Efficient NIZK Arguments with Straight-Line Simulation and Extraction*

Michele Ciampi[1][0000−0001−5062−0388] and Ivan Visconti[2][0000−0003−2381−5846]

[1] The University of Edinburgh, Edinburgh, UK, michele.ciampi@ed.ac.uk
[2] University of Salerno, Salerno, Italy, visconti@unisa.it

Abstract. Non-interactive zero-knowledge (NIZK) arguments allow a prover to convince a verifier about the truthfulness of an $\mathcal{NP}$-statement by sending just one message, without disclosing any additional information. In several practical scenarios, the Fiat-Shamir transform is used to convert an *efficient* constant-round public-coin honest-verifier zero-knowledge proof system into an efficient NIZK argument system. This approach is provably secure in the random oracle model, crucially requires the programmability of the random oracle and extraction works through rewinds. The works of Lindell [TCC 2015] and Ciampi et al. [TCC 2016] proposed efficient NIZK arguments with non-programmable random oracles along with a programmable common reference string.
In this work we show an efficient NIZK argument with straight-line simulation and extraction that relies on features that alone are insufficient to construct NIZK arguments (regardless of efficiency). More specifically we consider the notion of quasi-polynomial time simulation proposed by Pass in [EUROCRYPT 2003] and combine it with simulation and extraction with non-programmable random oracles thus obtaining a NIZK argument of knowledge where neither the zero-knowledge simulator, nor the argument of knowledge extractor needs to program the random oracle. Still, both the simulator and the extractor are straight-line. Our construction uses as a building block a modification of the Fischlin's transform [CRYPTO 2005] and combines it with the concept of dense puzzles introduced by Baldimtsi et al. [ASIACRYPT 2016]. We also argue that our NIZK argument system inherits the efficiency features of Fischlin's transform, which represents the main advantage of Fischlin's protocol over existing schemes.

1 Introduction

A proof system allows an entity, called prover, to convince another entity, called verifier, about the truthfulness of a claim. Informally, a proof[3] system is *zero-knowledge* (ZK) [GMR89] if the prover holds a secret that is required to successfully convince the verifier and moreover the proof does not disclose any information about the secret. In the *non-interactive* scenario only the prover can speak and sends just one message. This kind of proofs, introduced in [BFM88], are called *Non-Interactive Zero-Knowledge (NIZK)* proofs. Since it is impossible to construct a NIZK proof for non-trivial languages without setup assumptions, Blum et al. [BFM88] proposed the *Common Reference String* (CRS) model. In the CRS model there exists a trusted string (the exact shape of the CRS depends on the specific NIZK proof instantiation) that is given as input to both the prover and the verifier.

In [FLS90] the authors show NIZK proofs in the CRS model for any $\mathcal{NP}$-language in a setting where the same CRS can be reused to generate multiple proofs. Even though NIZK proofs exist for all $\mathcal{NP}$, the candidate constructions are rather inefficient due to the $\mathcal{NP}$-reduction that needs to be performed before computing the actual NIZK proof. One of the most used approaches to obtain efficient NIZK proofs consists in starting with an efficient interactive constant-round public-coin honest-verifier zero-knowledge (HVZK) proof system and making it non-interactive by replacing the role of the verifier with a hash function modelled as a random oracle [BR93] (RO). In particular, the hash function takes as input the transcript computed so far and returns the message on the behalf of a verifier. This approach is the so-called *Fiat-Shamir (FS)*

* Research partly supported by H2020 project PRIVILEDGE #780477.

[3] When discussing informally we will use the word proof to refer to both unconditionally and computationally sound proofs. Only in the more formal part of the paper we will make a distinction between arguments and proofs.

transform [FS86]. To prove the security of such a transform, the ZK simulator needs to program the RO (i.e., the simulator decides how the RO answers the queries). Similarly, the argument of knowledge property requires an extractor that programs the random oracle. Exploiting the programmability of the random oracle makes it difficult to prove the composability of the argument system in a setting where multiple instantiations of it are run in concurrency. The work of Canetti et al. [CJS14], for example, considers the natural scenario where multiple instantiations of different cryptographic protocols use the same random oracle without programming it. In general, in order to avoid such issues it is preferrable to avoid simulators that need to program the random oracle.

