6.842 Randomness and Computation September 25, 2017

Lecture 6

Lecturer: Ronitt Rubinfeld *Scribe: Leo de Castro*

1 Interactive Proof Systems

An interactive proof system is a protocol that defines an interaction between two machines, a prover and a verifier. For this model, we will consider verifiers that run in polynomial-time. The verifier will have access to random coins, which for now we will consider private. The prover, on the other hand, is unbounded in time and space. Without loss of generality, we can consider the prover to be a deterministic machine.

The prover and the verifier have a shared input x, and the prover attempts to convince the verifier that $x \in L$ for some language L. After the interaction with the prover, the verifier will output *Accept* if it believes that $x \in L$ or *Reject* if it outputs $x \notin L$. From this, we can define an interactive proof system and the corresponding complexity class of languages with these proof systems.

Definition 1 *Interactive Proof Systems (IPS) [Goldwasser, Micali, Rackoff]*

A language L has an Interactive Proof System if there is a protocol for an interaction between a polynomial-time verifier V and an unbounded prover P such that if V and P both follow the protocol and $x \in L$, the probability that V outputs *Accept* is at least $\frac{2}{3}$.

$$Pr[V(x) = Accept \mid x \in L] \geq 2/3$$

Note that this probability is taken over the random coins of the verifier. These proof systems must also meet a soundness condition such that if $x \notin L$ and the verifier V follows the protocol, regardless of what the unbounded prover P does, the probability that the verifier rejects is at least $\frac{2}{3}$.

$$Pr[V(x) = Reject \mid x \notin L] \geq 2/3$$

This probability is also taken over the random coins of the verifier.

Definition 2 *The complexity class IP*

A language L is in the class IP if it has an Interactive Proof System.

It is clear that the class $NP \subset IP$, since the prover P can always just send the short witness that x is in the language, which can be checked in polynomial time by the verifier.

It is also the case that $IP = PSPACE$, which is much less obvious and will not be proved here. It is also a fact that $PSPACE$ is closed under compliment. In order to illustrate the power of IPS, we will construct an interactive proof for graph non-isomorphism.

2 Graph Non-isomorphism

Graph isomorphism is in NP, so it is also in IP. Since $IP = PSPACE$, which is closed under compliment, we know that there exists a IPS for graph non-isomorphism. We will first consider an IPS were the verifier's random coins are private.

The input to this protocol is two graphs G and H, both with n nodes. The verifier is supposed to output *Accept* if $G \ncong H$ and output *Reject* if $G \cong H$. The protocol proceeds as follows, which is looped a constant number of times:

1. V uses its private random coins to compute a graph G' that is isomorphic to G and a graph H' that is isomorphic to H.
2. V then flips a coin to decide whether to send (G, G') or (G, H') to the prover.
3. P returns a bit indicating the result of coin flipped by V in step 2.

To better illustrate this protocol, here is a table of possible response and action combinations.

Coin Flip Result	Correct Response if $G \ncong H$	P response	V output
H	$\cong$	$\cong$	Continue
H	$\cong$	$\ncong$	Reject and Stop
T	$\ncong$	$\cong$	Continue
T	$\ncong$	$\ncong$	Reject and Stop

The only way that the prover can (consistently) determine the result of the coin flip is if the prover can distinguish between G' and H'. If G is not isomorphic to H, then H' is not isomorphic to G, so the prover will always by able to determine whether it was sent (G, G') or (G, H') by simply testing if the second graph is isomorphic to G. Therefore, if the two graphs are not isomorphic and the prover and the verifier both follow the protocol, the prover can always determine the result of the coin flip.

However, if G and H are isomorphic, the G' and H' are statistically indistinguishable, since they are equivalent up to isomorphism. Therefore, the prover cannot distinguish between (G, G') and (G, H') better than random guessing. More formally, let q be the fraction of random permutations π such that the prover outputs that $(G, \pi(G))$ are not isomorphic. For a given round of the protocol, we have the probability that the prover fails to pass the challenge is:

$$Pr[Prover\ fails\ the\ round] = \frac{1}{2}q + \frac{1}{2}(1-q) = \frac{1}{2}$$

This probability is over the graphs produced by the verifier, since both G' and H' are permutations of G. Since the prover cannot do better than random guessing when $G \cong H$, repeating the loop above twice will result in the following probabilities:

$$Pr[V(G, H) = Accept \mid G \ncong H] \geq 3/4$$

$$Pr[V(G, H) = Reject \mid G \cong H] \geq 3/4$$

This is sufficient to satisfy the IPS requirements, so this completes the construction of our private-coin IPS for graph non-isomorphism.

3 From Private Coins to Public Coins

In our IPS protocol above, it is crucial that the verifier's private coins are not revealed to the prover, otherwise the prover could always cheat and know how G' and H' were generated. However, in an amazing result by Goldwasser and Sipser, public coin IPS are just as powerful as private coin IPS

Fact 3 *Public Coin IP = Private Coin IP*

Here, we will begin to construct a public coin IPS for graph non-isomorphism. We will not finish here.

Consider the set $[A] = \{A' | A \cong A'\}$, which is the set of all graphs that are isomorphic to A. For any graph A with n nodes, the size of $[A]$ is $n!$. For two graphs A and B, the union $U = [A] \cup [B]$ will have a size of $n!$ if $A \cong B$ and $2n!$ if $A \ncong B$. What we need is an ISP that gives some indication of the size of U.

In order to do this, we should first consider the universe of all graphs with n nodes Z_n. The size of Z_n is 2^{n^2}, which is the number of ways that n nodes can be connected. It is not sufficient to simply test random graphs to get a sample of the size of U, since the fraction $\frac{n!}{2^{n^2}}$ is far too small for a polynomial time verifier to sample random graphs.

Instead, the verifier can pick a pairwise independent hash function that maps a space of size 2^{n^2} to a space of size 2^l. The criteria that should be met by this hash function is:

1. $|h(U)|$ is big if and only if $|U|$ is big.
2. $\frac{|h(U)|}{2^l}$ is $\frac{1}{poly(n)}$.
3. h is computable in polynomial time.

The public coin protocol will proceed as follows:

1. Given H, a collection of pairwise independent hash functions mapping $\{0,1\}^{n^2} \rightarrow \{0,1\}^l$, V randomly selects a function h from this family and sends h to P.
2. P sends to V a graph $x \in U$ such that $h(x) = 0^l$ along with a proof that $x \in U$.

We will show next time that if the graphs A and B are not isomorphic, then with high probability the image of the union $[A] \cup [B] = U$ in the hash function will contain 0^l. If A is isomorphic to B, then this occurs with low probability.