

# Problems in Algebra and Computation.

Neeraj Kayal

August 5, 2008

Just as every human undertaking pursues certain objects, so also mathematical research requires its problems. It is by the solution of problems that the investigator tests the temper of his steel; he finds new methods and new outlooks, and gains a wider and freer horizon.

-David Hilbert, International Congress of Mathematicians, 1900.

## Problems

1.  **$\mathbb{F}$ -linear dependence of polynomials.** Let  $\mathbb{F}$  be a field and  $\mathbf{x} \stackrel{\text{def}}{=} (x_1, \dots, x_n)$  be an  $n$ -tuple of indeterminates. A set of polynomials  $f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_m(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  is said to be  $\mathbb{F}$ -linearly dependent if there exist constants  $a_1, \dots, a_m \in \mathbb{F}$ , not all zero, such that

$$a_1 \cdot f_1(\mathbf{x}) + a_2 \cdot f_2(\mathbf{x}) + \dots + a_m \cdot f_m(\mathbf{x}) = 0.$$

Devise a randomized polynomial-time algorithm that given polynomials  $f_1(\mathbf{x}), \dots, f_m(\mathbf{x})$  in the form of arithmetic circuits, tests whether  $f_1, \dots, f_m$  are  $\mathbb{F}$ -linearly dependent.

2. **Radicals of matrices.** In this exercise we will devise an algorithm that given a matrix  $A$  computes its  $k$ -th radical,  $A^{\frac{1}{k}}$ . More precisely, the task is to devise an efficient randomized algorithm that given a prime  $p$ , an integer  $k \geq 2$  and an  $(n \times n)$ -matrix  $A \in \mathbb{F}_p^{n \times n}$ , computes a matrix  $B \in \mathbb{F}_p^{n \times n}$  such that  $B^k = A$ , if such a matrix  $B$  exists. Notes:

- The running time of the algorithm should be polynomial in  $(\log p \cdot n \cdot k)$ .
- It may be helpful to first devise an algorithm assuming that all the eigenvalues of  $A$  are in  $\mathbb{F}_p$  itself.

3. **GCDs of polynomials in NC.** In this exercise we show that computing the gcd of two polynomials with integer coefficients can be computed in NC. More specifically, suppose we are given two monic polynomials  $a(x)$  and  $b(x)$  of degree  $d$  with coefficients which are integers between  $-H$  and  $H$ . Thus the size of the input is  $2d \cdot \log H$ .

- i). Let  $m$  be the degree of  $\gcd(a(x), b(x))$ .  $0 \leq m \leq d$ . Then show that there exists a *unique* monic polynomial  $h(x) \in \mathbb{Q}[x]$  of degree  $m$  such that

$$f(x) \cdot a(x) + g(x) \cdot b(x) = h(x),$$

where  $f(x)$  and  $g(x)$  both have degree at most  $(d - 1)$ . Deduce that if in addition to  $a(x)$  and  $b(x)$  we are also given  $m$  then we can find the gcd of  $a(x)$  and  $b(x)$  in NC.

- ii). Show that we can compute the value of  $m$  in NC. This will show that overall, gcd of two monic polynomials with integer coefficients can be computed in NC.
4. **Perfect powers.** A positive integer  $n$  is said to be a *perfect power* if there exist two positive integers  $a$  and  $b \geq 2$  such that  $n = a^b$ . In this exercise, we show that determining whether a number is a perfect power can in fact be done in NC. In doing this, we proceed in steps and prove successively stronger results till we reach this goal.
- i). Devise a polynomial-time algorithm to test if an integer is a perfect power.
- ii). Show that in NC we can reduce the problem to the problem of determining if a number is a  $b$ -th power, where  $b$  is a *fixed* integer.
- iii). Show how to determine if a number is perfect square ( $b = 2$ ) in NC. (Hint: Use the Taylor series expansion for  $\sqrt{1+x}$ ).
- iv). Show how to determine if a number is a perfect cube ( $b = 3$ ) in NC.
- v). Generalize to arbitrary  $b$ .
5. **Elimination Ideals versus resultants.** Let  $f(x, y), g(x, y) \in \mathbb{C}[x, y]$  be any two polynomials. Let  $I_{-x}$  be the elimination ideal of  $f$  and  $g$  with respect to the variable  $x$ . That is,

$$I_{-x} \stackrel{\text{def}}{=} \langle f(x, y), g(x, y) \rangle \cap \mathbb{C}[y].$$

Let  $\rho(y) \stackrel{\text{def}}{=} \text{RES}_x(f(x, y), g(x, y))$ .

