A note on Constant-Round Zero-Knowledge Proofs for NP

Alon Rosen

Laboratory for Computer Science.
Massachusetts Institute of Technology.
200 Tech. Square, Cambridge, MA 02139 USA**
alon@lcs.mit.edu

Abstract. We consider the problem of constructing a constant-round zero-knowledge proof system for all languages in $\mathcal{NP}$. This problem has been previously addressed by Goldreich and Kahan (Jour. of Cryptology, 1996). Following recent works on concurrent zero-knowledge, we propose an alternative solution that admits a considerably simpler analysis.

Zero-knowledge ($\mathcal{ZK}$) protocols require no introduction. Since their conceptualization [10], they have become a widely used tool in the design and realization of many cryptographic tasks. The notion of zero-knowledge owes much of its wide applicability to its generality, and specifically, to the fact that every language in $\mathcal{NP}$ can be proved in $\mathcal{ZK}$ [11].

In this paper we consider the basic task of constructing a constant-round zero-knowledge interactive *proof* system for all languages in $\mathcal{NP}$ (with negligible error). Recall that an interactive proof system is required to protect the honest verifier from an all powerful prover that is trying to convince him of the validity of a false assertion. This should be contrasted with the case of an interactive *argument* system (cf. [3]), in which the soundness property is required to hold only w.r.t. computationally bounded provers.

Our goal is to design a "natural" protocol whose zero-knowledge property is demonstrated in as a simple as possible manner. This would be in contrast to previous solutions, which invloved a fairly complicated analysis (cf. Goldreich, Kahan [7]). Our solution is inspired by a new $\mathcal{ZK}$ protocol by Prabhakaran, Rosen and Sahai [18], originally introduced in the context of concurrent Zero-Knowledge. Constant-round, negligible-error, $\mathcal{ZK}$ proofs for $\mathcal{NP}$ are a fundamental and widely used cryptographic tool. Needless to say that a simple construction/analysis of such proofs would be most desirable.

1 Constructing a Constant-Round $\mathcal{ZK}$ Proof for $\mathcal{NP}$

We assume familiarity with the concepts of Interactive Proofs, Zero-Knowledge and Bit Commitment (see Appendix for the actual definitions) [10, 11, 15, 6]. The "typical" construction for a constant round interactive proof for any language

** Part of this work done while at the Weizmann Institute of Science, Israel.

in $\mathcal{NP}$ would use a protocol of the following sort as a building-block (here we use a protocol for the $\mathcal{NP}$-complete language of Hamiltonicity [2]).[3]

Common Input: A directed graph $G = (V, E)$ with $n \stackrel{\text{def}}{=} |V|$.
Auxiliary Input to Prover: A directed Hamiltonian Cycle, $C \subset E$, in G.

- $(\widehat{\mathtt{p1}})$: Pick a random permutation π of the vertices V and commit (using a perfectly binding commitment) to the adjacency matrix of the resulting permuted graph. That is, send an n-by-n matrix of commitments so that the $(\pi(i), \pi(j))^{\text{th}}$ entry is a commitment to 1 if $(i, j) \in E$, and is a commitment to 0 otherwise.
- $(\widehat{\mathtt{v1}})$: Send a randomly chosen bit $\sigma \in \{0, 1\}$.
- $(\widehat{\mathtt{p2}})$: If $\sigma = 0$, send π to the verifier along with the revealing (i.e., preimages) of all commitments. Otherwise, reveal only the commitments to entries $(\pi(i), \pi(j))$ with $(i, j) \in C$. In both cases also supply the corresponding decommitments.
- $(\widehat{\mathtt{v2}})$: If $\sigma = 0$, check that the revealed graph is indeed isomorphic, via π, to G. Otherwise, just check that all revealed values are 1 and that the corresponding entries form a simple n-cycle. In both cases check that the decommitments are proper (i.e., that they fit the corresponding commitments). Accept if and only if the corresponding condition holds.

Fig. 1. A 3-round interactive proof system for Hamiltonicity.

It can be seen that the above protocol is both complete and sound (with soundness error 1/2). An additional "useful" property of the protocol (which is also satisfied by many other known protocols) is that if the prover knows the contents of verifier's "challenge" message σ (sent in Step $(\widehat{\mathtt{v1}})$) prior to sending its own first message (sent in Step $(\widehat{\mathtt{p1}})$), then it is able to convince the verifier that G contains an Hamiltonian cycle even without knowing such a cycle (actually, it will convince the verifier even if G does not contain an Hamiltonian cycle).