Lindell in [Lin15] provides a NIZK argument that can be proven secure assuming the existence of a non-programmable random-oracle (NPRO) and a programmable CRS. In more detail, the ZK of Lindell's protocol is proved without relying on the RO at all (though, the CRS needs to be programmed), and the soundness is proved *without* programming the RO. In a follow-up work [CPSV16], the authors improve the construction of Lindell in terms of efficiency and generality under the same setup considered in [Lin15]. A different approach towards round-efficient ZK arguments consists in allowing the simulator to run in quasi-polynomial time instead of expected polynomial time [Pas03b]. This notion implies that a malicious verifier can learn from the prover anything that can be learned by a quasi-polynomial time algorithm. As observed in [BP04], the simulator is usually not run by parties running the NIZK argument, and thus quasi-polynomial simulation can still be useful in various applications. In [Pas04b] it is shown that two rounds are necessary and sufficient for quasi-polynomial time simulatable arguments. Therefore, even though the notion of ZK with quasi-polynomial time simulation allows to overcome some of the impossibility results of standard ZK, the impossibility of constructing NIZK arguments holds also in this less demanding model.

Given the impossibility shown in [Pas04b] of obtaining NIZK arguments with quasi-polynomial simulation, and the obvious impossibility of obtaining NIZK arguments without setup assumptions in the non-programmable RO (NPRO) model, we focus on the following relevant question:

> *Is it possible to construct an efficient NIZK argument of knowledge with quasi-polynomial time simulation where neither the zero-knowledge simulator nor the argument of knowledge extractor needs to program the random oracle?*

In this work we answer affirmatively to this question assuming that dense cryptographic puzzles exist. In more detail, our protocol is proved to enjoy *perfect* concurrent zero knowledge (ZK) via quasi-polynomial simulation and is moreover an argument of knowledge (AoK) with online extraction (i.e., the extraction process does not rewind the prover). Interestingly, even though we prove the ZK property using quasi-polynomial simulation, the security of our NIZK argument does not rely on any complexity-leveraging-type assumptions (i.e., we do not require hardness assumptions to hold against superpolynomial-time adversaries).

Our techniques. We start with the work of Fischlin [Fis05] that presents a NIZK AoK with an online extractor starting with a Σ-protocol (i.e., a 3-round public-coin proof system enjoying a special notion of soundness and a special notion of HVZK). In more detail, Fischlin's protocol is proved ZK by exploiting the programmability of the random oracle (RO), whereas the soundness is proved by relying just on the *observability* of the RO (i.e., the online extractor has only access to the queries made by the adversary to the RO). The aim of our work is to circumvent the need to program the RO by using quasi-polynomial time simulation (i.e., we allow the ZK simulator to run in quasi-polynomial time).

We present our construction in an incremental way. We first show that the modified version of Fischlin's protocol proposed in [Kas22] is witness indistinguishable (WI) with an on-line extractor that does not need to program the RO, therefore relying on a non-programmable RO (NPRO). Then we use it as the main building block together with dense cryptographic puzzles. Roughly speaking, a cryptographic puzzle is defined together with a hardness parameter g. So, if a randomly sampled puzzle is given to an adversary then she should not be able to find a solution with non-negligible probability in less than a number of steps that is function of g. We consider the notion of puzzle system proposed in [BKZZ16], where the hardness of a puzzle holds as long as the puzzle is sampled from a uniformly random distribution. Baldimtsi et al. [BKZZ16] also show how to instantiate such a puzzle from standard number-theoretic assumptions (e.g., the discrete logarithm (DL) problem) and from NPROs. In a nutshell, our NIZK argument combines a dense puzzle system PuzSys with

