6.841 Advanced Complexity Theory Apr 2, 2007

Lecture 14

Lecturer: Madhu Sudan *Scribe: Nate Ince, Krzysztof Onak*

1 Introduction

The idea behind interactive proofs came from cryptography. Transferring coded massages usually involves a password known only to the sender and receiver, and a message. When you send an encrypted message, you have to show you know the password to show that your message is official. However, if we do this by sending the password with the message, there's a high chance the password could be eavesdropped. So we would like a method to let you show you know the password without sending it. Goldwasser, Mikoli, and Rakoff were inspired by this to create the idea of the interactive proof.

2 The IP protocol

Let L be some language. We say that $L \in IP$ as a protocol between an exponential-time prover P and a polynomial time randomized verifier V so if $x \in L$, then the probabillity that after a completed exchange Π between P and V, $Pr[V(x, \Pi) = 1] \geq 1 - 1/2^{|x|}$ and if $x \notin L$, then the probability that after a completed exchange Π' between Q and V, where Q is any exponential algorithm, $Pr[V(x, \Pi') = 1] \leq 1/2^{|x|}$. We formally define an exchange between P and V as a series of questions and answers $q_1, a_1, a_2, q_2 \ldots, q_n, a_n$ so that $|q_i|, |a_i|, n \in poly(|x|)$, $q_1 = V(x)$, $a_i = P(x; q_1, a_1, \ldots, q_i)$, and $q_i = V(x, ; q_1, a_1, \ldots, a_{i-1}; r_{i-1})$ for all $i \leq n$, where r_i is a random string V uses at the ith exchange to determine all "random" decisions V makes. Note that only V knows r_i.

It is clear that NP and BPP are contained in IP. NP is just IP without random decisions. We will prove that, in fact, $IP = PSPACE$

3 Graph Non-Isomorphism

We have previous seen the graph isomorphism problem (GI) to be quite difficult. However, we will now show $GNI \in IP$. We briefly define:

Definition 1 *Two graphs G_1 and G_2 with vertices numbered from 1 to n are equivalent ($G_1 \simeq G_1$) if there is a permutation Π so that if ij is an edge in G_1 then $\Pi(i)\Pi(j)$ is an edge in G_2*

If two graphs are isomorphic, the permutation is all that is necessary for verifiction. More is necessary

We now demonstrate $GI \in IP$ by the following protocol:

1. V picks some permutation Π at random and chooses $i \in \{1, 2\}$. V sends $H = \Pi(G_i)$.

2. If $G_1 \ncong G_2$, P can search in exponential time for which graph H was before permutation (Or guess). Otherwise, it has a 1/2 chance of guessing. V sends its answer, j, to P.

3. P accepts if $i = j$, otherwise it goes back to step one until there have been a suitable number of rounds. Probabillity of an error goes down in the number of rounds.

4 The Arthur-Merlin Protocols

In a similar manner Bobai and Moran defined a type of proof known as an Arthur-Merlin protocol. This protocol is similar to IP. Arthur functions like the Verifier V, except that he must announce r_i to Merlin. Merlin is computationally the same as P (Including the possibillity of dishonesty for $x \notin L$. AM[k] is the class after an exchange Π between Arthur and Merlin in k queries by Arthur and responses by Merlin, then if $x \in L$ then Arthur accepts with probabillity at least 2/3, and if $x \notin L$, then Arthur accepts with probabillity at most 1/3. Similarly, MA consists of a one question by Merlin and an answer from Arthur, and qualification for acceptance remains the same.

5 Known inclusions between the classes

5.1 Brief overview

- NP $\subseteq$ MA

  This inclusion holds because Merlin can simply send Arthur a proof that x belongs to the language. Is it possible that NP = MA? One can show that BPP $\subseteq$ MA, so this would imply that BPP $\subseteq$ NP, a result that we do not know how to prove.

- MA $\subseteq$ AM

  We can let Arthur act first.

- AM = AMAMAM

  Any constant number of communication rounds between Arthur and Merlin can be replaced by a single round.

