KTH Computer Science and Communication

Homework III, Complexity Theory 2008

Due on May 20 at 10.15. The general rules on homework solutions available at the course homepage apply. In particular, discussions of ideas in groups of up to at most three people are allowed but solutions should be written down individually.

1 (15p) In class we showed that if we have two polynomials P and Q both of degree d in n variables such that $P \neq Q$ then

$$Pr[P(x) = Q(x)] \leq \frac{nd}{2R}$$

if we chose each x_i as a random integer in the range $[-R, R]$ and evaluate the polynomials over the rational numbers. We concluded that picking $R = 2nd$ is sufficient to get error probability $1/4$. Show that this is essentially sharp for the case $n = d$ by finding two unequal polynomials P and Q such that if you pick the values of the variables from a range that is significantly smaller than n^2 then you are likely to get equality. Note that degree is counted as maximal degree in a single variable and hence a polynomial like $x_1^3 x_2 + x_3^3 x_4 x_5$ has degree 3.

2 (15p) Show that any function of n bits can be computed by a circuit of size $O(2^n/n)$.

Hint: First compute all functions of the first, approximately $\log n$, bits, and then use the results many times.

3 (15p) In class we made a $O(\log n)$ depth circuit for multiplication using $\Omega(n^2)$ gates. Let us see if we can make it smaller. Recall the algorithm by Karatsuba which runs in time $O(n^{\log 3})$ and (recursively) uses the idea of replacing a multiplication of n-bit numbers by three multiplications of $n/2$ bit numbers (please consult any source if you need details of the algorithm). Can this algorithm be implemented by a circuit of size $O(n^{\log 3})$ and depth which is polylogarithmic? What power of the logarithm do you need for the depth? Of course we want as small depth as possible.

4 (40p) Let us study read-once branching programs for multiplication and addition. A branching program that outputs many bits is allowed to give the output bits one at the time and is not obliged to wait to read the entire input before it produces output. To be more precise any transition in the branching program can carry a label of the type "$z_i = b$" for some index i and value b with the intended meaning that the i'th bit of the output is b. It is supposed to output exactly one value of each z_i. Note that the branching program is read-once and thus each input may only be read one time.

You are given two numbers x and y where the bit representation of x is $x_{n-1} \ldots x_1 x_0$ where x_0 is the least significant bit with and we have similar notation for y. Suppose you have to read the bits in the order $x_0, y_0, x_1, y_1, \ldots x_{n-1}, y_{n-1}$.

4a (10p) Give a polynomial size read-once branching program for addition with the given ordering of the input.

4b (20p) Prove for some fast growing function[1] $S(n)$ that multiplication requires read-once branching programs of size at least $S(n)$ with the given variable ordering. Speculate whether you think any other ordering might be better. No proof is needed to support your speculations.

4c (15p) What happens with the addition problem if you are given the bits in the reverse order? Do you still have a small branching program?

5 (20p) In class we discussed different types of PRAMs and in particular to what extent different processors could access the same memory cell simultaneously. We mentioned the types EREW (Exclusive Read, Exclusive Write), CREW (Common Read, Exclusive Write) and CRCW (Common Read, Common Write). Within the last type we have subtypes depending on whether we require that processors, when writing into the same memory cell, write the same value. Let us, however, only consider the subtype where the processors might write different values and some arbitrary processor succeeds. This gives us a total of three types of PRAMs. Now each pair of types can be compared using simulation results and separation results. A simulation result is of the type

> One step of a PRAM of type A with P processors can be simulated by a PRAM of type B with S steps and T_2 processors.

for some T_2 related to P (usually in a polynomial way). A separation result is usually of the form

> There is a problem that can be solved on a PRAM of type B with a polynomial number of processors in T steps while it requires $S'T$ steps on a PRAM of type A with a polynomial number of processors.

If S and S' are close then we have an essentially tight separation of the two types. Search the literature to find as tight as possible separations between

5a EREW and CREW.

5b EREW and CRCW

5c CREW and CRCW

You need not understand the results beyond repeating the exact results. You need only look at results with a polynomial number of processors and memory cells.

6 (10p) As discussed in class I do not know the answer to the following problem and this makes it more interesting.

What algorithm for addition and multiplication is used by modern computer hard-ware?

[1] Maybe try $S(n) = 2^n$.