Public-Coin Parallel Zero-Knowledge for NP

Rafael Pass* Alon Rosen† Wei-Lung Dustin Tseng‡

August 8, 2011

Abstract

We show that, assuming the existence of collision-resistant hash functions, every language in NP has a constant-round public-coin zero-knowledge argument that remains secure under unbounded parallel composition (a.k.a. *parallel zero knowledge*.) Our protocol is a variant of Barak's zero-knowledge argument (FOCS 2001), and has a non-black-box simulator. This result stands in sharp constrast with the recent result by Pass, Tseng and Wikstrom (Crypto 2010) showing that only languages in BPP have public-coin parallel zero-knowledge arguments with black-box simulators.

1 Introduction

Zero-knowledge (ZK) interactive protocols [GMR89] are paradoxical constructs that allow one player P (called the prover) to convince another player V (called the verifier) of the validity of a mathematical statement $x \in L$, while providing zero additional knowledge to the verifier. A fundamental question regarding zero-knowledge protocols is whether their composition remains zero-knowledge. Three basic notions of compositions are sequential composition [GMR89, GO94], parallel composition [FS90, GK96b] and concurrent composition [FS90, DNS04]. In a sequential composition, the players sequentially run many instances of a zero-knowledge protocol, one after the other. In a parallel composition, the instances instead proceed in parallel, at the same pace. Finally, in a concurrent composition, messages from different instances of the protocol may be arbitrarily interleaved. While the definition of ZK is closed under sequential composition [GO94], this no longer holds for parallel composition [GK96b]. However, there are zero-knowledge protocols for all of NP that have been demonstrated to be secure under both parallel and concurrent composition [FS90, GK96a, RK99, KP01, PRS02].

Whereas the original ZK protocols of [GMR89, GMW91, Blu86] are public-coin—i.e., the verifier's messages are its random coin-tosses—all of the aforementioned parallel or concurrent ZK protocols use private coins. Recently, Pass, Tseng and Wikstrom [PTW09] show that this is no

*Cornell University. Email: rafael@cs.cornell.edu. Work supported in part by a Alfred P. Sloan Fellowship, Microsoft New Faculty Fellowship, NSF CAREER Award CCF-0746990, AFOSR Award FA9550-08-1-0197, BSF Grant 2006317.

†Efi Arazi School of Computer Science. IDC Herzliya, Israel. Email: alon.rosen@idc.ac.il. Work supported in part by BSF grant No. 2006317.

‡Cornell University. Email: wdtseng@cs.cornell.edu. Work supported in part by a NSF graduate research fellowship.

coincidence—only languages in BPP have public-coin black-box parallel ZK protocols (that is, protocols that remain ZK under parallel composition). This leaves open the question of whether non-black-box simulation techniques can be used to bypass this impossibility result.

In this paper we resolve this question. Namely, we show the existence of a public-coin parallel ZK argument for NP, relying on non-black-box simulation techniques. Our protocol only requires a constant-number of rounds and is based on the existence of collision-resistant hash-functions.

Theorem 1. *Assume the existence of collision-resistant hash-functions. Then there exists a constant-round public-coin parallel zero-knowledge argument for NP.*

Our protocol is a variant of Barak's [Bar01] non-black-box zero-knowledge argument for NP. We mention that Barak's original protocol already handles a notion of *bounded-concurrent* composition; that is, it remains secure under an *a priori* bounded number of concurrent execution. In contrast, our protocol handles an unbounded number of executions, but only if it is parallel composition (as opposed to concurrent composition).

Why study unbounded parallel composition. A standard motivation for studying parallel composition of zero-knowledge proofs is to be able to prove multiple statements without increasing the number of communication rounds. At first sight, having a protocol that is secure under bounded parallel composition (just as Barak's original protocol) seems to suffice in such a scenario: Since the protocol designer knows a bound b on the number of statements to be proved, we may simply use a zero-knowledge protocol that is secure under b-bounded parallel composition. However, the communication (and/or computational) complexity of the protocol may depend on the bound b—indeed, in Barak's protocol, the communication complexity of the protocol grows linearly with b. On the other hand, if we have a protocol that remains secure under *unbounded* parallel composition, this inefficiency disappears.[1]

Additionally, we hope that insights gained from studying the simpler case of parallel composition will be helpful in studying the more complicated case of concurrent composition. Indeed, a beautiful example of when results about parallel composition can be translated into results about concurrent composition can be found in Goldreich's work [Gol02].

