

Homework Assignment #2

Due: October 19, 2010, by 10:30 am

(in the drop box for your CSCC73 tutorial section, near room SW-626A)

*Appended to this document is a cover page for your assignment. Fill it out, staple your answers to it, and deposit the resulting document into the course drop box (without putting it in an envelope).*

**Question 1.** (10 marks) Solve Exercise 1 on page 246 of the textbook by Kleinberg and Tardös. You should not only describe your algorithm, but also explain why it works and why it achieves the required bound of  $O(\log n)$  queries.

**Question 2.** (10 marks) The standard algorithm for merging two sorted lists requires time proportional to the size of the resulting list. Suppose we are given  $k$  sorted lists, each containing  $n$  elements. We want to merge these into a single sorted list.

**a.** (5 marks) Here is one algorithm to solve our problem: We merge the first two lists; then we merge the resulting list with the third list; then we merge the resulting list with the fourth list, etc. What is the time complexity of this algorithm in terms of  $k$  and  $n$ ?

**b.** (5 marks) Give a more efficient divide-and-conquer algorithm to solve this problem. What is the time complexity of your algorithm?

**Question 3.** (10 marks) If  $G = (V, E)$  is a graph and  $V' \subseteq V$ , the *subgraph of  $G$  induced by  $V'$*  is the graph  $G' = (V', E')$ , where  $E' = \{(u, v) \in E : u, v \in V'\}$ . Intuitively, this is the graph we obtain from  $G$  if we delete all nodes except those in  $V'$  and all edges that have (at least) one endpoint outside of  $V'$ .

Professor N. O'Bright suggests the following divide-and-conquer algorithm to compute the MST of a connected, undirected graph  $G = (V, E)$  with edge costs  $c(e)$  for each  $e \in E$ . If  $G$  consists of a single node, the MST trivially has no edges. Otherwise, we partition the set of nodes of  $G$  arbitrarily into two sets  $V_1$  and  $V_2$  of about the same size; we recursively compute the MSTs of the subgraphs of  $G$  induced by  $V_1$  and  $V_2$ ; and we join these two MSTs by a minimum cost edge that crosses the  $V_1/V_2$  cut. More precisely, the algorithm that N. O'Bright proposes is this:

```
D&C-MST( $G$ )
  if  $|V| = 1$  then return  $\emptyset$ 
  else
    let  $V_1 \subseteq V$  be such that  $|V_1| = \lceil |V|/2 \rceil$ , and  $V_2 := V - V_1$ 
    let  $G_1$  and  $G_2$  be the subgraphs of  $G$  induced by  $V_1$  and  $V_2$ , respectively
    let  $e$  be a minimum cost edge connecting a node in  $V_1$  to a node in  $V_2$ 
    return D&C-MST( $G_1$ )  $\cup$  D&C-MST( $G_2$ )  $\cup$   $\{e\}$ 
```

Is this algorithm correct? If so, prove it; otherwise, provide a counterexample.

**Question 4.** (5 marks) Demonstrate the polynomial multiplication algorithm that uses the FFT by using it to multiply the polynomials  $1 + x + 2x^2$  and  $2 + 3x$ : Use the FFT to evaluate each polynomial at an appropriate set of points, and show the resulting vectors; multiply the vectors component-wise; and use the inverse FFT to obtain the coefficients of the final result. Show your work.

(continued)

**Question 5.** (15 marks) In class we discussed Karatsuba's divide-and-conquer algorithm for integer multiplication, which multiplies  $n$ -bit numbers by recursively multiplying  $n/2$  bit numbers (see Section 5.5 of the textbook). In this problem we show that it is possible to get an asymptotically faster algorithm by recursively multiplying  $n/3$ -bit numbers. To do this, we use ideas about fast multiplication of polynomials given their coefficients (see KT §5.6). In particular, we want to multiply two degree-2 polynomials in five (rather than the obvious nine) multiplications. Because here we'll be working with polynomials of fixed degree, we do not have to consider complex numbers as in KT §5.6.