a non-interactive WI argument of knowledge $\Pi^{\mathcal{WI}}$ as follows. The prover queries the random oracle with the statement x that he wants to prove, thus obtaining a puzzle $\mathtt{puz}$. Then the prover computes a non-interactive WI (NIWI) proof where he proves knowledge of either the witness corresponding to the instance x or the solution of the puzzle. We observe that a malicious prover could fool the verifier by finding a solution to the puzzle and using it as a witness for the WI proof. To avoid this we just need to carefully choose the hardness factor of the puzzle in such a way that a malicious probabilistic polynomial-time prover cannot solve it, but a quasi-polynomial time simulator can. As discussed above, one of the main building blocks of our construction is $\Pi^{\mathcal{WI}}$. More precisely, $\Pi^{\mathcal{WI}}$ is an argument of knowledge with an online extractor in which the WI property holds against *all-powerful* adversaries. Equipped with this tool we can prove the security of our NIZK argument of knowledge without using complexity leveraging arguments. We show how to obtain $\Pi^{\mathcal{WI}}$ starting from any Σ-protocol Π, this yields to the following theorem.

Theorem (informal). *Let Π be a Σ-protocol for the $\mathcal{NP}$ relation* Rel*. Assuming the hardness of the discrete logarithm problem then there exists an efficient NIZK AoK with online extraction and straight-line (perfect) quasi-polynomial time simulation for* Rel *where neither the simulator nor the AoK extractor needs to program the RO.*

We stress that our construction has a ZK straight-line simulator and an online AoK simulator. This yields a protocol that can be more easily composed concurrently with other cryptographic protocols. Indeed, in follow-up work [BCC+24] the authors show how to extend our approach to the UC setting, thus obtaining composable security in the non-programmable (global) random oracle, assuming no additional setup. Moreover, the construction that we propose is almost as efficient as Fischlin's construction and can be instantiated from a large class of Σ-protocols (even larger than the class considered in [Fis05]) and cryptographic puzzles. Indeed, in [Fis05], in order to prove the zero-knowledge property it is required that the first round of the underlying Σ-protocol has min-entropy that is superlogarithmic in the security parameter. In our approach, we do not need to rely on this additional requirement.

Related work. As observed by Fischlin in [Fis05], in the prior work of Pass [Pas03a] the author showed a NIZK AoK where the argument of knowledge extractor does not need to rewind the adversary. These are exactly the same security properties offered by the protocol of [Fis05]. Moreover, in [Pas03a] the author shows that the WI property can be obtained without the RO simulator programming RO. The main improvement of Fischlin's construction (and ours) over Pass' protocol is in the size of the proof. Indeed, in [Fis05] the author argues that even considering a modified[4] version Pass' protocol, for reasonable parameters the size of a single proof, requires between 10000 and 25000 bits. Therefore, with the goal of obtaining shorter proofs, in our work we study a variant of Fischlin's construction, instead of simply taking out from the box the NIWI protocol of [Pas03a]. For more details on the efficiency of our protocol, we refer the reader to Sec. 5.1.

We have already mentioned the work of Kondi et al. [Kas22] and Badertscher et al. [BCC+24]. In [Kas22], the authors propose a modified version of Fischlin's construction (that we rely on). We extend the results of [Kas22] showing that this modified version preserves WI and online extraction with a NPRO. The work of Kondi et al. [Kas22] proposes also other results, based on Fischlin's paradigm, that improve the efficiency of Schnorr/EdDSA signature aggregation schemes. In [CL24] the authors propose an alternative approach to improve the efficiency of the Fischlin's transform. Investigating whether it is possible to adapt the efficiency improvements of [CL24,Kas22] to our scheme it is an interesting future direction. In [BCC+24] the authors show how a relaxation of the zero-knowledge functionality, allows the design of UC secure non-interactive zero-knowledge assuming as the only form of setup the non-programmable (global) random-oracle. The approach of [BCC+24], despite being significantly different, follow a similar blueprint where the prover needs to show that either an certain NP statement is true or that he knows the solution of a puzzle generated via the random oracle.

