A Review of Super-Perfect Zero-Knowledge Proofs

Team 19
George Paul (2021121006),
Gaurav Singh (2020111014),
Akshit Gureja (2020112004)

Contents

The main objective of this project was to understand and review the topic of Super-Perfect Zero Knowledge Proofs. The paper under review is "Super-Perfect Zero-Knowledge Proofs" by Oded Goldreich and Liav Teichner.[1]

1 Preliminaries

1.1 Definitions

Proofs: The process by which the validity of an assertion is established.

Provers and Verifiers: During a proving process, The *prover* is the entity providing the proof. The notion of a *verifier* tends to be more explicitly in such discussions which typically emphasize the verification process.

Completeness: Completeness captures the ability of some prover to convince the verifier of true statements (belonging to some predetermined set of true statements).

Soundness: The soundness property asserts that the verification procedure cannot be 'tricked' into accepting false statements. It captures the verifier's ability to protect itself from being convinced of false statements (no matter what the prover does in order to fool the verifier).

Knowledge Gain: Consider a case in which a person Bob, ask Alice whether a decided graph is Eulerian (an easy to compute problem). Bob would gain no knowledge from Alice's answer since he could simply compute the solution himself. Suppose, the question was whether the graph is Hamiltonian. Alice giving her yes/no answer results in knowledge gain for Bob since Bob could not compute this easily himself.

Proof Systems: In general proof systems are defined in terms of the verification procedure. A 'proof' for a specific claim is always considered as coming from the outside (which can be viewed as another entity, called the prover). The verification procedure itself does not generate proofs but merely verifies their validity.

1.2 Interactive Proof Systems

- Interactive proof systems are intended to capture whatever can be efficiently verified via interaction with the outside.
- The verifier's verdict of whether to accept or reject a claim is probabilistic and hence a bounding error probability is allowed.
- Loosely speaking, we require that the prover be able to convince the verifier of the validity of true statements, while nobody can fool the verifier into believing false statements.
- Both of the aforementioned conditions are given a probabilistic interpretation: it is required that the verifier accepts valid statements with 'high' probability whereas the probability that it will accept a false statement is 'low.'
- The prover possesses unlimited computational resources but cannot be trusted, while the verifier has bounded computation power but is assumed to be always honest.

Definition (Interactive Proof System): A pair of interactive machines (P, V) is called an **interactive proof system for a language** L if machine V is polynomial time and the following two conditions hold:

Completeness: For every $x \in L$,

$$Pr[\langle P, V\rangle(x) = 1] \geq \frac{2}{3}$$

Soundness: For every $x \notin L$ and every interactive machine B,

$$Pr[\langle B, V\rangle(x) = 1] \leq \frac{1}{3}$$

Definition (Generalized Interactive Proof): Let $c, s : N \to R$ be functions satisfying $c(n) > s(n) + \frac{1}{p(n)}$ for some polynomial p. An interactive pair (P, V) is a generalized interactive proof system for the language L, with completeness bound c and soundness bound s. It satisfies the following conditions:

Completeness: For every $x \in L$,

$$Pr[\langle P, V \rangle(x) = 1] \geq c(|x|)$$

Soundness: For every $x \notin L$ and every interactive machine B,

$$Pr[\langle B, V \rangle(x) = 1] \leq s(|x|)$$

2 Zero Knowledge Proofs

2.1 Zero-knowledge proof systems

Zero-knowledge proofs are cryptographic protocols that enable one party, the prover, to prove to another party, the verifier, that a certain statement is true without revealing any additional information beyond the fact that the statement is indeed true. In other words, the verifier learns nothing other than the truth of the statement being proven.
The zero-knowledge property requires that there exists a simulator, Sim, that can generate a "fake" proof that is indistinguishable from a real proof, meaning that the verifier cannot tell the difference between a real proof and a fake proof generated by the simulator.
An interactive proof system (P, V) for a statement x is zero-knowledge if, for any polynomial-time verifier V* that interacts with the prover P on input x, there exists a polynomial-time simulator S that generates a transcript of the interaction that is indistinguishable from the actual transcript produced by P and V* on input x.
In this notation, P is the prover, V is the verifier, x is the statement being proven, and V* is a potentially malicious verifier who may try to gain additional information beyond the truth of the statement. The simulator S is a polynomial-time algorithm that generates a fake transcript of the interaction between P and V*, such that the fake transcript is indistinguishable from the actual transcript.
The interactive proof system involves multiple rounds of interaction between P and V. In each round, P sends a message to V, and V responds with a challenge

based on the previous messages. The goal is for P to convince V of the truth of the statement x, without revealing any additional information beyond the truth of the statement.
The zero-knowledge property of the proof system ensures that V* gains no additional information beyond the truth of the statement, even though P may have additional information. This is achieved by the existence of the simulator S, which generates a fake transcript that is indistinguishable from the actual transcript produced by P and V*.
Zero-knowledge proofs have many applications in cryptography, including password authentication, electronic voting, and secure transactions. They provide a powerful tool for proving the truth of a statement without revealing any additional information beyond the truth of the statement.
A zero-knowledge proof of some statement must satisfy three properties:

