18.705: Commutative Algebra February 5, 2003

Lecture 1: Interactive Proofs and Zero-Knowledge Proofs

1 Preface

What is this course about?

Cryptography is about lots of things. Linguistically it specifically refers to "secret writing." Lots of topics fit into the heading of "cryptography," but in this course we will not consider all of them.

Here, we talk about, generally, the union of two concepts: **correctness** and **privacy.** These are opposed notions, because to establish correctness, one has to disclose, which is the opposite of privacy. This conflict is at the heart of cryptography, and is critical to many cryptographic problems. For instance, in the encryption problem, we wish for the two parties to actually communicate messages but for privacy to be maintained. In signature schemes, we wish to allow users to demonstrate their identity and yet keep their ability to do so private. In particular, this course will begin its focus on zero-knowledge proofs. Roughly, a zero knowledge proof protocol is a way for one entity to convince another of a theorem and yet leak no other information.

2 Traditional Proofs

Traditional proofs are based on derivations involving *rules* and *axioms.* For example, if A and $A \to B$ are axioms, a proof of B might consist of a list of statements such as

1. A (axiom)
2. $A \to B$ (axiom)
3. B (rule of inference, using lines 1 and 2.)

There is no room for privacy here. That is, in order to be convincing, one must actually give the proof, there's really no other way to model it. However, this still can form the basis of talking about proofs more generally. Let us suppose that a prover (P) wishes to convince a verifier (V) of the truth of a theorem x. The traditional way of doing this is for the prover to simply write the proof of x down and show it to the verifier.

Note that if these parties are to be realistic entities, the proof, and the process of checking it, must be short (i.e., polynomial). However, we are still supposing that *finding* proofs is hard, and so we do not ask that the prover find their proof in polynomial time.

The class of theorems we can prove this way is the class of statements of the form $x \in L$ for some NP language L. A language L is in NP if and only if there is some polynomial time algorithm (or Turing machine) A such that for any x in L, there is some "proof" (or "witness") y such that A accepts the input (x, y), but if x is not in L then A will never accept (x, y') for any y'.

3 Interactive Proofs

Setting up the prover and verifier and this process of proof leads us to ask the natural question: can we prove more things if some kind of meaningful interaction takes place between the prover and verifier?

We describe interactive proofs by giving an example first. We do however note that the idea is that a proving entity (the "Prover," usually denoted P) and a verifier (usually denoted V.) For now we assume the prover is unbounded computationally, while the verifier is probabilistic polynomial time (denoted PPT.) We allow the verifier access to randomness for two reasons: first of all, it may be that randomized polynomial-time algorithms can solve more problems than deterministic ones, and second, our example will require randomness. (Indeed, once the example is given, randomness will seem to be *fundamental* to this kind of proof.)

We wish to prove membership in a language (a subset of all strings, written $L \subset \{0,1\}^*$. In particular, we consider the language $ISO = \{(G,H)|G \cong H\}$, the language of pairs of graphs which are isomorphic to each other. We also consider $NISO = \{(G,H)|G \ncong H\}$. Note that it is easy to conceive a (traditional) proof that a given (G,H) pair is in ISO; we just state the permutation π, which can be checked. However, it is not so easy to show that a given pair is in $NISO$. A conceivable proof would be to give, for every permutation, a pair of vertices which either should be connected and are not under that permutation, or should not be connected and are. However, clearly, this proof is more than polynomial length. This corresponds to the notion that $ISO \in$ NP; this is actually how NP was defined originally, while $NISO \in$ coNP.

But, we can establish the membership of a pair (G_0, G_1) is NISO *interactively*, as follows.

1. V picks a bit b at random.
2. V picks a random permutation π in $S_{|G|}$ and computes $C = \pi(G_b)$.
3. V sends C to P.
4. P finds $G_{b'}$ such that $G_{b'}$ is isomorphic to C, and sends b' to V.
5. V checks that b and b' are the same, and rejects if not.
6. (optional) They play this game k times. If V gets to the end without rejecting, V accepts.

Loosely speaking, the prover can only succeed in this challenge if the graphs are not isomorphic; if they are isomorphic, then C is isomorphic to both G_0 and G_1 so there is inherently a good chance the prover could be wrong with whatever guess b' the prover makes.

We wish to say that (P,V) is an interactive proof system (IPS) for $NISO$. We need a definition.

We say (P,V) is an IPS for a language L if

1. **Completeness:** "Any true theorem can be proven."

2. **Soundness:** "No false theorem can be proven."

Now, to make this definition reasonable, we need to find mathematical statements which properly correspond to these meanings.