- i). Show that  $\rho(y) \in I_{-x}$ .
- ii). Prove or disprove the following conjecture: *every polynomial in the elimination ideal  $I_{-x}$  is a multiple of the resultant polynomial  $\rho(y)$* . More concisely, the conjecture is that

$$I_{-x} = \langle \rho(y) \rangle$$

## 6. Discrete Logarithms for matrices.

*Background:* The discrete logarithm problem (for elements of a field) is the following: Given a finite field  $\mathbb{F}_q$  and two elements  $\alpha, \beta \in \mathbb{F}_q$ , the task is to compute the smallest integer  $m \geq 0$  such that  $\alpha^m = \beta$ , if such an  $m$  exists. No efficient algorithm is known for this problem and indeed, there are many cryptosystems based on the presumed difficulty of solving this problem.

*The problem at hand.* A new cryptosystem has been designed whose security is based on the assumption that the discrete logarithm problem for matrices (to be precisely defined shortly) is hard, possibly harder than solving discrete logarithm over finite fields. Your task is to show that the new cryptosystem is no more secure than the known cryptosystems by showing that the discrete logarithm problem for matrices is no harder than the usual discrete logarithm problem for field elements. More precisely, the task is to devise a randomized polynomial-time algorithm for solving discrete logarithm for matrices assuming that we have an oracle for solving the usual discrete logarithm problem for field elements.

The discrete logarithm problem for matrices is the following: given a finite field  $\mathbb{F}_p$  and two  $(n \times n)$  matrices  $A, B \in \mathbb{F}_p^{n \times n}$ , find the smallest integer  $m \geq 0$  such that  $A^m = B$ , if such an  $m$  exists.

7. **Isomorphism of finite abelian groups.** The structure theorem for finite commutative group (also known as the fundamental theorem of finite abelian groups) states that every finite commutative group  $G$  is isomorphic to a group of the form

$$\mathbb{Z}_{p_1^{e_1}} \oplus \mathbb{Z}_{p_2^{e_2}} \oplus \dots \oplus \mathbb{Z}_{p_k^{e_k}},$$

where  $\mathbb{Z}_m$  represents the cyclic group of order  $m$ . Use the fundamental theorem of finite abelian groups to devise an efficient (polynomial time) algorithm that given two sets of integers  $\{d_1, d_2, \dots, d_m\}$  and  $\{e_1, \dots, e_n\}$  determines whether the group

$$G_1 \stackrel{\text{def}}{=} \mathbb{Z}_{d_1} \oplus \dots \oplus \mathbb{Z}_{d_m}$$

is isomorphic to the group

$$G_2 \stackrel{\text{def}}{=} \mathbb{Z}_{e_1} \oplus \dots \oplus \mathbb{Z}_{e_n}.$$

8. **Search versus decision for determinant identity testing.** Let  $\mathbb{F}$  be a field and  $\mathbf{x} \stackrel{\text{def}}{=} (x_1, \dots, x_m)$  be an  $m$ -tuple of indeterminates. Recall the determinant identity testing problem: given an  $n \times n$  matrix  $M(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  each entry of which is either a constant from the field  $\mathbb{F}$  or is an indeterminate from the set  $\{x_1, x_2, \dots, x_m\}$ , the task is to compute whether the determinant of  $M(\mathbf{x})$ , denoted  $\text{DET}(M(\mathbf{x}))$ , is the identically zero polynomial or not. The search version of this problem is to compute an assignment  $\mathbf{a} \in \mathbb{F}^m$  to the indeterminates so that  $\text{DET}(M(\mathbf{a}))$  is a nonzero element of  $\mathbb{F}$ . *Show that the search version of the determinant identity testing problem is deterministic polynomial time equivalent to the decision version of the problem.* In other words, show that given an oracle for determining the existence of an assignment to the variables which makes the determinant of a given matrix nonzero, in deterministic polynomial time we can in fact compute such an assignment which makes the determinant nonzero.
9. **Bivariate polynomials over  $\mathbb{Q}$  which have a root for every  $y$ .**  $\mathbb{Q}$  as usual denotes the field of rational numbers. Design a polynomial-time algorithm that given a bivariate polynomial  $f(x, y) \in \mathbb{Q}[x, y]$  which is monic with respect to  $x$ , determines if it has the property that for all  $y \in \mathbb{Q}$ , there exists an  $x \in \mathbb{Q}$  such that