Main technique. Let us briefly recall the idea behind Barak's protocol (following a slight variant of this protocol due to [PR08]). Roughly speaking, for language L and common input $x \in \{0,1\}^n$, the prover P and verifier V proceed in two stages. In Stage 1, P starts by sending a computationally-binding commitment $c \in \{0,1\}^n$ to 0; V follows by sending a uniformly random "challenge" $r \xleftarrow{R} \{0,1\}^{2n}$. In Stage 2, P shows (using a witness indistinguishable argument of knowledge) that either there exists a "short" string $s \in \{0,1\}^n$ such that c is a commitment to a program Π such that $\Pi(s) = r$, or $x \in L$. Soundness follows from the fact that even if a malicious prover P^* tries to commit to some program Π (instead of committing to 0), with high probability, the string r sent by V will be different from $\Pi(s)$ for every string $s \in \{0,1\}^n$. To prove ZK, consider the non-black-box simulator that commits to the code of the malicious verifier V^*; note that by definition it holds that $\Pi(c) = V^*(c) = r$, and the simulator can use $s = c$ as a "fake" witness in the final proof.

[1] Furthermore, note that we if managed to construct a protocol where the communication and computational complexity depends only logarithmically on the bound b, then this protocol can easily be turned into a protocol that remains secure under unbounded composition: simply set $b = 2^n$, where n is the security parameter.

Now, let us consider parallel composition. That is, we need to simulate the view of a verifier that starts $m = \text{poly}(n)$ parallel executions of the protocol. The above simulator no longer works in this setting: the problem is that the verifier's code is now a function of *all* the commitments $\vec{c} = c_1, \ldots, c_m$ sent in the different executions. (Note that if we increase the length of r, and therefore the allowed length of s, we can handle a bounded number of execution, by simply letting $s = \vec{c}$). To get around this problem, we change the proof in the last stage as follows. Instead of proving the existence of a string s such that $\Pi(s) = r$, we show the existence of a seed $s \in \{0,1\}^n$ for a pseudorandom function f, and an index $i \in \{0,1\}^{\log^2 n}$, such that $\Pi_i(\vec{c}) = r$, where c_i is a commitment to Π using $f_s(i)$ as randomness, and $\Pi_i(x)$ is the "projection" of Π onto the i'th "coordinate"—i.e., the output of Π in the i'th parallel execution. To construct the zero-knowledge simulator, we start by picking a seed s, compute commitments c_i to Π using $f_s(i)$ as randomness, and use s and i as a "fake-witness" to simulate Stage 2 in execution i.

2 Preliminaries

Let $\mathbb{N}$ denote the set of positive integers, and $[n]$ denote the set $\{1, 2, \ldots, n\}$. We assume familiarity with interactive protocols.

2.1 Computational Indistinguishability

The following definition of computational indistinguishability originates in the seminal paper of Goldwasser and Micali [GM84]. Let X be a countable set of strings. A **probability ensemble indexed by** X is a sequence of random variables indexed by X. Namely, any element of $A = \{A_x\}_{x \in X}$ is a random variable indexed by X.

Definition 1 (Indistinguishability)**.** Let X and Y be countable sets. Two ensembles $\{A_{x,y}\}_{x \in X, y \in Y}$ and $\{B_{x,y}\}_{x \in X, y \in Y}$ are said to be **computationally indistinguishable over** $x \in X$, if for every probabilistic machine D (the distinguisher) whose running time is polynomial in its first input, there exists a negligible function $\nu(\cdot)$ so that for every $x \in X, y \in Y$:

$$|\Pr[D(x, y, A_{x,y}) = 1] - \Pr[D(x, y, B_{x,y}) = 1]| < \nu(|x|)$$

(In the above expression, D is simply given a sample from $A_{x,y}$ and $B_{x,y}$, respectively.) Note that indistinguisability applies to every index of the ensembles, even though only the size of x is used as the asymptotic measure.