- **Completeness**: if the statement is true, an honest verifier (that is, one following the protocol properly) will be convinced of this fact by an honest prover.
- **Soundness**: if the statement is false, no cheating prover can convince an honest verifier that it is true, except with some small probability.
- **Zero-knowledge**: if the statement is true, no verifier learns anything other than the fact that the statement is true. In other words, just knowing the statement (not the secret) is sufficient to imagine a scenario showing that the prover knows the secret. This is formalized by showing that every verifier has some simulator that, given only the statement to be proved (and no access to the prover), can produce a transcript that "looks like" an interaction between an honest prover and the verifier in question.

The paper defines zero-knowledge proof systems by "the following oversimplified definition": An interactive proof system (P, V) for a set S is called zero-knowledge (ZK) if for every probabilistic polynomial-time strategy V* there exists a (strict) probabilistic polynomial-time algorithm (called a simulator) A* such that A* (x) is distributed identically to the output of V* after interacting with P on common input x.

2.2 Interactive ZKPs

An interactive proof system (P,V) for a set S is called zero-knowledge (ZK) if for every PPT strategy V* there exists a (strict) probabilistic PPT algorithm

(called a simulator) A* such that A*(x) is distributed identically to the output of V* after interacting with P on common input x.
We say that an interactive proof system (P,V) for a language L is zero-knowledge if whatever can be efficiently computed after interacting with P on input x ∈ L can also be efficiently computed from x (without any interaction). This holds with respect to any efficient way of interacting with P, not necessarily the way defined by the verifier program V.
Actually, zero knowledge is a property of the prescribed prover P. It captures P's robustness against attempts to gain knowledge by interacting with it.

2.3 Non-Interactive ZKPs

Let $\epsilon_c, \epsilon_s : \mathbf{N} \to [0,1)$ such that $\epsilon_c(l)$ and $\epsilon_s(l)$ are computable in poly(l)-time and $\epsilon_c(l) + \epsilon_s(l) < 1 - 1/poly(l)$. Let P and V be algorithms such that V is a PPT. Let 'p' be a positive polynomial. We say that (P,V) is a non-interactive proof system for a set S with completeness error and soundness error being ϵ_c, ϵ_s respectively if the following 2 conditions hold:

- Completeness: For every $x \in S$, it holds that

$$Pr_{\omega \leftarrow U_{p(|x|)}}[V(x, \omega, P(x, \omega)) = 1] \geq 1 - \epsilon_c(|x|)$$

- Soundness: For every $x \notin S$, and every function P*, it holds that

$$Pr_{\omega \leftarrow U_{p(|x|)}}[V(x, \omega, P^*(x, \omega)) = 1] \leq \epsilon_s(|x|)$$

s If $\epsilon_c \equiv 0$ then the system has perfect completeness

---

3 Perfect Zero Knowledge Proofs

3.1 Definition

Let (P, V) be an interactive proof system for some language. L. We say that. (P, V) is perfect zero-knowledge if for every PPT interactive machine. V^* there exists a PPTM M^* such that for every $x \in L$, the following two conditions hold:

1. With probability at most $\frac{1}{2}$, on input x, machine M^* outputs a special symbol $\perp$, that is:

$$Pr[M^*(x) = \perp] \leq \frac{1}{2}$$

2. Let $m^*(x)$ be a random variable describing the distribution of $M^*(x)$ conditioned on $M^*(x) \neq \perp$, that is:

$$Pr[m^*(x) = \alpha] = Pr[M*(x) = \alpha | M^*(x) \neq \perp, \forall \alpha \in \{0,1\}^*$$

Then the following random variables are identically distributed:

- $\langle P, V^* \rangle(x)$ (i.e., the output of the interactive machine V^* after interacting with the interactive machine P on a common input x)
- $m^*(x)$ (i.e., the output of machine M^* on input x, conditioned on not being $\perp$)

Machine M^* is called a perfect simulator of the interaction of V^* with P.
Condition 1 can be replaced by a stronger condition requiring that M* output the special symbol (i.e., $\perp$) only with negligible probability. For example, one can require that (on input x) machine M* will output $\perp$ with probability bounded above by $2 - p(|x|)$, for any polynomial $p(x)$;

3.2 Non-Interactive PZKPs

The system (P,V) is perfect zero-knowledge if there exists a (strict) PPT algorithm A such that for every $x \in S$, it holds that

$$Pr[A(x) = \perp] \leq 1/2$$

and

$$Pr[A(x) = \gamma | A \neq \perp] = Pr_{\omega \leftarrow U_{p(|x|)}}[(\omega, P(x, \omega)) = \gamma]$$

for every $\gamma \in \{0,1\}^*$.

