CMSC 858K — Advanced Topics in Cryptography September 1, 2022

Lecture 13

Lecturer: Jonathan Katz *Scribe(s):* *Nagaraj Anthapadmanabhan*
Minkyoung Cho
Ji Sun Shin

1 Introduction

In the last few lectures, we introduced the hidden-bits model for non-interactive zero-knowledge (NIZK) and showed a conversion from any NIZK proof system in the hidden bits model to one in the real model, using trapdoor permutations. In this lecture, we complete the construction (which we had begin last lecture) of an NIZK proof system in the hidden-bits model. Putting these results together, we obtain our main goal: a construction of an NIZK proof system for any language in NP in the common random string model. As in the previous lecture, our presentation is based on [1, 2, 3, 4, 5] (As noted in the previous lecture, we focus on the case of non-adaptive NIZK but in fact the constructions given here achieve adaptively-secure NIZK as well.)

2 An NIZK Proof System in the Hidden-Bits Model

We begin by noting that in order to construct a proof system for an arbitrary language L in NP, it suffices to construct a proof system for a single NP-complete language.

Claim 1 *Given an NIZK proof system for an NP-Complete language L^*, we can construct an NIZK proof system for any language $L \in NP$.*

Proof Let $L \in NP$. Then there exists a polynomial-time function f such that

$$x \in L \Leftrightarrow f(x) \in L^*$$

(since L^* is NP-complete). So, given common input x, the prover and verifier can run the NIZK proof system for L^* on common input $f(x)$. Completeness and soundness clearly hold, and it is not too hard to see that zero-knowledge continues to hold as well. ■

For this reason, we now focus our attention on constructing an NIZK proof system for the NP-complete language of graphs containing Hamiltonian cycles (denoted HC).

2.1 Basic Construction

We review a basic construction of an NIZK proof system given in the last lecture. This proof system will be in a "modified" version of the hidden-bits model, where the hidden bits are chosen from a particular distribution which is not the uniform one. We then show how this "non-standard" distribution can be generated from a (long enough) sequence of uniformly-distributed bits — i.e., in the "actual" hidden-bits model.

Input: A directed graph $G = (V, E)$ with n vertices and containing a Hamiltonian cycle w which is known to the prover.

Hidden-bits string: Chosen uniformly from the set of strings representing n-vertex directed cycle graphs (using adjacency matrix representation). Call the given cycle graph C.

Prover: The prover chooses at random a permutation π on the vertices of G that lines up the Hamiltonian cycle w of G with the cycle in C. (This means that for every edge (i, j) in the cycle w, there is a corresponding edge $(\pi(i), \pi(j))$ in C.) The prover outputs π and also, for every non-edge in G, reveals a non-edge in the corresponding position (with respect to π) in C. Specifically, if (i, j) is a non-edge in G, then the prover reveals that entry $(\pi(i), \pi(j))$ in the adjacency matrix of C contains a "0" (i.e., is a non-edge).

Verifier: Verify that π is a permutation and also that for every non-edge in G, the prover has revealed the corresponding position in C (with respect to π), and this position contains a "0" (i.e., is a non-edge).

We now argue that this is indeed an NIZK proof system for graph Hamiltonicity:

Completeness: This follows by inspection. In particular, since an honest prover chooses a π mapping w to the cycle in C, and since C contains *only* this cycle (and no other edges), it will indeed be the case that all non-edges in G will map to non-edges in C.

Soundness: We show that the proof system as stated has *perfect* soundness (and so a prover has probability 0 of successfully proving a false statement). If the verifier accepts, this implies there is a permutation π and *some* cycle graph C (not necessarily known to the verifier) such that every non-edge in G corresponds to a non-edge in C. But then every edge in C corresponds to an edge in G. Since C is a cycle graph, this means that G contains a cycle. (Of course, this relies strongly on the fact that the hidden-bits string defines a cycle graph, but this is by assumption in the above proof system.)

