Derandomization in Cryptology

T-79.300 Postgraduate Course in Theoretical Computer Science

Seminar talk

Emilia Käsper

Overview

- Pseudorandom generators fooling nondeterministic circuits
- Hitting set generators as a weaker notion of pseudorandom generators
- Application 1: A witness indistinguishable one-message proof system
- Application 2: A noninteractive bit commitment scheme

Interactive proof systems

Let L be a **NP**-language. The (probabilistic polynomial-time) prover P wants to prove to the (PPT) verifier V the membership of $x \in L$. We require

- **completeness**: the prover (almost) never fails to prove the membership of valid inputs;
- **soundness**: the verifier (almost) never accepts invalid inputs.

Secure proof systems

Let W be a witness relation for L, i.e.

$$L = L(W) = \{x | \exists w\ (x, w) \in W\}.$$

Let $W(x) = \{w | (x, w) \in W\}$ be the *witness set* of x and call a $w \in W(x)$ a *witness* for x.

Zero-knowledge: prover, knowing a witness for $x \in L$ can convince the verifier to accept without revealing no information whatsoever except the fact that $x \in L$.

In particular, the verifier should learn nothing whatsoever about the witness.

ZK in cryptography

- Assume that the verifier is an adversary trying to gain knowledge.
- A dishonest verifier may compute its messages, using additional input from e.g. previous stages of the protocol.

Auxiliary-input zero-knowledge: no information is revealed, even if the verifier can use auxiliary input (but remains polynomial-time in the common input).

Can we remove interaction?

- Noninteractive ZK proofs require a shared random string selected by a trusted third party.
- From a truly noninteractive proof system, the verifier always gains the ability to prove the same statement to others.
- For auxiliary-input zero-knowledge, even two-message proofs are impossible.

Witness indistinguishability

- If there are two witnesses for $x \in L$, a proof system is **witness indistinguishable** if no polynomial time verifier — possibly nonuniform and using auxiliary input — can distinguish which of the two witnesses is being used by the prover.
- The verifier should not be able to distinguish, even if he knows both witnesses.
- Witness indistinguishability is preserved under parallel and concurrent composition of protocols (zero knowledge is not).

The goal

An **NP Proof System**

- consists of a single message from P to V;
- has a deterministic verifier; and
- satisfies perfect completeness and perfect soundness.

Trivial NP Proof: send the witness to the verifier.

The goal: an NP-proof system that is **witness indistinguishable**.

Preliminaries

Pseudorandom generators

- $G : \{0,1\}^l \to \{0,1\}^m$ is a (s, ϵ)**-pseudorandom generator against circuits** if for all circuits $C : \{0,1\}^m \to \{0,1\}$ of size at most s, it holds that

$$|Pr[C(G(U_l)) = 1] - Pr[C(U_m) = 1]| < \epsilon.$$

- Informally, we say that the generator *fools* circuits of size s.

Nondeterministic circuits

- A **nondeterministic Boolean circuit** $C(x, y)$ is a circuit that takes x as its primary input and y as a witness. For each x, we define $C(x) = 1$ if there exists a witness y such that $C(x, y) = 1$.
- A **co-nondeterministic Boolean circuit** $C(x, y)$ is a circuit that takes x as its primary input and y as a witness. For each x, we define $C(x) = 0$ if there exists a witness y such that $C(x, y) = 0$.

Blum-Micali-Yao type generators

- A function $G = \bigcup_m G_m : \{0,1\}^l \to \{0,1\}^m$ is a **BMY-type generator**, if G is computable in time poly(l) and for every constant c, G_m is a $(m^c, \frac{1}{m^c})$-pseudorandom generator for all sufficiently large m.
- In BMY-type generators, the adversarial circuit is allowed greater running time than the generator.
- Hence, a BMY-type generator cannot fool nondeterministic circuits.