[4] The original Pass' protocol is particularly inefficient due to the number of parallel repetitions required to amplify the soundness. In [Fis05] the author considers an improved version, that uses Merkle trees to reduce the size of the proof. We refer to the introductory section of [Fis05] for more details.

2 Definitions and Tools

Preliminaries. We denote the security parameter by λ and for a finite set Q, $x \leftarrow Q$ the sampling of x from Q with uniform distribution. We use the abbreviation PPT that stays for probabilistic polynomial time. We use $\mathbb{N}$ to denote the set of natural numbers and $\mathsf{poly}(\cdot)$ to indicate a generic polynomial function. A *polynomial-time $\mathcal{NP}$-relation* Rel (or *$\mathcal{NP}$-relation*, in short) is a subset of $\{0,1\}^* \times \{0,1\}^*$ such that membership of (x,w) in Rel can be decided in time polynomial in $|x|$. For $(x,w) \in \mathsf{Rel}$, we call x the *instance* and w a *witness* for x. For a polynomial-time relation Rel, we define the $\mathcal{NP}$-language L_{Rel} as $L_{\mathsf{Rel}} = \{x | \exists\ w : (x,w) \in \mathsf{Rel}\}$. Analogously, unless otherwise specified, for an $\mathcal{NP}$-language L we denote by $\mathsf{Rel}_{\mathsf{L}}$ the corresponding polynomial-time relation (that is, $\mathsf{Rel}_{\mathsf{L}}$ is such that $L = L_{\mathsf{Rel}_{\mathsf{L}}}$). We define $\hat{L}$ to be the input language that includes both the NP-language L and all well-formed instances that do not have a witness. Let A and B be two interactive probabilistic algorithms. We denote by $\langle A(\alpha), B(\beta)\rangle(\gamma)$ the distribution of B's output after running on private input β with A using private input α, both running on common input γ. A *transcript* of $\langle A(\alpha), B(\beta)\rangle(\gamma)$ consists of the messages exchanged during an execution where A receives a private input α, B receives a private input β and both A and B receive a common input γ. Moreover, we will refer to the *view* of A (resp. B) as the messages it received during the execution of $\langle A(\alpha), B(\beta)\rangle(\gamma)$, along with its randomness and its input. A function $\nu(\cdot)$ from non-negative integers to reals is called negligible, if for every constant $c > 0$ and all sufficiently large $\lambda \in \mathbb{N}$ we have $\nu(\lambda) < \lambda^{-c}$.

2.1 Argument Systems

Here we recall the notions of completeness and online extraction provided in [Fis05]. A pair $\Pi = (\mathcal{P}, \mathcal{V})$ of probabilistic polynomial-time algorithms is called a non-interactive argument of knowledge for the $\mathcal{NP}$-relation $\mathsf{Rel}_{\mathsf{L}}$ with an online extractor (in the non-programmable random oracle model) if the following holds.

Completeness. For any non-programmable random oracle $\mathcal{O}$, any $(x,w) \in \mathsf{Rel}_{\mathsf{L}}$ and any $\pi \leftarrow \mathcal{P}^{\mathcal{O}}(x,w)$ we have $\mathrm{Prob}\left[\, \mathcal{V}^{\mathcal{O}}(x,\pi) = 1 \,\right] = 1 - \nu(|x|)$, for some negligible function ν.