Specifically, knowing in advance that $\sigma = 0$, the prover will commit to the entries of the adjacency matrix of the permuted graph (in Step $(\widehat{\mathtt{p1}})$), thus being able to reveal a permutation π and the preimages of all commitments in Step $(\widehat{\mathtt{p2}})$. On the other hand, knowing in advance that $\sigma = 1$, the prover will commit to the full graph K_n, thus being able to open an arbitrary cycle in the supposedly permuted graph.

The above "useful" property is sufficient in order to prove that the above protocol is black-box zero-knowledge. All that the simulator has to do is to try and "guess" the value of σ prior to determining the value of the prover's first message (and keep trying until it succeeds). Using the computational-hiding property of the prover's commitment in Step $(\widehat{\mathtt{p1}})$ we would then have that no matter what an adversary verifier V^* does, the simulator is expected to guess σ's value in a constant number of attempts.

[3] The choice of the Hamiltonicity protocol (due to Blum) as a building block is arbitrary (and is made just for clarity of presentation). In fact, any protocol with similar properties (such as the 3-coloring protocol of Goldreich, Micali and Wigderson [11]) could have been used.

To obtain a useful protocol, however, one must make sure that whenever the statement proved is false, V accepts only with small probability (rather than $1/2$). To achieve this, the protocol described above is repeated many (say, n) times independently. V accepts if and only if it has accepted in all n repetitions. The probability of having V accept a false statement is now reduced to $1/2^n$ (by the independence of the repetitions). To save on the number of rounds, the repetitions are conducted in *parallel* (rather than sequentially).

Unfortunately, repeating the protocol many times in parallel brings up the following difficulty. Whereas in the case of a single execution, the probability that the $\mathcal{ZK}$ simulator "guesses" the value of σ correctly is at least $1/2$, the probability that he does so *simultaneously* for all n repetitions is $1/2^n$. For large n, this probability will be very small and might cause the simulator to run for too long. Thus, it is not clear that the $\mathcal{ZK}$ property of the protocol is preserved. Indeed, the above protocol cannot be proved to be $\mathcal{ZK}$ using black-box simulation (unless $\mathcal{NP} \subseteq \mathcal{BPP}$) [8].[4]

The Goldreich-Kahan Analysis [7]. To overcome the above problem, an additional (V0) message is added at the beginning of the protocol, in which the verifier commits to all n "challenge" bits prior to receiving $\widehat{(\texttt{p1})}$. The verifier then decommits to all challenge bits in message $\widehat{(\texttt{v1})}$. The secrecy property of the commitment used in (V0) should then guarantee that the soundness of the protocol is preserved.

At this point, it seems that all that the simulator has to do after obtaining V^*'s commitments in message (V0) is to feed V^* with a "dummy" $\widehat{(\texttt{p1})}$ and then obtain decommittment to all challenge bits in message $\widehat{(\texttt{v1})}$. Knowing the challenge bits, the simulator would then "rewind" the interaction with V^* and resend a modified $\widehat{(\texttt{p1})}$ that would convince the verifier of the validity of the assertion (this is possible due to the "useful" property of the underlying protocol).

Unfortunately, V^* may arbitrarily deviate from the prescribed strategy. In particular, it may be the case that throughout its interaction with the prover (simulator), V^* occasionally sends an `ABORT` message (that is, V^* may potentially refuse to decommit to any of the previous commitments). Clearly, such an action on behalf of the verifier is considered illegal, and the interaction stops.

Having V^* refuse to decommit may seem as good news (since, once this happens, the simulator does not really need to do anything). The problem is that V^* does not *always* refuse to decommit (but may refuse with some probability $0 \leq p \leq 1$, which is not known in advance by the simulator). Thus, the simulator may find himself in a situation in which the first run is answered with `ABORT` whereas the second run is "properly answered". This means that the simulator has not managed to obtain the "challenge" bits in the first run, and it thus fails to complete its task.

