6.841 Advanced Complexity Theory April 11, 2007

Lecture 17

Lecturer: Madhu Sudan *Scribe: Catherine Lennon*

1 Today

- We will prove $PSPACE \subset IP$.
- Ramblings on knowledge.

2 Review of Polynomial Construction Sequence

As in last class, a polynomial construction sequence has the parameters $w, l, m, d, t \in \mathbb{Z}$ and a prime p, which signify the following: t represents time, and $p_1, ..., p_l$, is a sequence of m-variable polynomials of degree d. We work over the field $\mathbb{F}_p$ of p elements and when computing p_i, we are allowed at most w oracle calls to p_{i-1}.

We have some $\bar{a} \in \mathbb{F}^m$, and $b \in \mathbb{F}$. The goal is to show that $p_l(\bar{a}) = b$. Can this be proven with a verifier in probabilistic time $poly(m, d, l, w, t, \log|\mathbb{F}|)$? The technique will be to compute this in stages, from l down to 1. At the ith phase, the following interaction takes place:

- Known: $a^{(i)}$, $b^{(i)}$, $p_i(a^{(i)}) = b^{(i)}$.
- Verifier: calculates and sends vectors $v_1, .., v_w \in \mathbb{F}^m$.
- Prover: sends $c : \mathbb{F} \to \mathbb{F}^m$ of degree w such that $C(i) = v_i$, $i \in \{1, ..., w\}$, and h such that $p_{i-1}|_C = h(t)$.
- Verifier: checks that $C(i) = v_i$, $b^{(i)} = f_i(h(1), ..., h(w))$. He then chooses $t_0 \in \mathbb{F}$ uniformly at random and sends t_0.
- Prover: computes $p_{i-1}(a^{(i-1)}) = b^{(i-1)}$, where $a^{(i-1)} = c(t_0)$, and $b^{(i-1)} = h(t_0)$.

3 Proof

Consider the pair (M, x), of Turing machine M with input x. Let s denote the number of bits needed to describe the configuration of (M, x). Define

$$F_i(\sigma, \tau) = \begin{cases} 1 & \text{if } \sigma, \tau \in \{0,1\}^s \text{ and } \sigma \Rightarrow \tau \text{ in } 2^i \text{ steps;} \\ 0 & \text{if } \sigma, \tau \in \{0,1\}^s \text{ but } \sigma \text{ does NOT} \Rightarrow \tau \text{ in } 2^i \text{ steps;} \\ \text{arbitrary} & \text{otherwise.} \end{cases}$$

Then $F_i : \{0,1\}\mathbb{F}^{2s} \to \mathbb{F}$ since it is defined on all elements of $\mathbb{F}^{2s}$, but it only gives meaningful output on inputs in $\{0,1\}^{2s}$.
Claim: (not proven) $F_0(\sigma, \tau)$ can be defined so as to be a polynomial in $2s$ variables of degree $C = O(1)$ in each variable.

Then $F_i(x,y) = \sum_{z \in \{0,1\}^s} F_{i-1}(x,z)F_{i-1}(z,y)$. Notice that there will be only one nonzero term in the sum, corresponding to the midpoint. Also, this is a polynomial of degree C in each variable if F_{i-1} is. But there is a catch: in order to compute F_i, we need to determine the value of F_{i-1} at exponentially many places, giving us an exponentially long sum. We will use an observation to simplify this.

Observe that the given definition of F_i is equal to the following:

$$F_i(x,y) = \sum_{z_1 \in \{0,1\}} \sum_{z_2 \in \{0,1\}} ... \sum_{z_s} F_{i-1}(x,z)F_{i-1}(z,y)$$

We may only wish to consider the sum of a smaller number of the nested summations, and this leads us to define for convenience the following functions: let $G_{i,j}(x,y,z_1,...,z_j) := \sum_{z_{j+1}} ... \sum_{z_s} F_{i-1}(x,z)F_{i-1}(y,z)$. Then in particular, $G_{i,j}(x,y,z_1,..,z_j) = G_{i,j+1}(x,y,z_1,...,z_j,0) + G_{i,j+1}(x,y,z_1,...,z_j,1)$, and $G_{i,s}(x,y,z_1,..,z_s) = F_{i-1}(x,z)F_{i-1}(z,y)$. Thus the sequence begins with $F_0 = G_{0,0}$ and using this we may define $G_{1,s}, G_{1,s-1}, ..., G_{1,0} = F_1$, etc until finally we have computed $G_{s,0} = F_s$ and we are done.

