CMPS 2200 Introduction to Algorithms – Fall 15

9/30/15

5. Homework

Due **10/8/15** at the beginning of the lab

1. **Multiplying polynomials (10 points)**
   A polynomial of degree n is a function

$$p(x) = \sum_{i=0}^{n} a_i x^i = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_3 x^3 + a_2 x^2 + a_1 x + a_0$$

   where the coefficients a_i are constants and $a_n \neq 0$. For example, $4x^3 + 5x^2 + 2x + 1$ is a polynomial of degree 3. For simplicity you may assume that $n + 1$ is a power of 2.

   A polynomial of degree n can be stored in the computer by storing an array of its coefficients. The coefficient array has length $n + 1$.

   (a) (+ 1 extra credit) Suppose you are given a degree-n polynomial $p(x)$ as its coefficient array $a_n, a_{n-1}, \ldots, a_1, a_0$. Now suppose you are given a value x_0. Give an algorithm to compute $p(x_0)$ in $O(n)$ time.

   (b) (1 point) Now, suppose you are given two degree-n polynomials $p(x)$ and $q(x)$. What is the runtime of the straight-forward algorithm to multiply both polynomials, i.e., to compute one coeffcient array that represents $p(x) \cdot q(x)$? For example, if $p(x) = 3x^2 - 2x + 4$ and $q(x) = 5x + 3$ then $p(x) \cdot q(x) = (3x^2 - 2x + 4) \cdot (5x + 3) = 15x^3 - 10x^2 + 20x + 9x^2 - 6x + 12 = 15x^3 - x^2 + 14x + 12$, so, the resulting coefficient array is $15, -1, 14, 12$.

   (c) (5 points) The polynomial

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_3 x^3 + a_2 x^2 + a_1 x + a_0 \ .$$

   can be rewritten as

$$x^{\frac{n+1}{2}} (a_n x^{\frac{n-1}{2}} + a_{n-1} x^{\frac{n-1}{2}-1} + \ldots + a_{\frac{n-1}{2}+2} x + a_{\frac{n-1}{2}+1}) + (a_{\frac{n-1}{2}} x^{\frac{n}{2}-1} + \ldots + a_1 x + a_0) \ .$$

   In other words, $p(x) = x^{\frac{n+1}{2}} p_1(x) + p_2(x)$, which effectively divides p into two parts of half the size each. For example, $4x^3 + 5x^2 + 2x + 1$ can be rewritten as $x^2(4x + 5) + (2x + 1)$; here $n = 3$. And $7x^7 + 6x^6 + 5x^5 + 4x^4 + 3x^3 + 2x^2 + x + 9$ can be rewritten as $x^4(7x^3 + 6x^2 + 5x + 4) + (3x^3 + 2x^2 + x + 9)$; here $n = 7$.
   Use this as a starting point to design a divide-and-conquer algorithm for multiplying two degree-n polynomials (recurse on polynomials of degree $(n - 1)/2$). Give a runtime analysis of your algorithm by setting up and solving a recurrence. The runtime of your algorithm should be the same as the runtime of part (b).

FLIP OVER TO BACK PAGE ⟹

(d) (1 point) Show how to multiply two degree-1 polynomials $ax+b$ and $cx+d$ using only three multiplications of coefficients. *Hint: One of the multiplications is* $(a+b)\cdot(c+d)$.

(e) (3 points) Design a divide-and-conquer algorithm for multiplying two polynomials of degree n in time $\Theta(n^{\log_2 3})$. Justify the running time. *(Hint: Reuse part* **(c)** *and speed it up with the knowledge of part* **(d)***)*

2. **Master theorem (12 points)**
   Use the master theorem to solve the following recurrences. Justify your results. Assume that $T(1) = 1$.

   (a) $T(n) = 16T(\frac{n}{2}) + n^2$

   (b) $T(n) = T(\frac{n}{2}) + \sqrt{n}$

   (c) $T(n) = 16T(\frac{n}{4}) + n^2 \log n$

   (d) $T(n) = 81T(\frac{n}{3}) + n^2 \log n$