[4] A recent result by Barak [1] suggests that black-box lower bounds should not be interpreted as impossibility results about $\mathcal{ZK}$, but rather as limitations of the black-box simulation as a technique for proving the $\mathcal{ZK}$ property of protocols. It should be noted, however, that Barak's protocol are only known to apply to certain kinds of *argument* systems (rather than proof systems).

One naïve solution would be to let the simulator always output the run in which V^* has refused to decommit. The problem with this solution is that it "skews" the distribution of transcripts outputted by the simulator towards transcripts that contain ill-formed messages.

Goldreich and Kahan [7] suggested to let the simulator always decide whether to output an aborted run according to the outcome of the first run. Specifically, the simulator will rewind only if "answered properly" in the first run and will continue doing so (i.e., rewinding) until it obtains another "proper answer". Unfortunately, while this simulation strategy guarantees that the simulator's output is correctly distributed, it also introduces technical difficulties. Loosely speaking, these difficulties arise from the fact that probability of V^* refusing to decommit might differ between the case it is fed with a "dummy" commitment (in step $(\widehat{\texttt{p1}})$) and the case it is fed with a "convincing" commitment. The solution to this problem is somewhat involved and requires having the simulator obtain an estimate on the probability of V^* decommits properly when fed with a "convincing" commitment in step $(\widehat{\texttt{p1}})$. As we have said before, our goal is to obtain a simpler analysis (even at the cost of analyzing a slightly different protocol).

2 The New Protocol

Consider the following protocol for Hamiltonicity (HC), which is a variant of the $c\mathcal{ZK}$ protocol by Prabhakaran, Rosen and Sahai [18] in which the preamble has only one iteration (rather than a super logarithmic number of iterations as in the PRS proocol).[5]

Common Input: A directed graph $G = (V, E)$ with $n \stackrel{\text{def}}{=} |V|$.
Auxiliary Input to Prover: A directed Hamiltonian Cycle, $C \subset E$, in G.
Additional parameter: A super-logarithmic function $k(n)$.
Stage 1: Commitment to challenge $\sigma \in \{0,1\}^n$ (independent of common input):
- (P1): Send first message for perfectly hiding commitment scheme.
- (V1): Commit to random σ, $\{\sigma_i^0\}_{i=1}^k$, $\{\sigma_i^1\}_{i=1}^k$ s.t. $\sigma_i^0 \oplus \sigma_i^1 = \sigma$ for all i.
- (P2): Send a random k-bit string $r = r_1, \ldots, r_k$.
- (V2): Decommit to $\sigma_1^{r_1}, \ldots, \sigma_k^{r_k}$.

Stage 2: Engage in the 3-round protocol for HC (n parallel repetitions) using $\sigma = \sigma_1, \ldots, \sigma_n$ as challenge:
- ($\texttt{p1}$): Produce first prover message of HC protocol (as in $(\widehat{\texttt{p1}})$).
- ($\texttt{v1}$): Decommit to σ and to $\{\sigma_i^{1-r_i}\}_{i=1}^k$.
- ($\texttt{p2}$): Answer σ with second prover message of HC protocol (as in $(\widehat{\texttt{p2}})$).
- ($\texttt{v2}$): Accept if and only if all corresponding conditions hold (as in $(\widehat{\texttt{v2}})$).

Fig. 2. A new 7-round, negligible error, $\mathcal{ZK}$ proof for Hamiltonicity.

As shown in [18], the above protocol is both complete and sound (with negligible error). In particular, the construction above is an interactive proof system for HC. The following theorem states that it is also $\mathcal{ZK}$.

[5] A related approach has been previously used in order to construct constant-round perfect $\mathcal{ZK}$ *arguments* for $\mathcal{NP}$ (see [5]).

Theorem 2.1 (Constant-round $\mathcal{ZK}$ proof for $\mathcal{NP}$) *Assume the existence of perfectly-hiding commitment schemes. Then, the protocol described in Figure 2 is a $\mathcal{ZK}$ proof system for HC.*

2.1 Zero-Knowledge

In order to demonstrate the $\mathcal{ZK}$ property of the protocol, we will show that there exists a "universal" black-box simulator, S, so that for every $G = (V, E) \in HC$ and adversary verifier V^* that runs in polynomial time (in $n = |V|$), $S(G)$ runs in expected time poly(n), and satisfies that the ensemble $\{\text{view}_{V^*}^{P}(G)\}_{G \in HC}$ is computationally indistinguishable from the ensemble $\{S^{V^*}(G)\}_{G \in HC}$.