---

4 Super-Perfect Zero Knowledge Proofs

4.1 Interactive Super-Perfect ZKPs

The system (P, V) is super perfect zero-knowledge if for every probabilistic polynomial time strategy V* there exists a (strict) probabilistic polynomial-time algorithm A* such that for every x ∈ S, it holds that A*(x) is distributed identically to $\langle P, V* \rangle(x)$

4.2 Non-Interactive Super-Perfect ZKPs

The system (P,V) is super perfect zero-knowledge if there exists a (strict) PPT algorithm A such that for every $x \in S$, it holds that A(x) is distributed identically to $(\omega, P(x, \omega))$ where $\omega \leftarrow U_{p(|x|)}$

4.3 From Perfect ZK to Super-Perfect ZK

Theorem 1: Suppose (P,V) is an interactive proof system for S and there exists a function $p : S \rightarrow [0, 0.5]$ such that for every PPT strategy V* there exists a PPT A* such that for every $x \in S$ it holds that

$$Pr[A^*(x) = \perp] \leq p(x)$$

and

$$Pr[A^*(x) = \gamma | A^*(x) \neq \perp] = Pr[< P, V^*(x) = \gamma]$$

for every $\gamma \in \{0,1\}^*$
Then S has a super perfect zero-knowledge proof system. Furthermore:

- The soundness error is preserved, and the increase in the completeness error is exponentially vanishing
- black-box simulation is preserved
- The communication complexities (number of rounds and length of messages) are preserved
- The new prover strategy can be implemented by a PPT oracle machine that is given access to the original prover strategy.

It is proved by observing the transformation proposed by Malka, which applies whenever all simulators fill with the same probability and not merely when the probability equals half.

4.3.1 Super-perfect zero-knowledge proof of exponentially vanishing completeness error.

We assume WLOG $p(x) < 2^{-|x|}$ for any $x \in S$. The transformation(construction below) lets the prover perform the original protocol with probability $1 - p(x)$ and abort otherwise. The verifier will reject in case the prover aborts and so perfect completeness is lost, but this allows for super-perfect simulation.
This transformation relies in the assumption that all simulators output $\perp$ with the same probability.
For sake of simplicity, we also assume WLOG that the original prover never sends the empty string denoted by λ.

THE TRANSFORMATION:
Let (P,V),S and p be as defined in Theorem 1 above and suppose that A' is a simulator for any finxed PPT strategy V'. On common input x, the two parties proceed as follows:

- The prover invokes A'(x) and sends the empty message λ if and only if A'(x) = $\perp$. In such a case, the verifier will reject
- Otherwise, the parties execute (P,V) on the common input x.

For every PPT strategy V*, consider the simulator A* guaranteed by the hypothesis of Theorem 1. On input x, the corresponding new simulator (for V*) computes $\gamma \leftarrow A^*(x)$, outputs γ if $\gamma \neq \perp$ *and* $V^*(x, \lambda)$ *otherwise*
Note that this protocol preserves the soundness error of V, whereas the completeness error on the input $x \in S$ increases by at-most $p(x) < 2^{-|x|}$ (i.e., from $\epsilon_c(|x|)$ *to* $p(x) + (1 - p(x).\epsilon_c(|x|))$. Indeed the verifier rejects if the prover got unlucky i.e. when A'(x) yields $\perp$ and so a cheating prover gainds nothing by claiming that it got $\perp$. The new simulators establish the super perfect ZK feature and hence theorem 1 follows.
Corollary 4.1 (honest verifier super-perfect ZK): Every set S that has an honest-verifier perfect ZK proof system has an honest-verifier super-perfect ZK proof system. All additional features of theorem 1 hold as well.
Corollary 4.2 (super-perfect ZK for NP): Assuming the existence of (non-uniformly strong) one-way permutations, every set in NP has a black-box super perfect ZK argument system.

4.3.2 Super-perfect zero-knowledge arguments with perfect completeness while assuming the existence of perfectly binding commitment schemes.