2.2 Zero-Knowledge

An interactive proof is said to be **zero-knowledge** if it yields nothing beyond the validity of the statement being proved [GMR89]. In an interactive protocol (P, V), the **view** of the verifier in an interaction consists of the common input x, followed by its random tape and the sequence of prover messages that it receives. For any adversarial verifier V^*, let $\mathsf{View}_{V^*}\langle P, V^*(z)\rangle(x)$ be the random variable that denotes V^*'s view in an interaction with the honest prover P, when V^* is given auxiliary input z and both parties are given common input x.

Definition 2 (Zero-knowledge)**.** An interactive protocol (P, V) for language L is **zero-knowledge** if for every PPT adversarial verifier V^*, there exists a PPT simulator S such that the following ensembles are computationally indistinguishable over $x \in L$:

$$\{\mathsf{View}_{V^*}\langle P, V^*(z)\rangle(x)\}_{x\in L, z\in\{0,1\}^*} \approx \{S(x,z)\}_{x\in L, z\in\{0,1\}^*}$$

In this work we consider the setting of parallel composition. An m-session **parallel adversarial verifier** V^* is PPT machine that, on common input x and auxiliary input z, interacts with $m(|x|)$ independent **sessions** of P in parallel. While V^* must schedule the messages of different sessions in parallel, V^* may choose to abort in some sessions and continue the protocol in other sessions.

Definition 3 (Parallel Zero-knowledge)**.** An interactive protocol (P, V) for language L is **parallel zero-knowledge** if for every polynomial m and every PPT m-session parallel adversarial verifier V^*, there exists a PPT simulator S such that the following ensembles are computationally indistinguishable over $x \in L$:

$$\{\mathsf{View}_{V^*}\langle P, V^*(z)\rangle(x)\}_{x\in L, z\in\{0,1\}^*} \approx \{S(x,z)\}_{x\in L, z\in\{0,1\}^*}$$

2.3 Witness Indistinguishability

An interactive protocol is **witness indistinguishable** (WI) [FS90] if the verifier's view is "independent" of the witness used by the prover for proving the statement. In this context, we focus on languages $L \in \mathsf{NP}$ with a corresponding witness relation $\mathbf{R}_L$. Namely, we consider interactions in which on common input x the prover is given a witness in $\mathbf{R}_L(x)$. For any adversarial verifier V^*, let $\mathsf{View}_{V^*}\langle P(w), V(z)\rangle(x)$ be the random variable that denotes V^*'s view in an interaction with P, when V^* is given auxiliary input z, P is given witness w, and both parties are given common input x.

Definition 4 (Witness-indistinguishability)**.** An interactive protocol (P, V) for $L \in \mathsf{NP}$ is **witness indistinguishable** for $\mathbf{R}_L$ if for every PPT adversarial verifier V^*, and for every two sequences $\{w_x^1\}_{x\in L}$ and $\{w_x^2\}_{x\in L}$, such that $w_x^1, w_x^2 \in R_L(x)$ for every $x \in L$, the following ensembles are computationally indistinguishable over $x \in L$:

$$\{\mathsf{View}_{V^*}\left\langle P(w_x^1), V^*(z)\right\rangle(x)\}_{x\in L, z\in\{0,1\}^*} \approx \{\mathsf{View}_{V^*}\left\langle P(w_x^2), V^*(z)\right\rangle(x)\}_{x\in L, z\in\{0,1\}^*}$$

2.4 Universal Arguments

Universal arguments (introduced in [BG02] and closely related to CS-proofs [Mic00]) are used in order to provide "efficient" proofs to statements of the form $y = (M, x, t)$, where y is considered to be a true statement if M is a non-deterministic machine that accepts x within t steps. The corresponding language and witness relation are denoted $L_{\mathcal{U}}$ and $\mathbf{R}_{\mathcal{U}}$ respectively, where the pair $((M, x, t), w)$ is in $\mathbf{R}_{\mathcal{U}}$ if M (viewed here as a two-input deterministic machine) accepts the pair (x, w) within t steps. Notice that every language in NP is linear time reducible to $L_{\mathcal{U}}$. Thus, a proof system for $L_{\mathcal{U}}$ allows us to handle all NP-statements. In fact, a proof system for $L_{\mathcal{U}}$ enables us to handle languages that are presumably "beyond" NP, as the language $L_{\mathcal{U}}$ is NE-complete (hence the name universal arguments).[2]