Zero-knowledge: Let Sim be defined as follows:

$\underline{\mathsf{Sim}(G)}$
Pick a random permutation π on the vertices of G
Output π
For every non-edge in G, reveal a "0"

The output generated by Sim is *perfectly* indistinguishable from that generated by a real prover, assuming G is Hamiltonian. (Recall that indistinguishability is required to hold *only* in this case.) In particular: for a real prover, since C is a random cycle graph and π is randomly chosen from those mapping w onto the cycle in C, the permutation π is a random permutation (recall that the verifier never sees C). And both the real proof and the simulated proof reveal a "0" for all positions corresponding to non-edges in G.

2.2 Modified Construction

A problem with the previous construction is that we had assumed that the hidden-bits string was drawn uniformly from the set of (strings representing) cycle graphs. But in the actual hidden-bits model, the string is uniform (indeed, this property was used in the conversion from the hidden-bits model to the model in which a common random string is

available). So, we need to modify the previous construction; we do so by showing how to generate a random directed cycle graph from a uniform string with noticeable probability (here, "noticeable" means "inverse polynomial").

Toward this goal, we will work with *biased bits* that take on the value 1 with probability $1/4n^3$ and 0 otherwise. (Here and in what follows, we let n denote the number of vertices in the graph, as well as the security parameter.) It is easy to obtain such biased bits from a uniform string by simply parsing the original string in "chunks" of size $2 + 3\log_2 n$, and calling a "chunk" a 1 only if all bits are 1, and a 0 otherwise.

We use the following terminology:

Definition 1 A *permutation matrix* is a binary $n \times n$ matrix where each row and each column contains only a single 1. A *Hamiltonian matrix* is a permutation matrix that corresponds to a cycle; i.e., viewed as an adjacency matrix, it corresponds to a directed cycle graph. ♢

Consider the following procedure for generating a random Hamiltonian matrix from a string of biased bits of length $4n^4$: View the string as an $2n^2 \times 2n^2$ matrix M. We say that M is *useful* if it contains an $n \times n$ Hamiltonian sub-matrix and all other entries in M are 0. We show that this generates a Hamiltonian matrix with noticeable probability.

Claim 2 $\Pr[M \textit{ is useful}] = \Omega(\frac{1}{n^2})$.

Proof We first show that the probability that M has exactly n entries equal to 1 is $\Omega(1/n)$. We then show that the probability, conditioned on this event, that M has an $n \times n$ permutation sub-matrix is $\Omega(1)$. Finally, we show that the probability, conditioned on the prior two events, that the permutation sub-matrix is a Hamiltonian sub-matrix is $\Omega(1/n)$. Multiplying these probabilities proves the claim.

Let X be the random variable indicating how many entries in M are equal to 1. Since each entry of M is equal to 1 with probability $1/4n^3$, the expectation of X is n and the variance of X is $n \cdot (1 - 1/4n^3) = n - 1/4n^2$. Chebyshev's inequality thus shows that $X < 2n$ except with probability at most $1/n$. When $X < 2n$ then $X \in \{0, \ldots, 2n-1\}$ and the most-likely value is $X = n$; thus, the probability that $X = n$ is at least $(1 - 1/n)/2n = \Omega(1/n)$.

Assume M has exactly n entries equal to 1. Say two 1-entries *collide* if they are in the same row or column. The probability that any two particular 1-entries collide is at most $1/n^2$. Taking a union bound over all $\binom{n}{2}$ pairs of 1-entries shows that the probability of any colliding pairs is at most 1/2. Thus, the probability that M contains an $n \times n$ permutation sub-matrix is at least 1/2.

The probability that a random permutation matrix is a Hamiltonian matrix is easily seen to be $\frac{(n-1)!}{n!} = 1/n$. ■

Given the preceding claim, we now show the full construction of a proof system in the hidden-bits model.

Construction (A Proof System in the Hidden Bits Model for HC):

- Common input: A directed graph $G = (V, E)$ with $|V| = n$, where n is also the security parameter.