Let (P,V) be an argument system for S, let P_0 denote the strategy derived from P by having it abort just before sending the last message. We say that P is admissible if for every PPT strategy V* there exists a (strict) PPT algorithm A* such that for every $x \in S$ the following 3 conditions hold:

1. $Pr[A^*(x) = (1, .)] = Pr[A^*(x) = (0, .)] = 1/2$
2. $Pr[A^*(x) = (1, .)|A^*(x) = (1, .)] = Pr[< P, V^* > (x) = \gamma]$ for every $\gamma \in \{0, 1\}^*$
3. $Pr[A^*(x) = (0, .)|A^*(x) = (0, .)] = Pr[< P_0, V^* > (x) = \gamma]$ for every

In addition to requiring A* to output $\langle P_0, V\rangle(x)$ whenever it fails (Cond 3) we also require the failure probability to be exactly equal to one half. The latter condition can be assumed WLOG whenever p is efficiently computable provided that we adopt the non-standard model of PPT machines in which a machine can sample $[n]$ uniformly at the cost n.

THE TRANSFORMATION:
Let C denote the binding commitment. Let (P,V) be an argument system for S such that P is admissible according to the above 3 conditions. On common input x, the 2 parties proceed as follows:

1. The parties execute (P0,V) on common input x; that is; they invoke the original protocol, except that the prover doesn't send its last message, denoted by β.
2. The parties perform a standard coin tossing protocol. Specifically, the verifier sends a commitment $c \leftarrow C(v)$ to a random bit v, the prover responds (in the clear) with a random bit u, and the verifier de commits to the commitment (i.e. provides (v,s) such that $c = C_s(v)$)
3. If the verifier has de-committed improperly (i.e. $c \neq C_s(v)$) then the prover sends the empty message denoted by lambda. Otherwise u = v, the prover sends 0, and otherwise, it sends β (where we assume WLOG that $\beta \notin \{0, \lambda\}$.
4. If u = v, then the verifier accepts otherwise (i.e $u \neq v$) it acts as $V(\alpha, \beta)$ where α denotes the view of V in the interaction with P0 (as conducted in step 1)

5 Non-Interactive Super-Perfect Zero Knowledge Proofs

5.1 Getting Non-Interactive SPZKPs from Non-Interactive PZKPs

Although the construction in Section 4.3.1 is a valid Non-Interactive ZKP, the perfect completeness isn't preserved by it. This can be done by "transferring" the simulation attempt from the prover (who cannot be trusted to perform it at random) to a common reference string (which is uniformly distributed by definition).[1] Consider the following:

Construction: Consider a system (P, V) for S, an algorithm A and a polynomial-time computable function p. Let ρ be the length of the common reference string. and ρ' be the number of coins used by the simulator A. The new NIZK for inputs of length l is as follows:

Common Random String: An $(\rho(l) + \rho'(l))$-bit string, denoted (ω, r), where r is interpreted as an integer in $\{0, ..., 2\rho'(l) - 1\}$

Prover: (on input $x \in \{0,1\}^l$): If $r < p(x)2^{\rho'(l)}$, then the prover outputs the empty message λ. Otherwise (i.e. if, $r \geq p(x)2^{\rho'(l)}$), the prover outputs $P(x, \omega)$.

Verifier: (on input $x \in \{0,1\}^l$ and alleged proof y): If $r < p(x)2^{\rho'(l)}$, then the verifier accepts. Otherwise (i.e. if, $r \geq p(x)2^{\rho'(l)}$, the verifier decides according to $V(x, \omega, y)$.

The new simulator invokes $A(x)$ obtaining the value v. If $v = \perp$, then the simulator selects uniformly $\omega \in 0, 1^{\rho(l)}$ and $r \in \{0, ..., p(x)2^{\rho'(l)} - 1\}$, and outputs $((\omega, r), \lambda)$. Otherwise (i.e. if, $v = (\omega, y)$), the simulator selects uniformly $r \in p(x)2^{\rho'(l)}, ..., 2^{\rho'(l)} - 1$, and outputs $((\omega, r), y)$.

The Completeness error was found to be bounded by

$$(1 - p(x)) \cdot \epsilon_c(|x|) \leq \epsilon_c(|x|)$$

Whereas the soundness error is bounded by

$$p(|x|) + (1 - p(x)) \cdot \epsilon_s(|x|) \leq \epsilon_s(|x|) + 2^{-|x|}$$

Although, this construction relies on the ability to generate uniform distributions on the sets $[p(x) \cdot 2^{\rho'(l)}]$ and $[2^{\rho'(l)} - p(x) \cdot 2^{\rho'(l)}]$. This is possible only because we started with reducing $p(x) \leq 2^{-|x|}$. Instead we just start with $p \equiv 1/2$. The construction then becomes a Super-perfect zero knowledge proof in the standard PPT model.