**a.** (5 marks) We first consider *polynomial evaluation*. Our input consists of the three coefficients of a degree-2 polynomial  $p(z) = a_2z^2 + a_1z + a_0$ , and we want to find the value of this polynomial at five integers  $z = 0, 1, 2, 3, 4$

Give a  $3 \times 5$  matrix  $M_1$  such that for every row vector  $\vec{a} = (a_0, a_1, a_2)$ ,

$$\vec{a} \cdot M_1 = \vec{v}$$

where  $\vec{v} = (p(0), p(1), p(2), p(3), p(4))$  — i.e., the row vector consisting of the five values of  $p(z)$  at  $z = 0, 1, 2, 3, 4$ .

**b.** (5 marks) We next consider *polynomial interpolation*. Our input consists of the values of a degree-4 polynomial  $p(z)$  at  $z = 0, 1, 2, 3, 4$ , and we want to find the coefficients of  $p(z)$ . That is, our input is a row vector  $\vec{v} = (v_0, v_1, v_2, v_3, v_4)$ , and we wish to find the coefficients  $\vec{a} = (a_0, a_1, a_2, a_3, a_4)$  of a polynomial  $p(z) = a_4z^4 + a_3z^3 + a_2z^2 + a_1z + a_0$  such that  $(v_0, v_1, v_2, v_3, v_4) = (p(0), p(1), p(2), p(3), p(4))$ .

Give a  $5 \times 5$  matrix  $M_2$  of rational numbers such that for every  $\vec{v} = (v_0, v_1, v_2, v_3, v_4)$ , if we compute  $\vec{a} = \vec{v} \cdot M_2$  and let  $(a_0, a_1, a_2, a_3, a_4) = \vec{a}$  and  $p(z) = a_4z^4 + a_3z^3 + a_2z^2 + a_1z + a_0$ , then  $(v_0, v_1, v_2, v_3, v_4) = (p(0), p(1), p(2), p(3), p(4))$ .

**Hint.** Consider part (a) and find the inverse of an appropriate matrix. You may use software (e.g., Mathematica) or a website (e.g., <http://www.math.odu.edu/~bogacki/lat/>) to help you with your calculations, but be sure to credit your source.

**c.** (5 marks) We now consider a divide-and-conquer algorithm for integer multiplication, where we multiply two  $n$ -bit numbers by recursively multiplying *five*  $n/3$ -bit numbers.

Suppose we are given two  $n$ -bit numbers  $A$  and  $B$ ; for convenience, assume that  $n$  is a power of 3. Let  $a_0, a_1, a_2$  be the integers comprising the rightmost third, middle third, and leftmost third of  $A$ ; let  $b_0, b_1, b_2$  be defined similarly for  $B$ . We then have

$$A = a_2(2^{n/3})^2 + a_12^{n/3} + a_0 \text{ and } B = b_2(2^{n/3})^2 + b_12^{n/3} + b_0.$$

Our goal is to compute

$$C = A \cdot B = c_4(2^{n/3})^4 + c_3(2^{n/3})^3 + c_2(2^{n/3})^2 + c_12^{n/3} + c_0$$

where  $(c_0, c_1, c_2, c_3, c_4)$  are the coefficients of the degree-4 polynomial obtained by multiplying the two degree-2 polynomials  $p(z) = a_2z^2 + a_1z + a_0$  and  $q(z) = b_2z^2 + b_1z + b_0$ .

Using the matrices  $M_1$  and  $M_2$  from parts (a) and (b), give the details of this algorithm. Give a recurrence that describes its time complexity, and explain why it does. Finally, explain why this algorithm is (asymptotically) faster than Karatsuba's algorithm discussed in class.

**Question 6 — EXTRA CREDIT.** (10 marks) Consider the following modification of the randomized algorithm to select the  $k$ th smallest element in a given list of numbers: Instead of randomly picking the splitter from the entire list, we randomly pick the splitter from the *first half* of the list.

Does the resulting algorithm still run in expected linear time? Carefully justify your answer.

# Cover page for CSCC73 Assignment #2

Submitted by

(1) Family Name: \_\_\_\_\_

(2) Family Name: \_\_\_\_\_

Given Name: \_\_\_\_\_

Given Name: \_\_\_\_\_

Student Number: \_\_\_\_\_

Student Number: \_\_\_\_\_

*By virtue of submitting this homework I/we acknowledge that I am/we are aware of the policy on homework collaboration for this course.*