- Hidden-bits string : A uniform string of length $n^3 \cdot 4n^4 \cdot (2 + 3\log_2 n)$. This is parsed as n^3 matrices, each represented using $4n^4$ biased bits as stated above. Denote these matrices by $M_1, \ldots, M_{n^3}$.
- For each M_i, check if M_i is useful.
  - If not, reveal all the entries of M_i.
  - Otherwise (M_i is useful), let C_i denote the $n \times n$ Hamiltonian sub-matrix of M_i. Reveal all $4n^4 - n^2$ entries of M_i that are *not* in C_i. Also, use C_i to give a proof as described in Section 2.1.
- Verifier: (The verifier does not see the hidden string, but recall that it is given the positions of the bits revealed by the prover. So it makes sense to talk about the i^{th} matrix M_i even though the verifier does not necessarily see the entire matrix (i.e., besides what is revealed to it by the prover).) The verifier accepts only if the following are true for all n^3 matrices:
  - If the prover has revealed all of M_i, the verifier checks that M_i is not useful.
  - Otherwise, the prover checks that (i) the prover has revealed $4n^4 - n^2$ entries in M_i that are 0, while the remaining n^2 entries of M_i form an $n \times n$ sub-matrix; and (ii) call the remaining $n \times n$ sub-matrix C_i. The verifier verifies the prover's proof with respect to C_i exactly as in Section 2.1.

We now argue that the above is an NIZK proof system for HC in the hidden-bits model:

Completeness is immediate. For any M_i that is not useful, the prover can easily convince the verifier by simply revealing all entries. When M_i is useful, the argument in Section 2.1 holds.

Soundness is no longer perfect, but instead holds with all but negligible probability. Following the argument given in Section 2.1, soundness holds with probability 1 whenever at least one of the M_i are useful. The probability that none of the M_i are useful is at most

$$(1 - \Omega(1/n^2))^{n^3} \leq e^{-\Omega(n)},$$

which is negligible.

To show *zero-knowledge* we simply need to modify the simulator given in Section 2.1. The simulator now proceeds in n^3 sequential iterations as follows: in the i^{th} iteration, it generates $4n^4 \cdot (2+3\log_2 n)$ uniform bits. If this defines a matrix M_i which is *not* useful, the simulator simply outputs these bits and moves to the next iteration. If this defines a *useful* matrix M_i with Hamiltonian sub-matrix C_i, the simulator outputs all $4n^4 - n^2$ entries of M_i that are not in C_i (these entries are all 0), and then runs the simulator of Section 2.1. Note that this simulator will ignore C_i, and will instead just output a permutation π and "reveal" a 0 for every non-edge in G (as in Section 2.1).

References

[1] U. Feige. *Alternative Models for Zero-Knowledge Interactive Proofs*. PhD Thesis, Dept. of Computer Science and Applied Mathematics, Weizmann Institute of Science, 1990. Available from http://www.wisdom.weizmann.ac.il/~feige.

[2] U. Feige, D. Lapidot, and A. Shamir. Multiple Non-Interactive Zero-Knowledge Proofs Based on a Single Random String. In *FOCS*, pp. 308–317, 1990.

[3] U. Feige, D. Lapidot, and A. Shamir. Multiple Non-Interactive Zero-Knowledge Proofs Under General Assumptions. *SIAM Journal on Computing* 29(1): 1–28, 1999.

[4] O. Goldreich. *Foundations of Cryptography, vol. 1: Basic Tools*, Cambridge University Press, 2001.

[5] D. Lapidot and A. Shamir. Publicly Verifiable Non-Interactive Zero-Knowledge Proofs. In *Advances in Cryptology - CRYPTO '90*, pp. 353-365, 1990.

A Chebyshev's Inequality

Let X be a random variable with mean μ and variance σ^2. Chebyshev's inequality says that for any $k > 0$ we have

$$\Pr[|X - \mu| \geq k] \leq \frac{\sigma^2}{k^2}.$$