[2]Furthermore, every language in NEXP is polynomial-time (but not linear-time) reducible to $L_{\mathcal{U}}$

Definition 5 (Universal argument)**.** A pair of interactive Turing machines (P, V) is called a **universal argument** system if it satisfies the following properties:

- Efficient verification: There exists a polynomial p such that for any $y = (M, x, t)$, the total time spent by the (probabilistic) verifier strategy V, on common input y, is at most $p(|y|)$. In particular, all messages exchanged in the protocol have length smaller than $p(|y|)$.

- Completeness by a relatively efficient prover: For every $((M, x, t), w)$ in $\mathbf{R}_{\mathcal{U}}$,

$$\Pr[(P(w), V)(M, x, t) = 1] = 1$$

  Furthermore, there exists a polynomial q such that the total time spent by $P(w)$, on common input (M, x, t), is at most $q(T_M(x, w)) \leq q(t)$, where $T_M(x, w)$ denotes the running time of M on input (x, w).

- Computational Soundness: For every polynomial size circuit family $\{P_n^*\}_{n \in N}$, and every triplet $(M, x, t) \in \{0, 1\}^n \setminus L_{\mathcal{U}}$,

$$\Pr[(P_n^*, V)(M, x, t) = 1] < \nu(n)$$

  where $\nu(\cdot)$ is a negligible function.

- Weak proof of knowledge: For every positive polynomial p there exists a positive polynomial p' and a probabilistic polynomial-time oracle machine E such that the following holds: for every polynomial-size circuit family $\{P_n^*\}_{n \in N}$, and every sufficiently long $y = (M, x, t) \in \{0, 1\}^*$ if $\Pr[(P_n^*, V)(y) = 1] > 1/p(|y|)$ then

$$\Pr[\exists w = w_1, \ldots w_t \in \mathbf{R}_{\mathcal{U}}(y) \;\; s.t. \;\; \forall i \in [t], \; E_r^{P_n^*}(y, i) = w_i] > \frac{1}{p'(|y|)}$$

  where $\mathbf{R}_{\mathcal{U}}(y) \stackrel{\text{def}}{=} \{w : (y, w) \in \mathbf{R}_{\mathcal{U}}\}$ and $E_r^{P_n^*}(\cdot, \cdot)$ denotes the function defined by fixing the random-tape of E to equal r, and providing the resulting E_r with oracle access to P_n^*.

2.5 Commitment Schemes

Commitment protocols allow a *sender* to commit itself to a value while keeping it secret from the *receiver*; this property is called **hiding**. At a later time, the commitment can only be opened to a single value as determined during the commitment protocol; this property is called **binding**. Commitment schemes come in two different flavors, statistically binding and statistically hiding; we only make use of statistically binding commitments in this paper. Below we sketch the properties of a statistically binding commitment; full definitions can be found in [Gol01].

In statistically binding commitments, the binding property holds against unbounded adversaries, while the hiding property only holds against computationally bounded (non-uniform) adversaries. The statistical-binding property asserts that, with overwhelming probability over the randomness of the receiver, the transcript of the interaction fully determines the value committed to by the sender. The computational-hiding property guarantees that the commitments to any two different values are computationally indistinguishable.

Non-interactive statistically-binding commitment schemes can be constructed using any one-to-one one-way function (see Section 4.4.1 of [Gol01]). Allowing some minimal interaction (in which the receiver first sends a single random initialization message), statistically-binding commitment schemes can be obtained from any one-way function [Nao91, HILL99].