Nisan-Widgerson type generators

- A function $G = \bigcup_m G_m : \{0,1\}^l \to \{0,1\}^m$ is a **NW-type generator**, if G is computable in time $2^{\mathcal{O}(l)}$ and G_m is a $(m^2, \frac{1}{m^2})$-pseudorandom generator for all m.
- In BMY-type generators, the generator is allowed greater running time than the adversarial circuit.
- NW-type generators can fool nondeterministic circuits.

Hitting set generators

- A hitting set generator outputs a set that intersects every dense set recognizable by a small circuit. Formally,
- H is an ϵ-hitting set generator against circuits, if for every circuit $C : \{0,1\}^m \to \{0,1\}$ of size at most s, the following holds. If $Pr[C(U_m) = 1] > \epsilon$ then there exists $y \in H(1^m, 1^s)$ such that $C(y) = 1$.
- H is *efficient* if its running time is polynomial in m and s.

HSG-s vs NW-type generators

- A pseudorandom generator $G : \{0,1\}^l \to \{0,1\}^m$ fooling circuits of size s induces a HSG, by taking the set of outputs over all seeds.
- The obtained HSG is efficient, if G is computable in time poly(s, m) and has logaritmic seed length $l = \mathcal{O}(\log m + \log s)$.
- HSG-s are allowed to run in greater time than the fooled circuits, thus they correspond to NW-type generators.

HSG-s vs NW-type generators

- If $\mathbf{E}$ has a function of circuit complexity $2^{\Omega(n)}$, then there exists a NW-type generator with logarithmic seed length (and thus polynomial running time).
- If $\mathbf{E}$ has a function of nondeterministic circuit complexity $2^{\Omega(n)}$, then there exists an efficient $\frac{1}{2}$-HSG against co-nondeterministic circuits.
- A similar result has been obtained for pseudorandom generators, but we are satisfied with a HSG.

End of preliminaries

Once again: the goal

An **NP Proof System**

- consists of a single message from P to V;
- has a deterministic verifier; and
- satisfies perfect completeness and perfect soundness.

Trivial NP Proof: send the witness to the verifier.

The goal: an NP-proof system that is **witness indistinguishable**.

Noninteractive zero-knowledge

- Doable assuming trapdoor permutations
- Requires a shared random string
- How to "generate" the random string δ?
- Let verifier choose a string B — might harm witness protection.
- Let prover choose C and set $\delta = B \oplus C$ — might violate soundness.
- Solution: balance both ideas

Protocol: a ZAP

First round: $V \to P$: The verifier sends to the prover m random strings $B_1, \ldots, B_m$. Denote this message by r.

Second round: $P \to V$: The prover chooses a random string C, defines $\delta_j = B_j \oplus C$ and sends to verifier m noninteractive proofs. Denote this message by π.

Final check: Verifier accepts if all proofs result in acceptance.

Properties of ZAP

The ZAP protocol is

- complete;
- sound;
- witness indistinguishable.

Note that at this point we don't require perfect soundness.

The construction

- Say that r is sound with respect to $x \notin L$ if there is no prover message π such that (x, r, π) is accepting.
- For every $x \notin L$, there exists a co-nondeterministic circuit C_x of size $p(n) < q(n)^2$ that outputs 1 iff r is sound with respect to x (where $q(n)$ is the running time of the honest verifier).
- Due to statistical soundness of the ZAP scheme, for every $x \notin L$, the probability that r is sound is larger than $\frac{1}{2}$.
- Equivalently, $Pr[C_x(U_{|r|}) = 1] > \frac{1}{2}$.
- Now we can use the HSG to hit a sound random string r, whenever $x \notin L$.

The protocol

Prover's message

1. Compute the hitting set $(r_1, \ldots, r_m)$.
2. Compute m responses π_i to verifier's messages r_i in a ZAP.
3. Send $(\pi_1, \ldots, \pi_m)$ to verifier.

Verifier's test

1. Compute the hitting set $(r_1, \ldots, r_m)$.
2. Run the ZAP verifier on prover's messages.
3. Accept if the ZAP verifier accepts all messages.