$$f(x, y) = 0.$$

10. **An algebraic analogue of the Diffie-Hellman key exchange protocol.**

*Background.* In 1976, Whitfield Diffie and Martin Hellman gave birth to a new era of computational complexity-based Cryptography. They devised a cryptographic protocol that allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel. This key can then be used to encrypt subsequent communications using a symmetric key cipher. It is known as the Diffie-Hellman key exchange protocol and it has not been broken yet. It is based on the presumed difficulty of the discrete-logarithm problem.

Dr. K has recently discovered an algebraic analogue of the Diffie-Hellman key exchange protocol. Dr. K's protocol is based on the presumed difficulty of a different problem: *the polynomial equivalence problem*. This problem is the following:

Given a pair of  $n$ -variate polynomials  $F(\mathbf{x})$  and  $G(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  compute a matrix  $A \in \mathbb{F}^{n \times n}$  such that  $F(A \cdot \mathbf{x}) = G(\mathbf{x})$ , if such a matrix  $A$  exists.

No efficient algorithm is known for the polynomial equivalence problem. The protocol devised by Dr. K works as follows.

- i). Alice and Bob agree to use a randomly chosen prime  $p$ , a matrix  $A \in \mathbb{F}_p^{n \times n}$  and a polynomial  $F(\mathbf{x}) \in \mathbb{F}_p[\mathbf{x}]$ , where  $\mathbf{x} = (x_1, \dots, x_n)$  is an  $n$ -tuple of indeterminates.
- ii). Alice chooses a secret polynomial  $g(x) \in \mathbb{F}_p[x]$  and computes the matrix  $B \stackrel{\text{def}}{=} g(A) \in \mathbb{F}_p^{n \times n}$ . Alice sends Bob the polynomial  $G(\mathbf{x}) \stackrel{\text{def}}{=} F(B \cdot \mathbf{x})$ .
- iii). Bob chooses a secret polynomial  $h(x) \in \mathbb{F}_p[x]$  and computes the matrix  $C \stackrel{\text{def}}{=} h(A) \in \mathbb{F}_p^{n \times n}$ . Bob sends Alice the polynomial  $H(\mathbf{x}) \stackrel{\text{def}}{=} F(C \cdot \mathbf{x})$ .
- iv). Alice computes  $K_a(\mathbf{x}) \stackrel{\text{def}}{=} H(B \cdot \mathbf{x})$ .
- v). Bob computes  $K_b(\mathbf{x}) \stackrel{\text{def}}{=} G(C \cdot \mathbf{x})$ .

Notice that  $K_a(\mathbf{x}) = F(BC \cdot \mathbf{x})$  and  $K_b(\mathbf{x}) = F(CB \cdot \mathbf{x})$ . The two exercises below show that  $K_a(\mathbf{x})$  and  $K_b(\mathbf{x})$  are in fact equal and this is the common key that is exchanged in the protocol.

- Show that in the protocol above, the matrices  $B$  and  $C$  commute, i.e.  $B \cdot C = C \cdot B$ .
- Deduce that  $K_a(\mathbf{x}) = K_b(\mathbf{x})$ . This common polynomial is the common key.

We now turn our attention to the security of the cryptosystem. Note that only the matrices  $B$ ,  $C$  and the key  $F(BC \cdot \mathbf{x}) = F(CB \cdot \mathbf{x})$  are kept secret. All the other values are sent in the clear.

- Show that if the eavesdropper Eve can solve the polynomial equivalence problem efficiently then she can also recover the key.
- Show that the polynomial equivalence problem can be solved efficiently in the special case that the transformation matrix is a diagonal matrix.
- Use the above result to show that the cryptosystem designed by Dr. K can be efficiently broken. That is, Eve can recover the key polynomial  $F(BC \cdot \mathbf{x}) = F(CB \cdot \mathbf{x})$  from the information sent in the clear in randomized polynomial time.