3 A Non-Black-Box Public-Coin Parallel ZK Argument

3.1 The Protocol

Our non-black-box public-coin parallel zero-knowledge argument is similar to the non-black-box public-coin bounded-concurrent zero-knowledge argument of Barak [Bar01]. Our argument ParallelZKArg is described in Fig. 1 which utilizes an additional relation $\mathbf{R}_S$ defined in Fig. 2. ParallelZKArg is constant round, and can be based on collision resistant hash-functions. Intuitively, the zero-knowledge simulator will use a witness of the relation $\mathbf{R}_S$ as a trapdoor.

In our construction, $\boldsymbol{c}$ denotes a vector, and $\boldsymbol{c}_{-i}$ denotes the same vector with the i^{th} element removed (i.e., $\boldsymbol{c}_{-1}$ is one shorter than $\boldsymbol{c}$); $\{\mathcal{H}_n\}_n$ denotes a family of collision resistant hash-functions indexed by integer n; $\{f_s\}_s$ denotes a family of pseudorandom functions indexed by $s \in \{0,1\}^*$; and, $\mathsf{Com}(x;r)$ denotes a statistically binding commitment of x using randomness r. We also make use of a witness-indistinguishable universal argument of knowledge, WI UARG [BG02], because the relation $\mathbf{R}_S$ is quasi-polynomial time ($n^{\log n}$) instead of polynomial time.

Protocol ParallelZKArg

Common Input: An instance x of a language $L \in \mathsf{NP}$ with witness relation $\mathbf{R}_L$.

Stage One:

$V \to P$: Send $h \leftarrow \mathcal{H}_n$.

$P \to V$: Send $c = \mathsf{Com}(0^n)$.

$V \to P$: Send $r \stackrel{R}{\leftarrow} \{0,1\}^{3n}$.

Stage Two:

$P \Leftrightarrow V$: A WI UARG $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}} \rangle$ proving the OR of the following statements:

1. $\exists\ w \in \{0,1\}^{\text{poly}(|x|)}$ s.t. $\mathbf{R}_L(x,w) = 1$.
2. $\exists\ \langle \Pi, m, i, s, \boldsymbol{h}, \boldsymbol{c} \rangle$ s.t. $\mathbf{R}_S(\langle h, c, r \rangle, \langle \Pi, m, i, s, \boldsymbol{h}, \boldsymbol{c} \rangle) = 1$.

Figure 1: A public-coin non-black-box parallel zero-knowledge protocol.

Simplifying Assumptions. We remark that the relation presented in Fig. 2 is slightly over-simplified and only works when $\{\mathcal{H}_n\}_n$ is collision resistant against "slightly" super-polynomially sized circuits. To make it work assuming collision resistance against polynomially sized circuits, one should use a "good" error-correcting code $\mathtt{ECC}$ (i.e., with constant distance and with polynomial-time encoding and decoding), and replace the commitments $\mathsf{Com}(m \| h_j(\boldsymbol{h}_{-j}) \| h_j(\Pi); f_s(j))$ with $\mathsf{Com}(m \| h_j(\mathtt{ECC}(\boldsymbol{h}_{-j})) \| h_j(\mathtt{ECC}(\Pi)); f_s(j))$ [BG02]. We also assume that Com is a one-message commitment scheme. Such schemes can be constructed based on any one-to-one one-way function. At the cost of a small complication, the one-message scheme could have been replaced by the 2-message commitment scheme of [Nao91], which can be based on any one-way function [HILL99].

[3] In the description of $\mathbf{R}_S$, when we require $c_j = \mathsf{Com}(m \| h_j(\boldsymbol{h}_{-j}) \| h_j(\Pi); f_s(j))$, it is possible to replace $h_j(\boldsymbol{h}_{-j})$ with $h_j(\boldsymbol{h})$ (i.e., hashing the description of all the hash functions, including h_j). For the sake of minimality, we choose to keep $h_j(\boldsymbol{h}_{-j})$ instead.

Instance: A triplet $\langle h, c, r\rangle \in \mathcal{H}_n \times \{0,1\}^{\mathrm{poly}(n)} \times \{0,1\}^{3n}$.

Witness: $\langle \Pi, m, i, s, \boldsymbol{h}, \boldsymbol{c}\rangle$: A program $\Pi \in \{0,1\}^*$, an integer m, an index $i \in [m]$, a seed s, a m-vector of hash functions $\boldsymbol{h} = (h_1, \cdots, h_m) \in \mathcal{H}_n^m$, and a m-vector of commitments $\boldsymbol{c} = (c_1, \cdots, c_m)$.