The Simulator. On input $G = (V, E)$ with $n = |V|$, the simulator S starts by selecting and fixing a random tape $s \in \{0,1\}^{\text{poly}(n)}$ for V^*. It then proceeds by exploring various prefixes of possible interactions between P and V^*. This is done while having only black-box access to V^*.

```
Step (S1): Randomly generate (P1) and obtain (V1) = $V^*(G, (P1); s)$.
Step (S2): Randomly generate (P2) and obtain (V2) = $V^*(G, (P1), (P2); s)$.
    1. If (V2) $\neq$ ABORT, proceed to Step (S3).
    2. If (V2) = ABORT, output $\langle$(P1), (V1), ABORT$\rangle$ and stop.
Step (S3): For $j = 1, 2, \ldots$
    1. Randomly generate $(P2)_j$ and obtain $(V2)_j = V^*(G, (P1), (P2)_j; s)$.
    2. If $(V2)_j \neq$ ABORT, proceed to Step (S4).
    3. If $(V2)_j =$ ABORT continue.
  end(for)
Step (S4): Let (P2) = $r_1, \ldots, r_k$ be the prover message generated in Step (S2) of
    the simulation and let $(P2)_j = r'_1 \ldots, r'_k$ be the last prover message generated in
    Step (S3):
    1. If (P2) = $(P2)_j$, output $\perp$ and stop.
    2. If (P2) $\neq (P2)_j$, there exists $i \in \{1, \ldots, k\}$ so that $r_i \neq r'_i$. Let $\sigma = \sigma_i^{r_i} \oplus \sigma_i^{r'_i}$.
    3. Use $\sigma$ to produce an accepting transcript (p1), (v1), (p2) for $G \in HC$.
    4. Output $\langle$(P1), (V1), (P2), (V2), (p1), (v1), (p2)$\rangle$ and stop.
```

Fig. 3. The black-box simulator S.

Notice that simulator always picks the $(\text{P2})_j$ messages uniformly at random. Since the length of the (P2)'s is super-logarithmic, the probability that *any* two (P1) messages sent during the simulation are equal is negligible (see Section 2.1 for further details). We note that in previous simulators (cf. [7, 19, 13, 14]), the values of the (Pj) messages depended on the values revealed by the verifier in the corresponding (V2) answers, and were *not* chosen uniformly and independently each time. This is the main reason in the complication of previous analysises of the simulator's output distribution.

The simulator's running time. For any $G \in HC$, for any choice of s and of (P1), let $\zeta = \zeta(G, (\mathrm{P1}), s)$ denote the probability that the verifier V^* does not send an `ABORT` message in message (V2). The probability ζ is taken over the random choices of message (P2). (Or, in other words, over the coin-tosses used by the simulator to generate (P2) during the simulation (both in Steps (S2) and (S3).1).)

Using this notation, the simulator proceeds to Step (S3) with probability ζ and is then expected to reach Step (S4) after repeatedly rewinding in Step (S3).1 for $1/\zeta$ times (since the probability of successfully rewinding in each one of the rewinds is precisely ζ, independently of other rewinds). For $i \in \{1, 2, 3, 4\}$, let $p_i(\cdot)$ be a polynomial bound on the work required in order to perform Step (Si) of the simulation (where in Step (S3), the value $p_3(\cdot)$ represents the work of a single execution of Step (S3).1). The expected running time of the simulator is then:

$$\begin{aligned}
&p_1(n) + (1-\zeta) \cdot p_2(n) + \zeta \cdot \left(p_2(n) + \frac{1}{\zeta} \cdot p_3(n) + p_4(n)\right) \\
&\leq p_1(n) + p_2(n) + p_3(n) + p_4(n) \\
&= \mathrm{poly}(n)
\end{aligned}$$

Since the above holds for any choice of s and (P1), then it is also true for randomly chosen s and (P1) (and offcourse for any $G \in HC$). We thus have,

Proposition 2.2 *The simulator S runs in expected polynomial-time (in $|V|$).*

The simulator's output distribution. We now turn to show that for every $G \in HC$, the simulator's output distribution is computationally indistinguishable from V^*'s view of interactions with the honest prover P. Specifically,

