Computational Complexity Theory — Avi Wigderson

Problem Set 4 — Due: May 16, 1999

1. *A Paradox.*
   It is known that if $3SAT \in P$ then $PH = P$. In particular we have a polynomial time algorithm for $MIN - CIRCUIT$ that was defined in class. On the other hand it is believed in the CS community that there is no polynomial time algorithm for $MIN - CIRCUIT$ even if we are allowed to use $3SAT$ as an Oracle. Write your opinion about this seemingly paradox.

2. *Chernoff Bound.*
   The aim of this exercise is to show that if we have many independent experiments the outcome is exponentialy likely to be extremely close to the expectation. This is one of the most usefull facts in probability and in finite combinatorics.

   Let $X_1, ..., X_n$ be independent Bernoulli trials such that, for $1 \leq i \leq n$, $\Pr[X_i = 1] = p_i$, where $0 < p_i < 1$. Define $X = \Sigma_{i=1}^n X_i$, $\mu = \mathbf{E}[X] = \Sigma_{i=1}^n p_i$.

   (a) Prove that for and any $\delta > 0$, $\Pr[X > (1+\delta)\mu] \leq \frac{e^{\delta\mu}}{(1+\delta)^{\mu(1+\delta)}}$.

   (b) Prove that for $0 < \delta \leq 1$, $\Pr[X > (1-\delta)\mu] \leq e^{-\mu\delta^2/2}$.

   Hints:

   1. Instead of looking at the event $X > (1+\delta)\mu$ consider the equivalent event $e^{tX} > e^{t(1+\delta)\mu}$ for some fixed t. Now use Markov inequality and then maximize over t.
   2. Recall that $\forall z \; 1 + z \leq e^z$.
   3. For part (b) use the fact (which you can easily verify by taking derivatives) that for $0 < \delta \leq 1$, $(1-\delta)^{1-\delta} > e^{-\delta+\delta^2/2}$.

3. *BPP.*

   Definition 1 *The class $BPP(\epsilon)$ (for Bounded-error Probabilistic Polynomial time) consists of all languages L that have a randomized algorithm A running in worst-case polynomial time such that for any input $x \in \{0,1\}^*$,*

   - $x \in L \Rightarrow \Pr[A(x)\ accepts] \geq 1 - \epsilon$.
   - $x \notin L \Rightarrow \Pr[A(x)\ accepts] \leq \epsilon$.

   BPP $= BPP(1/3)$.

   (Note: The probabilistic aspect of A does not affect the running time that is always polynomial. However wrong answers can occur by bad luck.)
   In this question n is the input length. The following equation says that it is possible to transform a BPP algorithm with an error as large as $\frac{1}{2} - 1/poly(n)$ to one with error probability as small as $2^{-poly(n)}$:

   Prove that $\forall c$, $BPP(\frac{1}{2} - n^{-c}) = BPP(2^{-n^c})$.

4. Oracles.

   Definition 2 *Let $P^{NP[\log n]}$ be the class of all languages that can be decided by a polynomial time Oracle TM which on input of length n asks a total of $O(\log n)$ SAT queries. Let $P_{||}^{NP}$ (for an Oracle machine that asks its queries in parallel) be the class of all languages that can be decided by an Oracle TM operating as follows: On input x, the machine first computes in polynomial time a polynomial number of instances of SAT, and receives the correct answers for them (from the Oracle). Based on these answers, the machine decides whether $x \in L$ in polynomial time.*

   Prove that $P^{NP[\log n]} = P_{||}^{NP}$.

5. The Permanent.

   Definition 3 *For a matrix $A \in M_{n\times n} = (a_{i,j})$ let*

$$PERM(A) = \sum_{\sigma\in S_n}\prod_{i=1}^{n} A_{i,\sigma(i)}.$$

   where S_n is the group of all permutations of n elements (this is like the Determinant with no ± signs).

   a. Show that the Permanent is the coefficient of $y_1 y_2 \cdots y_n$ in the polynomial

$$p(y_1, y_2, ..., y_n) = \prod_{i=1}^{n}\sum_{j=1}^{n} y_j a_{i,j}.$$

   b. Prove that

$$PERM(A) = (-1)^n \sum_{T\subset[n]} (-1)^{|T|} \prod_{i=1}^{n}\sum_{j\in T} a_{i,j}$$

   in two different ways:

   i. Verify the formula directly.

   ii. Use interpolation to get the coefficient of $y_1 y_2 \cdots y_n$ in the polynomial from (1.).

   c. Let $X_{1,1}, ..., X_{n,n}$ be independent random variables such that, for $1 \le i \le n$, $\mathbf{E}(X_{i,j}) = 0$, $\mathbf{VAR}(X_{i,j}) = 1$. Denote by $B \in M_{n\times n}$ the following random matrix : $B = (b_{i,j})$, $b_{i,j} = \sqrt{a_{i,j}} X_{i,j}$.
   Show that $PERM(A) = \mathbf{E}(DET(B)^2)$ (we thing about B as a random matrix, and take the expected value of $DET(B)^2$).

   Since the Determinant is easy we managed to find a representation of the Permanent as the expectation of an "easy" random variable. The natural thing to do would be to approximate it.

   d. Let $\Pr[X_{i,j} = 1] = \Pr[X_{i,j} = -1] = 1/2$. How many estimation of $(DET(B)^2)$ should we do in order to approximate $PERM$ up to a constant, λ, according to Chevichev's inequality?