Relation: $\mathbf{R}_S(\langle h, c, r\rangle, \langle \Pi, m, i, s, \boldsymbol{c}, \boldsymbol{h}\rangle) = 1$ if and only if:

1. $i \in [m]$, $|s| \leq n$
2. $\Pi(\boldsymbol{c}) = \boldsymbol{r} = (r_1, \ldots, r_m) \in \{0,1\}^{mn}$ within $n^{\log n}$ steps, and $r_i = r$.
3. $h = h_i$, $c = c_i$.
4. For $j \in [m]$ we have $c_j = \mathsf{Com}(m \| h_j(\boldsymbol{h}_{-j}) \| h_j(\Pi); f_s(j))$.[3]

Figure 2: $\mathbf{R}_S$, an NP relation that extend Barak's construction [Bar01] for parallel repetitions.

3.2 Completeness and Soundness

ParallelZKArg is clearly complete; an honest prover can use the real witness to complete the Stage Two argument. The soundness of ParallelZKArg follows from the same ideas as Barak's non-black-box zero-knowledge protocol.

Lemma 2. ParallelZKArg *has negligible soundness error against polynomial-time bounded provers.*

Proof. The main idea is that any deterministic program Π produces only one output for any given input. When the prover commits to c, it can no longer change the program Π, m or $\boldsymbol{h}$ because Com is statistically binding and $\mathcal{H}_n$ is collision resistant. Since we use a pseudo-random function to determine the randomness of the commitments, the value of $\Pi(\boldsymbol{c})$ would only depend on i and s. Since there are $m = \mathrm{poly}(n)$ values of i and 2^n values of s, $\Pi(\boldsymbol{c})$ may take on at most $\mathrm{poly}(n)2^n$ values after the commitment c is fixed. On the other hand, the verifier chooses $r \in \{0,1\}^{3n}$ randomly after the prover fixes c. Therefore, the probability that there exists some i and s so that the i^{th} component of $\Pi(\boldsymbol{c})$ is r is less than $\mathrm{poly}(n)2^n/2^{3n} < 2^{-n}$ and is negligible.

We now prove the lemma formally. Suppose the contrary that some efficient cheating prover P^* breaks soundness of ParallelZKArg with polynomial probability on an infinite sequence of inputs $\{x_n\}_n$, $x_n \in \{0,1\}^n \setminus L$. Using P^*, we construct an adversary $\mathcal{A}$ that acts either as a collision finder for $\mathcal{H}_n$, or a cheating committer for Com.

$\mathcal{A}$ runs P^* internally. On input 1^n, $\mathcal{A}$ starts by receiving a random $h \leftarrow \mathcal{H}_n$, and presents h to P^* as the first message of ParallelZKArg; P^* responds by generating a commitment c. The following step is then repeated twice: $\mathcal{A}$ sends a random challenge $r \in \{0,1\}^{3n}$ to P^*, and uses the witness extractor of the Stage Two UARG on P^* to extract a (potentially quasi-polynomial-length) witness of the relation $\mathbf{R}_S$ (we cannot extract a witness $w \in \mathbf{R}_L(x)$ since $x \notin L$). Because P^* breaks soundness with noticeable probability, $\mathcal{A}$ succeeds in extracting two witnesses, $\langle \Pi, m, i, s, \boldsymbol{c}, \boldsymbol{h}\rangle$ and $\langle \Pi', m', i', s', \boldsymbol{c}', \boldsymbol{h}'\rangle$, to the statements $\langle h, c, r\rangle$ and $\langle h, c, r'\rangle$ (where r and r' are independent and uniform in $\{0,1\}^{3n}$), also with noticeable probability. We split into two cases:

Case 1: $\Pi \neq \Pi'$ or $m \neq m'$ or $\boldsymbol{h}_{-i} \neq \boldsymbol{h}'_{-i'}$.