Proposition 2.3 *Suppose that the commitment used in Step* (`p1`) *is computationally hiding. Then, the ensemble $\{S^{V^*}(G)\}_{G \in HC}$ is computationally indistinguishable from the ensemble $\{\mathrm{view}^P_{V^*}(G)\}_{G \in HC}$.*

Proof: As a hybrid experiment, consider what happens to the output distribution of the simulator S if we (slightly) modify its simulation strategy in the following way: Suppose that on input $G = (V, E) \in HC$, the simulator S obtains a directed Hamiltonian Cycle $C \subset E$ in G (as auxiliary input) and uses it in order to produce real prover messages whenever it reaches the second stage of the protocol. Specifically, when it reaches the second stage, the hybrid simulator checks whether the original simulator S should output $\perp$ (in which case it also does). If S does not have to output $\perp$, the hybrid simulator follows the prescribed prover strategy and generates prover messages for the corresponding second stage (by using the cycle it possesses rather than its prior knowledge of σ). We claim that the ensemble consisting of the resulting output (which we denote by $\widehat{S}^{V^*}(G, C)$) is computationally indistinguishable from $\{S^{V^*}(G)\}_{G \in HC}$. Namely,

Claim 2.4 *Suppose that the commitment used in Step* (p1) *is computationally hiding. Then, the ensemble* $\{S^{V^*}(G)\}_{G \in HC}$ *is computationally indistinguishable from the ensemble* $\{\widehat{S}^{V^*}(G, C)\}_{G \in HC}$.

Proof Sketch: The claim is proved by reducing the proof to the indistinguishability of Blum's simulator's output (that is, if the output of Blum's simulator [2] is computationally indistinguishable from the view of real executions of the basic Hamiltonicity proof system, then $\{S^{V^*}(G)\}_{G \in HC}$ and $\{\widehat{S}^{V^*}(G, C)\}_{G \in HC}$ are indistinguishable as well). The latter is proved to hold based on the computational-hiding property of the commitment scheme that is used by the prover in Step $(\widehat{\texttt{p1}})$ (see [2, 6] for further details). Here we also use the extra property that the output of Blum's simulator is indistinguishable from true interactions even if the distinguisher has a-priori knowledge of a Hamiltonian cycle $C \subset E$. ■

We next consider what happens to the output distribution of the hybrid simulator $\widehat{S}$ if we assume that it does not output $\perp$. It turns out that in such a case, the resulting output distribution is *identical* to the distribution of $\{\text{view}^P_{V^*}(G)\}_{G \in HC}$. Namely,

Claim 2.5 *The ensemble* $\{\widehat{S}^{V^*}(G, C)\}_{G \in HC}$ *conditioned on it not being* $\perp$, *is identically distributed to the ensemble* $\{\text{view}^P_{V^*}(G)\}_{G \in HC}$.

Proof: Notice that the first stage messages that appear in the output of the "original" simulator (that is, S) are identically distributed to the first stage messages that are produced by an honest prover P (since they are uniformly and independently chosen). Since the first stage messages that appear in the output of the "modified" simulator (that is, $\widehat{S}$) are identical to the ones appearing in the output of S, we infer that they are identically distributed to the first stage messages that are produced by an honest prover P. Using the fact that the second stage messages that appear in the output of the "modified" simulator are (by definition) identically distributed to the second stage messages that are produced by an honest prover P, we infer that the ensemble $\{\widehat{S}^{V^*}(G, C)\}_{G \in HC}$ is identically distributed to $\{\text{view}^P_{V^*}(G)\}_{G \in HC}$. ■

As we will show in Proposition 2.7 below, $\widehat{S}$ outputs $\perp$ only with negligible probability. In particular, the ensemble $\{\widehat{S}^{V^*}(G, C)\}_{G \in HC}$ is computationally indistinguishable from (and in fact statistically close to) the ensemble $\{\widehat{S}^{V^*}(G, C)\}_{G \in HC}$, conditioned on it not being $\perp$. Namely,

Claim 2.6 *The ensemble* $\{\widehat{S}^{V^*}(G, C)\}_{G \in HC}$ *is computationally indistinguishable from the ensemble* $\{\widehat{S}^{V^*}(G, C)\}_{G \in HC}$ *conditioned on it not being* $\perp$.

As mentioned above, Claim 2.6 follows by establishing the following claim.