- AM $\subseteq$ IP

  We can trivially replace a constant number of communication rounds by a polynomial number of them, and public randomness by private randomness.

- IP = PSPACE, AM $\subseteq \Sigma_2^{\mathrm{p}}$

  [Lund-Fortnow-Karloff-Nisan], [Shamir]

5.2 The containment of IP in PSPACE

It turns out that the actions of the optimal prover can be computed in PSPACE. Given a verifier V we wish to find out what is the maximum probability that V accepts a word x. More formally, we are interested in the value of

$$\max_P \Pr[P \leftrightarrow V(x)],$$

which, as we will shortly see, can be computed in polynomial space.

We consider the tree of all possible interactions between the prover and the verifier. The internal nodes represent the turns of either the verifier or the prover, and the edges outgoing from nodes to their children correspond to all possible messages sent to the other party. In each leaf of the tree, we consider all the sequences of coin tosses of the verifier that are consistent with the messages on the path from the root to this leaf. We compute the conditional probability that x was accepted given this sequence of messages.

We can compute the desired quantity recursively by DFS. At each node representing the prover's round, we choose the message that maximizes the probability of x being accepted (this is how the optimal prover would respond), that is, we take the maximum of the probabilities at the children.

At each node representing a turn of the verifier, we compute a weighted average of the probabilities at children. The average is weighted by the probabilities of choosing each message by the verifier.

The desired quantity can be computed recursively without constructing the interaction tree explicitly, and by the definition of IP, it suffices to know it to be able to tell if x is in the language or not.

5.3 One-sided error public = two-sided error private

Now we will show that every protocol with private coin tosses and two-sided error can be replaced by a protocol with public coin tosses and one-sided error, for interactive proof systems with polynomially many rounds. The proof that we present is due to Kilian.

Let V be a prover with two-sided error and private coin tosses. We are interested in the number of accepting paths of V on x with an optimal prover. If we knew it, we would be able to tell if x is in the language or not.

We will construct a verifier V', and we will show how to bound the correct number of accepting paths for x. Consider the tree of interaction of V with an

optimal prover. Each edge corresponds to a message sent by the verifier and a reply sent by the optimal prover. We assume that V' can ask what is the optimal answer to a query of V, and moreover, for each sequence of messages exchanged by V and the optimal prover, we can ask what is the number of the accepting computations in the subtree corresponding to this sequence of messages.

For simplicity suppose that V always sends a single bit. This implies that each internal node in the interaction tree has two children. V' will descend down the interaction tree, asking at each node about the number N of accepting paths in the subtree rooted at that node, and about the same numbers, N_0 and N_1, for the children of that node. Verifier V' always makes sure that $N = N_0 + N_1$, and with probability N_i/N follows the path corresponding to the situation when V answers i. If eventually it reaches a leaf corresponding to an accepting situation it accepts, and otherwise it rejects.

Let N be the actual number of accepting paths, and let N' be the number of accepting paths according to a prover. If the prover tries to overestimate the number of accepting paths ($N' > N$), we can detect this with probability at least $1 - N/N'$. This can easily be proven by induction. Suppose that in the subtrees of the node the actual numbers of accepting paths are N_0 and N_1, and the claimed numbers of accepting paths are N'_0 and N'_1. We assume that $N' = N'_0 + N'_1$, otherwise we detect that the prover is cheating instantly. The probability of detection of cheating is

$$\left(1 - \frac{N_0}{N'_0}\right) \cdot \frac{N'_0}{N'} + \left(1 - \frac{N_1}{N'_1}\right) \cdot \frac{N'_1}{N'} = 1 - \frac{N}{N'}.$$

Therefore, in our new proof system a prover cannot significantly overestimate the number of accepting paths.

6 Next lecture

It turns out that one can convert a two-sided error private-randomness verifier into a one-sided error public-randomness verifier preserving the number of rounds of interaction ([Goldwasser-Sipser], [Goldreich-Mansour-Sipser]). As a component of this result we will show the Goldwasser-Sipser protocol for Approximate Counting.