By the definition of $\mathbf{R}_S$, we have

$$\begin{aligned} c = c_i &= \mathsf{Com}(m \| h(\boldsymbol{h}_{-i}) \| h(\Pi); f_s(i)) \\ &= c'_{i'} = \mathsf{Com}(m' \| h(\boldsymbol{h}'_{-i'}) \| h(\Pi'); f_{s'}(i')) \end{aligned}$$

If $\Pi \neq \Pi'$, then either $\mathcal{A}$ has found a collision to h (if $h(\Pi) = h(\Pi')$), or $\mathcal{A}$ has broken the binding property of Com (by decommiting c to two different strings, $\cdot \| h(\Pi)$ and $\cdot \| h(\Pi')$). These conclusions hold in the same manner if $m \neq m'$ or $\boldsymbol{h}_{-i} \neq \boldsymbol{h}'_{-i'}$.

Case 2: $\Pi = \Pi'$ and $m = m'$ and $\boldsymbol{h}_{-i} = \boldsymbol{h}'_{-i'}$.
In this case, given the first witness $\langle \Pi, m, i, s, \boldsymbol{c}, \boldsymbol{h} \rangle$, the value of $\Pi'(\boldsymbol{c}')_{i'}$ is fixed modulo the value of $i' \in [m]$ and $s' \in \{0,1\}^n$; that is, $\Pi'(\boldsymbol{c}')_{i'}$ can take on at most $m \cdot 2^n$ values. However, $r' \in \{0,1\}^{3n}$ is chosen independently from the first witness. Therefore $\Pi'(\boldsymbol{c}')_{i'} = r'$ (required for this case to occur) is possible with probability at most $m \cdot 2^n / 2^{3n} < 2^{-n}$.

Therefore we conclude that except with exponentially small probability (when Case 2 occurs), $\mathcal{A}$ either finds a collision for h, or breaks the binding property of Com. This gives the desired contradiction. □

3.3 Zero Knowledge

Given a m-session parallel adversarial verifier V^*, we construct a (non-black-box) zero-knowledge simulator $S = S_{V^*}$ as follows. On input x and auxiliary input z, $S = S_{V^*}(x, z)$ first picks a random $s \in \{0,1\}^n$ to fix the pseudorandom function f_s, and starts a straight-line simulation of V^*'s view. During the simulation, V^* starts by opening m sessions of ParallelZKArg and sending hash functions h_i in session i. S is expected to respond in session i with some commitment c_i, after which V^* responds with a string r_i. The crux of the simulation is for S to commit to a program Π such that $\Pi(\boldsymbol{c}) = \boldsymbol{r}$ (including the case where some components of $\boldsymbol{r}$ are aborts). But V^* is just such a program. Therefore, S sets Π to $V^*(x, z)$, and sets $c_i = \mathsf{Com}(m \| h_i(\boldsymbol{h}_{-i}) \| h_i(\Pi); f_s(i))$. By construction, we now have for each session i, $(\langle h_i, c_i, r_i \rangle, \langle \Pi, m, i, s, \boldsymbol{c}, \boldsymbol{h} \rangle) \in \mathbf{R}_S$; this is because the messages $\boldsymbol{r} = (r_1, \ldots, r_m)$ are indeed what V^* outputs given $\boldsymbol{c}$. These witnesses allow S to complete the Stage Two arguments. After the simulation ends, S outputs the view of V^* during the simulation.

Clearly S runs in polynomial time. The following lemma establishes zero-knowledge.

Lemma 3. *The following ensembles are computationally indistinguishable over $n \in \mathbb{N}$:*

$$\{S_{V^*}(x, z)\}_{x \in L, z \in \{0,1\}^*} \approx \{\mathsf{View}_{V^*} \langle P, V^*(z) \rangle (x)\}_{x \in L, z \in \{0,1\}^*}$$

Proof. We use a simple hybrid argument.

Let S_1 be a simulator that is given a witness w for $x \in L$. S_1 proceeds as S but instead uses w to complete the Stage Two argument. By the witness-indistinguishable property of the Stage Two argument (and recall that witness-indistinguishability is preserved under parallel repetition [FS90]), the output of $S(x)$ and $S_1(x)$ are indistinguishable.