Claim 2.7 *For any* $G = (V, E) \in HC$, *the probability that* $\widehat{S}^{V^*}(G, C) = \perp$ *is negligible* (*in* $|V|$).

Proof: Let $G \in HC$ with $n = |V|$. We will show that for any choice of $s \in \{0,1\}^{\mathrm{poly}(n)}$ and (P1) the probability of $\widehat{S}$ outputting $\perp$ (over random choices of $(\mathrm{P2}) = r \in \{0,1\}^k$) is precisely $1/2^k$. Since k is super-logarithmic it will immediately follow that the probability that $\widehat{S}^{V^*}(G,C) = \perp$ is negligible. Let $\widetilde{V}^* = \widetilde{V}^*((\mathrm{P1}),s)$ denote the "residual" strategy of V^* when $\langle(\mathrm{P1}),s\rangle$ are fixed (i.e., $\widetilde{V}^*(G,r) \stackrel{\mathrm{def}}{=} V^*(G,(P1),r;s)$), and let ζ be as in Section 2.1. We then have:

$$\begin{aligned}
&\Pr_r\left[\widehat{S}^{\widetilde{V}^*}(G,C) = \perp\right] \\
&= \Pr_r\left[\widehat{S}^{\widetilde{V}^*}(G,C) = \perp \;\middle|\; \widehat{S} \text{ reaches (S3)}\right] \cdot \Pr_r\left[\widehat{S} \text{ reaches (S3)}\right] \qquad (1)\\
&= \Pr_r\left[\widehat{S}^{\widetilde{V}^*}(G,C) = \perp \;\middle|\; \widehat{S} \text{ reaches (S3)}\right] \cdot \zeta \\
&= \Pr_r\left[(\mathrm{P2}) = (\mathrm{P2})_j\right] \cdot \zeta \qquad (2)
\end{aligned}$$

Now, since (P2) and $(\mathrm{P2})_j$ are uniformly and independently chosen in $\{0,1\}^k$, and since the number of $r \in \{0,1\}^k$ for which $\widetilde{V}^*(G,r)$ is not equal to `ABORT` is precisely $2^k \cdot \zeta$, then it holds that $\Pr[(\mathrm{P2}) = (\mathrm{P2})_j] = 1/(2^k \cdot \zeta)$. Using Eq. 2 we infer that:

$$\Pr_r\left[\widehat{S}^{\widetilde{V}^*}(G) = \perp\right] \;=\; \frac{1}{2^k \cdot \zeta} \cdot \zeta \;=\; \frac{1}{2^k}$$

as required. ■

It can be seen that Claims 2.4, 2.5 and 2.6 imply Proposition 2.3. ■

Acknowledgements. I would like to thank Oded Goldreich, Yehuda Lindell and Moni Naor for helpful conversations on the subject.

References

1. B. Barak. How to go Beyond the Black-Box Simulation Barrier. In *42nd FOCS*, pages 106–115, 2001.
2. M. Blum. How to prove a Theorem So No One Else Can Claim It. *Proc. of the International Congress of Mathematicians,* Berekeley, California, USA, pages 1444-1451, 1986.
3. G. Brassard, D. Chaum and C. Crépeau. Minimum Disclosure Proofs of Knowledge. *JCSS*, Vol. 37, No. 2, pages 156–189, 1988.
4. I. Damgard, T. Pedersen and B. Pfitzmann. On the Existence of Statistically Hiding Bit Commitment Schemes and Fail-Stop Signatures. In *Crypto93*, Springer LNCS 773, pages 250–265, 1993.
5. U. Feige. Ph.D. thesis, Alternative Models for Zero Knowledge Interactive Proofs. Weizmann Institute of Science, 1990.
6. O. Goldreich. *Foundations of Cryptography – Basic Tools.* Cambridge University Press, 2001.