---

6 A Super-Perfect NIZK for Blum Integers

6.1 Some Definitions

Blum Integers: A natural number is called a (generalized) Blum Integer if it is of the form $p^e q^d$ such that $p \equiv q \equiv 3 \pmod 4$ are different odd primes and $e \equiv d \equiv 1 \pmod 2$. The set of Blum integers is denoted B.

6.2 The Proof System

Construction
Input: A natural number n. Let $l \equiv \lceil \log_2 n \rceil$

Common Reference String: An l-bit string denoted ω, interpreted as an integer in Z_{2^l}

Prover: If $\omega \in S_n$ and there exists $r \in S_n$ such that $f_n(r) = \omega$, then the prover outputs r (otherwise it outputs 0).
Verifier: When receiving an alleged proof r, the verifier proceeds as follows.

1. if n is a prime power or $n \not\equiv 1 \pmod 4$, then the verifier halts outputting 0 (indicating reject).

2. The verifier checks if there exists a prime $p \in \{3, \cdots, l\}$ that divides n and finds the largest e such that p^e divides n .If n/p^e is not a prime power, then the verifier rejects. Otherwise letting q^d be this prime power (i.e. $n = p^e q^d$), the verifier accepts if $p \equiv q \equiv 3 \pmod 4$ and rejects otherwise.

3. If $\omega \notin S_n$, then the verifier halts, outputting 1 (indicating accept).

4. If $r \in S_n$ and $f_n(r) = \omega$, then the verifier outputs 1. Otherwise, it outputs 0.

The authors find this construction to be a super-perfect NIZK for Blum Integers with perfect completeness and a soundness error of 16/17. In short, the proof proceeds as follows:

1. First we find that the probability of the random string being in Z_n is $n/2^l > 1/2$

2. It is shown that $\omega \notin Z_n^*$ is unlikely. And if $\omega \in Z_n^*$, then the verifier rejects with probability of at least 1/2.

3. Since $|S_n| \leq |Z_n^*|/4$ (shown), we get that $\omega \in S_n$ with probability of at least $0.5 \cdot 0.99 \cdot 0.25 > 2/17$.

4. Next step is to prove that, $|f_n(S_n)| \leq |S_n|/2$, by showing that for each $r \in S_n$ there exists $s \in S_n$ such that $s \neq r$ and $f_n(s) = f_n(r)$

5. Now we can say that, If $n \notin B$, then the verifier rejects with probability at least $P_\omega[\omega \in S_n] \cdot P\omega[\omega \notin f(S_n) | \omega \in S_n] \geq (2/17) \cdot 0.5 = 1/17$. And hence Soundness error is 16/17

6.3 A Complete Promise Problem

The paper then cites Sahai and Vadan's work[2] and points out that they have received some analogous results for the class of promise problems that has statistical zero knowledge proof systems. It is proved that:

Theorem: $((U_{\text{yes}}, U_{\text{no}})$ is complete for $\text{SPNIZK}_1)$: Let SPNIZK_1 denote the class of promise problems having a super-perfect NIZK proof system of perfect completeness. Then,$(U_{\text{yes}}, U_{\text{no}})$ is in SPNIZK_1 and every problem in SPNIZK_1 is Karp-reducible to $(U_{\text{yes}}, U_{\text{no}})$.

6.4 Future Work

Some open problems problems were also highlighted, such as:

Open Problem: Let PZK be the class of sets having perfect zero-knowledge interactive proof system. Prove or disprove, under reasonable assumptions,

the conjecture by which not all sets in PZK have super-perfect zero-knowledge interactive proof systems. Ditto for zero-knowledge proof system with perfect completeness, where the question may refer both to the standard and non-standard models of PPT machines.

---

7 Conclusion

All in all, in this paper, we learned the basic vocabulary and implications of zero knowledge proofs and their next level in perfect zero knowledge proofs. We learned how to convert PZKPs into SPZKPs (both interactive ones and non-interactive ones). Using, the concept of Blum integers, another SPZKP system was innovated.

References

[1] Oded Goldreich and Liav Teichner. *Super-Perfect Zero-Knowledge Proofs*, pages 119–140. Springer International Publishing, Cham, 2020.

[2] Amit Sahai and Salil Vadhan. A complete promise problem for statistical zero-knowledge. *Journal of the ACM*, 50(2):196–249, 2003. Preliminary version in 38th FOCS, 1997.