An interesting detail is that the inputs we were given was (M,x), but where were these used in the proof? The machine M affects F_0 and the input x affects F_s, but all of the intermediate computation steps are oblivious to the problem involved.

4 Comments about IP

We often wish to show that some problems are not very hard. For example, say we have a language L, and we wish to determine if L is NP-complete. If we can show that its complement $\bar{L} \in AM$ then under the infinite hierarchy assumption, L cannot be NP-complete. Although IP, AM do not seem to be such different complexity classes, the equivalent statement using IP instead of AM is not true. Thus IP and AM are far more different than they may appear at first glance.

Another use for IP is studying knowledge and secrecy. For more about this, take 6.875, 6.876. We will not go into detail on these topics in this course, but we will briefly discuss the concept of knowledge.

4.1 Shannon's Theory of Information

One question that was asked is how can someone measure information. To Shannon, tossing an unbiased coin n times independently generates n bits of "information". However, this does not really capture the "real life" meaning of the

term "information". To remedy this, we introduce the concept of "knowledge"- in this context, the n random coin tosses do not contain any bits of knowledge, so this comes closer to the usual notion.

Example: If I pick random n-bit primes P, Q, and send $N = PQ$ to you, we claim that you do not know P or Q. But how do we define "knowing"?

4.2 Graph Isomorphism Protocol

This concept was formalized by Goldwasser, Micali, and Rackoff by developing what are called *zero-knowledge proofs*. Before we define this formally, consider the following protocol for proving that two graphs G_1, G_2 are isomorphic. Graphs G_1, G_2 are known to the Prover and Verifier. The prover knows π_0 where $\pi_0(G_1) = G_2$ is an isomorphism. She wishes to prove that $G_1 \approx G_2$ to the Verifier without revealing anything about the isomorphism. The protocol works as follows:

- Prover: picks $i \in \{1, 2\}$, $\pi \in S_n$ and sends the graph $H = \pi(G_i)$.
- Verifier: chooses a challenge bit $b \in \{0, 1\}$ and sends it.
- Prover: if $b = i$, she sends π, if $b = 1$, $i = 2$, she sends $\pi \circ \pi_0$, and if $b = 2$, $i = 1$, she sends $\pi \circ \pi^{-1}$.
- Verifer: checks that this is in fact an isomorphism from G_b to H.

Intuitively, this protocol has the following properties:
Completeness: if $G_1 \approx G_2$, then $V \leftrightarrow P$ always accepts.
Soundness: if G_1, G_2 are nonisomorphic, then it rejects with high probability.
Zero-Knowledge: of $G_1 \approx G_2$, the verifier learns nothing other than this fact.

4.3 Zero-Knowledge Proofs

In order to formalize these concepts, [GMR] introduced the concepts of simulators and transcripts. A transcript is a sequence of strings, which is a random variable even if we fix the verifier's random coins R, and it has a distribution D_R.

Claim: we can sample strings from this distribution.

In the case of the above example, we sampled from the distributions exactly, and our transcript was H, b, π. Such a protocol is a *Perfect Zero Knowledge* protocol. However, we may not be able to sample exactly, and so we define weaker concepts of zero-knowledge protocols. If the simulator produces a $D' \neq D$ define the "statistical distance" of these two distributions to be

$$||D' - D|| := \frac{1}{2} \sum_x |D'(x) - D(x)|$$

Equivalently, if we fix any $T : \{0,1\}^n \to \{0,1\}$, then $\max |Pr_{x \in D}[T(x) = 1] - Pr_{x \in D'}[T(x) = 1]| = ||D' - D||$. A *Statistical Zero-Knowledge* proof is one where $||D_{\text{prover}} - D_{\text{simulator}}|| \leq \frac{1}{n^c} \forall c$.

Finally, we also have the concept of *Computational Zero-Knowledge* proofs, where we define a new distance

$$||D' - D||_{\text{comp}} = \max_{T \text{ a poly-sized circuit}} |Pr[T(D) = 1] - Pr[T(D') = 1]|$$

.

4.4 Results

Following the formulation of PZK, SZK, CZK, the following results were proven

- Goldreich, Micali, Wigderson showed that graph isomorphism is in $PZK \subset SZK \subset CZK$.
- if $L \in SZK$ and L is NP-complete, then the hierarchy collapses.
- Okamoto showed that $SZK = coSZK$
- Vadhu showed that if $L_1, L_2 \in CZK$, then $L_1 \cup L_2 \in CZK$