Exercises Zero-Knowledge (2)

1 Graph Isomorphism

Remember the zero-knowledge protocol we gave for graph isomorphism:
`Input:` graphs G and H
`Claim:` G and H are isomorphic (i.e., $G = \psi(H)$)
`Protocol:`

(1) Prover sends a random isomorphic copy $F = \varphi(G)$to the verifier;

(2) Verifier randomly selects G or H;

(3) Prover sends φ if the verifier selected G and $\varphi \circ \psi$ otherwise;

(4) Verifier accepts if the function he received is indeed an isomorphism from the graph he selected to F.

In class, we showed that this protocol is perfect zero-knowledge. Now it is up to you to show that it is an interactive proof system.

2 Almost-Perfect Zero-Knowledge

Recall the definition of almost-perfect zero-knowledge. Show that allowing M to output $\perp$ (like in perfect zero-knowledge) with probability bounded above by $\frac{1}{2}$ does not add to the power of this definition.

3 Perfect vs. Almost-Perfect Zero-Knowledge

Show that every perfect zero-knowledge system is also almost-perfect zero-knowledge.
Hint: show that the statistical dierence between M(x) and m*(x) is negligible.*

4 Alternative formulation of Zero-Knowledge

Consider the alternative definition of computational zero-knowledge, which uses the view of the interaction with the prover. With this we mean the entire interaction consisting of the messages sent back and forth, and the final output given by the verifier.

Computational Zero-Knowledge, alternative formulation *Let (P,V), L and V* be as in the original definition of computational zero-knowledge. We denote by* $\text{view}^P_{V^*}(x)$ *a random variable describing the content of the random tape of V* and the messages V* receives from P during a joint computation on common input x. We say that (P,V) is* **zero-knowledge** *if for every probabilistic polynomial-time interactive machine V* there exists a probabilistic polynomial-time algorithm M* such that the ensembles* $\{\text{view}^P_{V^*}(x)\}_{x\in L}$ *and* $\{M^*(x)\}_{x\in L}$ *are computationally indistinguishable.*

These definitions differ in that where the original version asks for $\{(P, V^*)(x)\}$, this one asks for $\text{view}^P_{V^*}(x)$. This means that we need different simulators for each of them.

It is clear that a simulator for the second one, that simulates the view of the verifier, can also simulate the output as it has access to the internal configuration of the verifier. However, the other way around also works.

Show that the original definition of computational zero-knowledge implies the alternative definition.
Hint: Show that for every probabilistic polynomial-time V^* there exists a probabilistic polynomial-time V^{**} such that $\text{view}_{V^*}^P = \langle P, V^* * \rangle(x)$.