Let S_2 be the same simulator as S_1 except that S_2 uses fresh randomness for the commitments $c_1, \ldots, c_m$. Since S_1 chooses s uniformly, f_s is a pseudorandom function, and both S_1 and S_2 are efficient, the output of $S_1(x)$ and $S_2(x)$ are indistinguishable. (Formally, this step can be further

split into m hybrids; the i^{th} hybrid uses fresh randomness in the first i parallel sessions, and uses pseudorandomness in the other sessions.)

Let S_3 be the same simulator as S_2 except that S_3 commits to 0^n in the commitments $c_1, \ldots, c_m$. By the computational-hiding property of Com, the output of $S_2(x)$ and $S_3(x)$ are indistinguishable. (Formally, this step can be further split into m hybrids; the i^{th} hybrid commits to 0^n in the first i parallel sessions, and commits following the strategy of S in the other sessions.)

But S_3 is identical to the honest prover, i.e., the output of $S_3(x)$ is identical to the view of V^*. Therefore, we have shown that the output of S is computationally indistinguishable from the view of V^*. □

Remark. Our simulation technique breaks down in the case of concurrent zero-knowledge, where the adversarial verifier V^* is allowed to arbitrarily schedule and intertwine messages among multiple sessions. In particular, after the prover sends its first message c for some session i, if V^* nests messages from other sessions, and uses these additional messages to generate its response r in session i, then it is no longer the case that V^* on input just $\boldsymbol{c}$ would produce $\boldsymbol{r}$.

References

[Bar01] Boaz Barak. How to go beyond the black-box simulation barrier. In *FOCS '01*, pages 106–115, 2001.

[BG02] Boaz Barak and Oded Goldreich. Universal arguments and their applications. In *Computational Complexity*, pages 162–171, 2002.

[Blu86] M. Blum. How to prove a theorem so no one else can claim it. *Proc. of the International Congress of Mathematicians*, pages 1444–1451, 1986.

[DNS04] Cynthia Dwork, Moni Naor, and Amit Sahai. Concurrent zero-knowledge. *J. ACM*, 51(6):851–898, 2004.

[FS90] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *STOC '90*, pages 416–426, 1990.

[GK96a] Oded Goldreich and Ariel Kahan. How to construct constant-round zero-knowledge proof systems for NP. *Journal of Cryptology*, 9(3):167–190, 1996.

[GK96b] Oded Goldreich and Hugo Krawczyk. On the composition of zero-knowledge proof systems. *SIAM Journal on Computing*, 25(1):169–192, 1996.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems. *J. ACM*, 38(3):691–729, 1991.

[GO94] Oded Goldreich and Yair Oren. Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology*, 7:1–32, 1994.

[Gol01] Oded Goldreich. *Foundations of Cryptography — Basic Tools*. Cambridge University Press, 2001.

[Gol02] Oded Goldreich. Concurrent zero-knowledge with timing, revisited. In *STOC '02*, pages 332–340, 2002.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28:12–24, 1999.

[KP01] Joe Kilian and Erez Petrank. Concurrent and resettable zero-knowledge in poly-loalgorithm rounds. In *STOC '01*, pages 560–569, 2001.

[Mic00] Silvio Micali. Computationally sound proofs. *SIAM Journal on Computing*, 30(4):1253–1298, 2000.

[Nao91] Moni Naor. Bit commitment using pseudorandomness. *Journal of Cryptology*, 4(2):151–158, 1991.

[PR08] Rafael Pass and Alon Rosen. New and improved constructions of nonmalleable cryptographic protocols. *SIAM Journal on Computing*, 38(2):702–752, 2008.

[PRS02] Manoj Prabhakaran, Alon Rosen, and Amit Sahai. Concurrent zero knowledge with logarithmic round-complexity. In *FOCS '02*, pages 366–375, 2002.

[PTW09] Rafael Pass, Wei-Lung Dustin Tseng, and Douglas Wikström. On the composition of public-coin zero-knowledge protocols. In *CRYPTO '09*, pages 160–176, 2009.

[RK99] Ransom Richardson and Joe Kilian. On the concurrent composition of zero-knowledge proofs. In *Eurocrypt '99*, pages 415–432, 1999.