7. O. Goldreich and A. Kahan. How to Construct Constant-Round Zero-Knowledge Proof Systems for NP. *Jour. of Cryptology*, Vol. 9, No. 2, pages 167–189, 1996.
8. O. Goldreich and H. Krawczyk. On the Composition of Zero-Knowledge Proof Systems. *SIAM J. Computing*, Vol. 25, No. 1, pages 169–192, 1996.
9. O. Goldreich and Y. Oren. Definitions and Properties of Zero-Knowledge Proof Systems. *Jour. of Cryptology*, Vol. 7, No. 1, pages 1–32, 1994.
10. S. Goldwasser, S. Micali and C. Rackoff. The Knowledge Complexity of Interactive Proof Systems. *SIAM J. Comput.*, Vol. 18, No. 1, pp. 186–208, 1989.
11. O. Goldreich, S. Micali and A. Wigderson. Proofs that Yield Nothing But Their Validity or All Languages in NP Have Zero-Knowledge Proof Systems. *JACM*, Vol. 38, No. 1, pages 691–729, 1991.
12. J. Hastad, R. Impagliazzo, L.A. Levin and M. Luby. Construction of Pseudorandom Generator from any One-Way Function. *SIAM Jour. on Computing*, Vol. 28 (4), pages 1364–1396, 1999.
13. J. Kilian and E. Petrank. Concurrent and Resettable Zero-Knowledge in Polylogarithmic Rounds. In *33rd STOC*, pages 560–569, 2001.
14. D. Micciancio and E. Petrank. Simulatable Commitments and Efficient Concurrent Zero-Knowledge. In *EUROCRYPT03*, Springer LNCS 2656, pages 140–159, 2003.
15. M. Naor. Bit Commitment using Pseudorandomness. *Jour. of Cryptology*, Vol. 4, pages 151–158, 1991.
16. M. Naor, R. Ostrovsky, R. Venkatesan and M. Yung. Zero-Knowledge Arguments for NP can be Based on General Assumptions. *Jour. of Cryptology*, Vol. 11, pages 87–108, 1998.
17. M. Naor and M. Yung. Universal One-Way Hash Functions and their Cryptographic Applications. In *21st STOC*, pages 33–43, 1989.
18. M. Prabhakaran and A. Rosen and A. Sahai. Concurrent Zero Knowledge with Logarithmic Round-Complexity. In *43rd FOCS*, pages 366-375, 2002.
19. R. Richardson and J. Kilian. On the Concurrent Composition of Zero-Knowledge Proofs. In *EuroCrypt99*, Springer LNCS 1592, pages 415–431, 1999.

A Definitions

A.1 Basic notation

We let N denote the set of all integers. For any integer $k \in N$, denote by $[k]$ the set $\{1, 2, \ldots, k\}$. For any $x \in \{0,1\}^*$, we let $|x|$ denote the size of x (i.e., the number of bits used in order to write it). For two machines M, A, we let $M^A(x)$ denote the output of machine M on input x and given oracle access to A. The term negligible is used for denoting functions that are (asymptotically) smaller than one over any polynomial. More precisely, a function $\nu(\cdot)$ from non-negative integers to reals is called negligible if for every constant $c > 0$ and all sufficiently large n, it holds that $\nu(n) < n^{-c}$.

A.2 Interactive Proofs

We use the standard definitions of interactive proofs (and interactive Turing machines) [10, 6] and arguments (a.k.a computationally-sound proofs) [3]. Given a pair of interactive Turing machines, P and V, we denote by $\langle P, V\rangle(x)$ the random variable representing the (local) output of V when interacting with machine P on common input x, when the random input to each machine is uniformly and independently chosen.

Definition A.1 (Interactive Proof System) *A pair of interactive machines $\langle P, V\rangle$ is called an* interactive proof system *for a language L if machine V is polynomial-time and the following two conditions hold with respect to some negligible function $\nu(\cdot)$:*

– Completeness: *For every $x \in L$,*

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

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

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

In case that the soundness condition is required to hold only with respect to a computationally bounded prover, $\langle P, V\rangle$ is called an interactive argument *system.*

A.3 Zero-Knowledge

Loosely speaking, an interactive proof is said to be *zero-knowledge* ($\mathcal{ZK}$) if it yields nothing beyond the validity of the assertion being proved. This is formalized by requiring that the view of every probabilistic polynomial-time adversary V^* interacting with the honest prover P can be simulated by a probabilistic polynomial-time machine S_{V^*} (a.k.a. the *simulator*). The idea behind this definition is that whatever V^* might have learned from interacting with P, he could have actually learned by himself (by running the simulator S). The transcript of an interaction consists of the common input x, followed by the sequence of prover and verifier messages exchanged during the interaction. We denote by $\text{view}^P_{V^*}(x)$ a random variable describing the content of the random tape of V^* and the transcript of the interaction between P and V^* (that is, all messages that V^* sends and receives during the interaction with P, on common input x).

