 + b_2) - a_1 b_1 - a_2 b_2 \\ (a_0 b_2 + a_2 b_0) &= (a_0 + a_2) \cdot (b_0 + b_2) - a_0 b_0 - a_2 b_2 \\ (a_0 b_1 + a_1 b_0) &= (a_0 + a_1) \cdot (b_0 + b_1) - a_0 b_0 - a_1 b_1 \end{aligned}$$

Using this, we can derive a recursive multiplication algorithm.

(a) Rewrite the product $a \cdot b$ in terms of as few distinct recursive multiplications as possible (i.e. one such recursive multiplication might be the product $a_0 \cdot b_0$.

(b) Write the recurrence relation for running time $T(n)$ if we perform multiplications in this way.

(c) What is the running time of the algorithm?