1. **Completeness:** $\forall x \in L,\ Pr[(P,V)[x] = 1] = 1$. More generally, we could allow probability $1 - \nu(k)$ where ν is negligible, meaning, it vanishes faster than any polynomial, where $k = |x|$. However, it turns out we don't need to be so general.

2. **Soundness:** Our first attempt would probably be $\forall x \notin L,\ Pr[(P,V)[x] = 1] \leq 1/2$, but this would be insufficient: this only guarantees that the honest P will not convince the honest V. But what about a prover that doesn't follow the rules? For one thing, the honest prover may say something meaning "the theorem is false!" if the theorem is false. Thus, the right definition is

$$\forall x \notin L \forall P'\ Pr[(P',V)[x] = 1] \leq 1/2.$$

Note that we ask for probability $\leq 1/2$ but this leaves open the possibility that a cheating prover could succeed. The idea is that as long as this probability is bounded away from 1, we can just repeat the protocol many times, and then the probability that the prover convinces the verifier *every* time becomes negligible.

Now, is the given (P,V) an IPS for $NISO$? Completeness is rather obvious. But, if $G \cong H$ then there is a problem. The verifier chooses a permutation π at random and thus produces a graph C uniformly from the set of all graphs isomorphic to both G_0 and G_1. Thus, the case that V picks G_0 and the case that V picks G_1 are indistinguishable to the prover. In a sense, this is like the "Mama mia!" game, which works as follows.The professor flips a coin with his back turned. If the result is heads, he turns around and says "Mama mia." If the result is tails, he turns around and says "Mama mia." If one were to attempt to guess the result of the coin flip, the "Mama mia" would be no help at all. Similarly, in this protocol, the fact that V gives a random graph isomorphic to both G_0 and G_1 is no help at all, so the prover P' has probability $1/2$ of being correct no matter what he does.

This first example of a proof system for $NISO$ shows that interactive proofs have power; no efficient proofs are known for showing membership in $NISO$ traditionally, but here is an interactive one. In fact, a theorem of Shamir [Sha90] shows that any language in $PSPACE$ (i.e., languages which can be decided by a Turing machine with unlimited time but only a polynomial amount of space in the size of its input) has an efficient IPS, and vice versa: any language in IP (the class of languages having IPS) is in $PSPACE$.

4 Another example [GMR85]

One drawback of the above example is that the prover is expected to be able to check whether two graphs are isomorphic for *any* two graphs, which seems unreasonable in that we would have to add a lot of computational power to a bounded prover to be able to execute the prover's side of that protocol.

However there is another way to look for more powerful proofs. As we said before, traditional proofs leave no room for privacy – if we send a proof of a theorem x then we convey much more than whether x is actually true or not. For one thing, if we actually give the proof to V, then V could turn around and convince someone else, which V was unable to do before seeing the proof.

We do have the traditional proof system for ISO (simply transmitting the permutation), but here is an interactive one. Assume the two parties both know a pair of graphs (G_0, G_1) which are isomorphic, and that the prover P knows ϕ such that $\phi(G_0) = G_1$.

1. P randomly generates a permutation π and computes $C = \pi(G_1)$, and sends this to V.
2. V picks a bit b at random and sends b to P.
3. If $b = 0$, P sends $\sigma = \pi \circ \phi$ to V, otherwise, P sends $\sigma = \pi$ to V.
4. V checks that $\sigma(G_b) = C$. If not, V rejects.

First of all, is this an IPS for ISO? If $G_0 \cong G_1$ then we note that $\pi(G_1) = C$ and $(\pi \circ \phi)(G_0) = \pi(\phi(G_0)) = C)$, so V will always accept what P sends. Secondly, if $G_0 \ncong G_1$ then either $C \ncong G_0$ or $C \ncong G_1$, so with probability 1/2, there is no permutation the adversarial prover can give that will fool the verifier.

But the real power of this protocol is that it leaks no information. If the prover were to give *both* $\pi \circ \phi$ and π in the same round then we could use these two permutations to recover ϕ. However, in a sense, both $\pi \circ \phi$ and π in isolation are just random permutations of the right size, and convey no information inherently.

However, in order to say this properly we will need careful definitions, which we don't have yet.

References

[GMR85] S. Goldwasser, S. Micali, and C. Rackoff. The Knowledge Complexity of Interactive Proof Systems. Proceedings of 17th ACM Symposium on the Theory of Computing (STOC), pp. 291-304, 1985.

[Sha90] A. Shamir. IP = PSPACE. Proceedings, 31st Annual IEEE Symposium on Foundations of Computer Science (FOCS), pp. 11-15, 1990.