Properties of the protocol

The protocol is

- perfectly(?) complete;
- perfectly sound — we are guaranteed to hit a sound r for any $x \notin L$.
- witness indistinguishable — this property is preserved under parallel composition.

The main result

Assume that there exists an efficient $\frac{1}{2}$-HSG against co-nondeterministic circuits and that trapdoor permutations exist. Then every language in $\mathbf{NP}$ has a witness-indistinguishable $\mathbf{NP}$ proof system.

Note that the result can be restated, assuming the existence of NIZK systems (under the assumption of a shared random string) instead of the existence of trapdoor permutations.

Bit commitment schemes

First step: The sender gives the receiver a commitment to a secret bit b.

Second step: The sender decommits the bit b by revealing a secret key.

We require

- **hiding**: the commitment without the key must not reveal any information about b;
- **binding**: the sender is not able to decommit to a different bit $\overline{b}$.

Noninteractive bit commitment schemes

- There exists an interactive bit commitment scheme based on any one-way function.
- There exists a noninteractive bit commitment scheme based on any *one-to-one* one-way function.
- Can we relax the security requirements?

Interactive bit commitment scheme

- Let $G : \{0,1\}^k \to \{0,1\}^{3k}$ be a BMY-type pseudorandom generator computable in time k^d for some constant d.
- Such a generator can be constructed based on any one-way function.
- An interactive bit commitment scheme is given based on this generator.

Interactive bit commitment scheme

Commitment stage

1. **Receiver's step** Select a random $r \leftarrow \{0,1\}^{3k}$ and send r to sender.
2. **Sender's step** Select a random $s \leftarrow \{0,1\}^{k}$. If $b = 0$, send $\alpha = G(s)$ to receiver. Else, if $b = 1$, send $\alpha = G(s) \oplus r$ to receiver.

Decommitment stage Sender reveals s and b. Receiver accepts if $b = 0$ and $\alpha = G(s)$, or $b = 1$ and $\alpha = G(s) \oplus r$.

Derandomizing the scheme

- Define a string $r \in \{0,1\}^{3k}$ to be *good* for G if for all $s, s' \in \{0,1\}^k$, it holds that $G(s) \neq G(s') \oplus r$.
- As in the case of ZAP, the receiver does not need to send a random string; it is sufficient to send a "good" string.
- The probability that r is good is very high (due to binding property).

Derandomizing the scheme

- Good strings can be recognized by a polynomial-time co-nondeterministic uniform algorithm (running in time $3k^d$).
- If $\mathbf{E}$ has a function of nondeterministic circuit complexity $2^{\Omega(n)}$, then there exists an efficient $\frac{1}{2}$-HSG against co-nondeterministic uniform algorithms.
- We can use the HSG to always hit a good random string.

Noninteractive bit commitment scheme

Commitment stage

1. The sender computes the hitting set $(r_1, \ldots, r_m)$.
2. The sender chooses m strings $s_1, \ldots, s_m$ at random.
3. If $b = 0$, the sender sends $\alpha = (G(s_1), \ldots, G(s_m))$. Else, if $b = 1$, the sender sends $\alpha = G(s_1) \oplus r_1 \ldots, G(s_m) \oplus r_m$ to receiver.

Decommitment stage

1. Sender reveals b and $(s_1, \ldots, s_m)$.
2. Receiver accepts if $b = 0$ and $\alpha = (G(s_1), \ldots, G(s_m))$, or $b = 1$ and $\alpha = G(s_1) \oplus r_1 \ldots, G(s_m) \oplus r_m$.

The main result

Assume that there exists an efficient $\frac{1}{2}$-HSG against co-nondeterministic uniform algorithms and that one-way functions exist. Then there exists a noninteractive bit commitment scheme.

Conclusions

- The presented WI NP-proof is the first known noninteractive proof system for **NP** that satisfies a secrecy property.
- The presented bit commitment scheme relaxes underlying assumptions from previously known noninteractive bit commitment schemes: the existence of *any* one-way function is now required.