Definition A.2 (Zero-Knowledge) *Let $\langle P, V\rangle$ be an interactive proof system for a language L. We say that $\langle P, V\rangle$ is* zero-knowledge*, if for every probabilistic polynomial-time interactive machine V^* there exists a probabilistic polynomial-time algorithm S_{V^*} such that the ensembles $\{\text{view}^P_{V^*}(x)\}_{x\in L}$ and $\{S_{V^*}(x)\}_{x\in L}$ are computationally indistinguishable.*

To make Definition A.2 useful in the context of protocol composition, Goldreich and Oren [9] suggested to augment the definition so that the corresponding conditions hold also with respect to all $z \in \{0,1\}^*$, where both V^* and S_{V^*} are allowed to obtain z as auxiliary input. Jumping ahead, we comment that in the context of black-box simulation,, the original definition implies the augmented one (i.e., any black-box $\mathcal{ZK}$ protocol is also $\mathcal{ZK}$ w.r.t. auxuliary inputs). Since in this work we only consider the notion of black-box $\mathcal{ZK}$, we may ignore the issue of auxiliary inputs while being guaranteed that all results hold with repsect to the augmented definition as well.

A.4 Black-Box zero-Knowledge

Loosely speaking, the definition of black-box zero-knowledge requires that there exists a "universal" simulator, S, so that for every $x \in L$ and every probabilistic polynomial-time adversary V^*, the simulator S produces a distribution that is indistinguishable from $\mathrm{view}_{V^*}^{P}(x)$ while using V^* as an oracle (i.e., in a "black-box" manner). Essentially, the definition of black-box simulation says that the black-box simulator mimics the interaction of the prover P with any polynomial-time verifier V^* relative to any random input r it might choose. The simulator does so merely by using oracle calls to $V^*(x;r)$ (which specifies the next message that V^* sends on input x and random input r). The simulation is indistinguishable from the true interaction even if the distinguisher (i.e., D) is given access to the oracle $V^*(x;r)$. For more details see Section 4.5.4.2 of [6].

Definition A.3 (Black-Box Zero-Knowledge) *Let $\langle P, V\rangle$ be an interactive proof system for a language L. We say that $\langle P, V\rangle$ is* black-box zero-knowledge, *if there exists a probabilistic polynomial-time algorithm S, so that for every probabilistic polynomial-time interactive machine V^*, the ensembles $\{\mathrm{view}_{V^*}^{P}(x)\}_{x\in L}$ and $\{S^{V^*}(x)\}_{x\in L}$ are computationally indistinguishable.*

A.5 Commitment Schemes

Commitment schemes are used to enable a party, known as the *sender*, to commit itself to a value while keeping it secret from the *receiver* (this property is called hiding). Furthermore, the commitment is binding, and thus in a later stage when the commitment is opened, it is guaranteed that the "opening" can yield only a single value determined in the committing phase.

Perfectly-binding commitments. In a perfectly binding commitment scheme, the binding property holds even for an all-powerful sender, while the hiding property is only guaranteed with respect to a polynomial-time bounded receiver.

Non-interactive perfectly-binding commitment schemes can be constructed using any 1–1 one-way function (see Section 4.4.1 of [6]). Allowing interaction (in which the receiver first sends a single message), (almost) perfectly-binding commitment schemes can be obtained from any one-way function [15, 12].

Perfectly-hiding commitments. In a perfectly hiding commitment scheme, the binding property is guaranteed to hold only with respect to a probabilistic polynomial-time sender. On the other hand, the hiding property is information-theoretic. That is, the distributions of commitments to 0 and commitments to 1 are identical (statistically-close), and thus even an all-powerful receiver cannot know the value committed to by the sender. (See Section 4.8.2 of [6].)

Perfectly hiding commitment schemes can be constructed from any one-way permutation [16]. However, *constant-round* schemes are only known to exist under stronger assumptions; specifically, assuming the existence of collision-resistant hash functions [17, 4] or the existence of a collection of certified clawfree functions [7] (see also [6], Section 4.8.2.3).

This article was processed using the LaTeX macro package with LLNCS style