1 Lecture 9: More on Zero-Knowledge Proofs

1.1 ZKP for GNI

Now, we return to graph non-isomorphism. Recall the interactive proof from before:

1. V generates a random bit b and a permutation π on G_b
2. V sends $H = \pi(G_b)$
3. P sends back a bit b'
4. V accepts if $b = b'$

And this is repeated k times.

Now, this might seem like zero-knowledge, but we must remember that zero-knowledge is trying to account for a dishonest verifier, so we have to consider all the ways the verifier might be dishonest, especially if the prover is being honest.

In particular, what if H is not generated properly. What if H is found somewhere else, and the verifier just sends it to see what it gets back from the prover? Well, if the prover is being honest and tests it against each of G_0 and G_1, and sends back a bit b', the verifier has obviously just learned something, since he didn't know whether H was even isomorphic to either of the two graphs before!

So, we're gonna have to add an intermediate step before P sends back b', where V proves that it knows what b is, from which it will follow that H was generated properly, so P sending b' will not be giving away any new information to V.

1.1.1 Proof of Knowledge

This is called a "proof of knowledge", and here's what we're going to do:

1. V generates two graphs, H_0 and H_1, isomorphic to G_0 and G_1, respectively.
2. V picks a random bit b, and sends the ordered pair $(H_b, H_{b'})$
3. Proof of Knowledge (repeat k times, possibly in parallel, for soundness):
   (a) V sends two graphs C_0, C_1, in any order.
   (b) P sends back a bit d
   (c) If $d = 1$, then V must show isomorphisms mapping one of the C's to G_0, and the other to G_1. If $d = 0$, then V must map the C's to the H's.
4. P sends a bit b'
5. V accepts if $b = b'$

If $G_0 \sim G_1$, we can see that the only way for V to pass the proof-of-knowledge with non-negligible probability is to actually have $H_0 \sim G_0$ and $H_1 \sim G_1$. Otherwise, on any given pair of C's, V wouldn't have an answer with probability $1/2$.

And, the proof-of-knowledge doesn't reveal anything. Suppose that it somehow revealed which H corresponds to which G. Then, P must have found an isomorphism between C's of different trials. But since all the isomorphic graphs are either identical to both $G_0 \sim H_0$ or both $G_1 \sim H_1$, this is just as difficult as actually figuring out what b is. So using this proof to cheat is just as difficult as actually solving for b as intended. So, the original soundness of the protocol remains.

It is easy to see that since an honest prover can still employ the same strategy as before, this protocol also remains perfectly complete.

Finally, we have that the protocol is zero-knowledge by the following strategy of a simulator.

Note that the verifier takes no input before sending the first pair of C's. So all randomness up to this point can be completely controlled to be the same, every time, by the simulator. In particular, the simulator can run the verifier once, and then ask for $d = 1$ to get a mapping of the C's to the G's. Then, it can start over and run the verifier again, with everything the same, except this time it asks for $d = 0$ to get a mapping of these same C's to the H's. Then, since the C's are the same, this reveals which H corresponds to which G, so the simulator can then proceed and correctly guess b.

1.2 ZKP of "AND", "OR"

If an honest prover wants to prove that X is true and Y is true, then it must prove that both statements are true. So, we don't risk anything by simply providing two separate ZKP's, one for X and one for Y.

But what about proving X is true *OR* Y is true? If only one is true, then we don't want to reveal which one is false, but we want to show that at least one is true.

If we are working with GNI and GI, we have strategies to make this work.

1.2.1 GNI "OR" ZKP

This protocol turns out to have a very simple idea. If we want to prove $G_0 \not\sim G_1$ OR $H_0 \not\sim H_1$, then we will simply run the ZKP's for both in parallel. We will have V choose the same bit b for both of them. Then, since the prover only has to reveal the bit, and the bit is the same for both, nothing is revealed about which graphs are isomorphic or not.

One interesting thing to consider is that if we somehow know our prover is honest, the verifier may cheat and figure out which pairs are actually isomorphic by picking different bits. However, this isn't actually feasible for a dishonest verifier, since this would allow a dishonest prover to be right with probability 1, so the soundness guarantee is ruined for the verifier if it tries this.

1.2.2 GI "OR" ZKP

Assuming $G_0 \sim G_1$ or $H_0 \sim H_1$, consider the following protocol:

1. P generates bits d_1, d_2, and permutations π_1, π_2.
2. P sends graphs $C = \pi_1(G_{d_1})$ and $D = \pi_2(H_{d_2})$.
3. V sends a random bit b
4. P sends permutations $\pi_1' : C \to G_{b_1}$ and $\pi_2' : C \to H_{b_2}$, where $b_1 \oplus b_2 = b$.

It is important to note that b_i and d_i are not necessarily the same, and that the only way for P to produce $b_i \neq d_i$ is to know an isomorphism between $G_0 \sim G_1$ or $H_0 \sim H_1$.

So, if P is dishonest, it will fail with probability $1/2$. And if it is honest, it knows one of the isomorphisms between the G's or H's, allowing it to produce any b via XOR. So this protocol has strong soundness ($1/2^k$ breaking probability) and perfect completeness.

And, since V only ever sees an isomorphism between C and one of the G's, and between D and one of the H's, there is no way for it to figure out which of the statements is true, except by figuring it out on its own. So, the protocol seems intuitively ZK.

In order to create a simulation, we may find that we have to add the same sort of commitment scheme as for the original GI ZKP, and once we do that, there should be no problem creating a simulation just as we did for the original GI ZKP.

1.3 Every **NP** Problem has a Zero-Knowledge Proof

Recall that Graph 3-coloring is NP-complete. That is, every problem in NP can be converted to an instance of a graph 3-coloring problem in polytime.

It follows that if we are able to show a general protocol for zero-knowledge proofs of graph 3-coloring, we will then have a way to prove any NP statement with a zero-knowledge proof. The only assumption we will need to make is a commitment scheme.

The basic idea is:

0. P has a 3-coloring of a graph
1. P permutes the colors on the graph and commits all the colors
2. V sends back one edge
3. P opens the colors of the two vertices incident to that edge
4. V accepts if the two opened colors are different

Steps 1-4 are repeated as many times as necessary to have a satisfactory level of soundness.

We can see easily that we have perfect completeness.

And, for a single run of the protocol a dishonest verifier will fail with probability $\geq \frac{1}{|E|}$, where E is the set of edges in the graph. Thinking in terms of Bernoulli trials, we expect to catch a dishonest verifier after $|E|$ trials, and over $k \cdot |E|$ trials, we expect a dishonest verifier to fail k times.

Similarly, a dishonest verifier gets away with being dishonest with probability $\leq \frac{|E|-1}{|E|}$ on any given trial. So thinking probabilistically we can see that the probability of a dishonest verifier getting away with being dishonest decreases exponentially as the number of trials increases linearly.

So, we have computational soundness.

The simulator is not difficult to construct either; the strategy is very similar to our original GI simulator. If the verifier is honest, we can simply get the edge before coloring the graph, which allows us to get it right every time. If the verifier is dishonest, we can do the rewind-and-cut strategy, and we expect to only have to do this $|E|$ times for each trial we want to produce, meaning the full simulation is still poly-time.