Complexity Theory

Homework Sheet 5

(Turn in before the exercise session of Thursday 8 Oct.)

1 October 2015

Definition 1. Let $f : \{0,1\}^n \to \{0,1\}$ be a boolean function. Let $\mathbb{F}$ be a field and let $p \in \mathbb{F}[x_1, \ldots, x_n]$ be a multivariate polynomial with coefficients in the field $\mathbb{F}$. The function f is *represented* by p if

$$p(s_1, \ldots, s_n) = f(s_1, \ldots, s_n)$$

for every $s \in \{0,1\}^n$. (For the left side of the equation we take 0 and 1 to be the additive and multiplicative identities of the field $\mathbb{F}$, respectively.) The *degree* of f over $\mathbb{F}$, written $\deg_{\mathbb{F}}(f)$, is the smallest degree of any polynomial in $\mathbb{F}[x_1, \ldots, x_n]$ representing f.

A polynomial $p \in \mathbb{F}[x_1, \ldots, x_n]$ is *multilinear* if in every monomial of p, every variable x_i occurs with power 0 or 1, that is, p is of the form

$$p = \sum_{b \in \{0,1\}^n} \alpha_b \, x_1^{b_1} x_2^{b_2} \cdots x_n^{b_n}$$

with $\alpha_b \in \mathbb{F}$.

Exercise 1. Let f be a boolean function. Let $\mathbb{F}$ be a field. Show that f can be represented by a *multilinear* polynomial $p \in \mathbb{F}[x_1, \ldots, x_n]$, and that such a p is unique.

Exercise 2. Let $\oplus_n$ denote the parity function over n bits. What is:

(a) $\deg_{\mathbb{R}}(\oplus_n)$?

(b) $\deg_{\mathbb{F}_4}(\oplus_n)$? (Hint: If you don't know what $\mathbb{F}_4$ is, read wikipedia on finite fields)

Exercise 3. Show, using polynomials, that the NAND function can not be computed by a circuit consisting only of XOR and NOT gates.

(Hint: What is the degree of XOR over fields you could consider?)

Exercise 4. Given three $n \times n$ matrices A, B, C with real entries, consider the problem of checking whether the product of A and B equals C. That is, to check whether $C_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$, for all $i, j \in \{1, \ldots, n\}$. The naïve algorithm would just compute the product AB and compare this to C, but computing it entry by entry takes time $O(n^3)$, and even the fastest known algorithm for matrix multiplication takes $O(n^{2.3729})$ steps.[1] Give a randomized algorithm that solves this problem in time $O(n^2)$, with constant one-sided error.

Hint: Multiply both AB and C with a random 0/1-vector v. Show that having a wrong C_{ij} will always be detected either by v, or by v with entry v_j negated.

[1] Finding the exact value of the exponent ω in the running time of the fastest matrix multiplication algorithm $O(n^{\omega})$ is an open problem – it is known that $2 \leq \omega < 2.3729$.