Argument of Knowledge with Online Extractor. There exist a probabilistic polynomial-time algorithm Ext called *AoK online extractor* such that for any PPT adversary $\mathcal{A}^{\mathcal{O}}$ there exists a negligible function ν such that the following holds: $\mathrm{Prob}\left[\, (x,w) \notin \mathsf{Rel} \text{ and } \mathcal{V}^{\mathcal{O}}(x,\pi) = 1 \,\right] \leq \nu(\lambda)$, where $(x,\pi) \leftarrow \mathcal{A}^{\mathcal{O}}(\lambda)$ (where x is the theorem and π is the proof generated by the adversary), $w \leftarrow \mathsf{Ext}(x, \pi, \mathcal{Q}_{\mathcal{O}}(\mathcal{A}))$ and $\mathcal{Q}_{\mathcal{O}}(\mathcal{A})$ represents the sequence of queries/answers for the oracle $\mathcal{O}$. Not to overburden the descriptions of protocols and simulators, we omit to specify that the parties have access to the NPRO $\mathcal{O}$ whenever it is clear from the context.

Quasi-polynomial time simulation. Since the verifier in an interactive argument is often modeled as a PPT machine, the classical zero-knowledge definition requires that the simulator runs also in (expected) polynomial time. In [Pas03b], the simulator is allowed to run in time $\lambda^{\mathtt{poly}(\log(\lambda))}$. Loosely speaking, we say that an interactive argument is $\lambda^{\mathtt{poly}(\log(\lambda))}$-perfectly simulatable if for any adversarial verifier there exists a simulator running in time $\lambda^{\mathtt{poly}(\log(\lambda))}$, where λ is the size of the statement being proved, whose output is identically distributed to the output of the adversarial verifier.

Definition 1 (straight-line $T(\lambda)$ simulatability, Def. 31 of [Pas04a]). *Let $T(\lambda)$ be a class of functions that is closed under composition with any polynomial. We say that an interactive argument $(\mathcal{P}, \mathcal{V})$ for the language $L \in \mathcal{NP}$, with the witness relation $\mathsf{Rel}_{\mathsf{L}}$, is straight-line $T(\lambda)$-simulatable if for every PPT machine $\mathcal{V}^\star$ there exists a probabilistic simulator S with running time bounded by $T(\lambda)$ such that the following two ensembles are computationally indistinguishable*

- $\{(\langle \mathcal{P}(w), \mathcal{V}^\star(z)\rangle(x))\}_{z \in \{0,1\}^*, x \in L}$ *for arbitrary w s.t. $(x,w) \in \mathsf{Rel}_{\mathsf{L}}$*
- $\{(\langle S, \mathcal{V}^\star(z)\rangle(x))\}_{z \in \{0,1\}^*, x \in L}$

The following theorem shows the power of straight-line $\lambda^{\mathtt{poly}(\log(\lambda))}$-perfect simulatability by connecting it to concurrent composition of arguments.

Theorem 1 ([Pas04a]). *If an argument system $\Pi = (\mathcal{P}, \mathcal{V})$ is straight-line $\lambda^{\mathtt{poly}(\mathtt{log}(\lambda))}$-simulatable then it is also straight-line concurrent $\lambda^{\mathtt{poly}(\mathtt{log}(\lambda))}$-simulatable.*

We also consider the notion of *perfect straight-line simulation*. This is equal to the Definition 1 with the difference that the malicious distinguisher can be unbounded instead of being PPT and that the two ensembles (the simulated execution and the real execution) are identically distributed (See Def. 30 of [Pas04a]).

2.2 Cryptographic Puzzles

In [BKZZ16] the authors introduce a new class of protocols with prover and verifier called Proof of Work or Knowledge (PoWorK). To formalize PoWorK, the authors give the notion of *puzzle system*. A puzzle system PuzSys is a tuple of algorithms $\mathsf{PuzSys} = (\mathsf{Sample}, \mathsf{Solve}, \mathsf{Verify})$ that are defined in the following way. Sample on input the security parameter 1^λ and the hardness factor h outputs a puzzle $\mathtt{puz}$; Solve on input the security parameter 1^λ, a hardness factor h and a puzzle instance $\mathtt{puz}$ outputs a potential solution $\mathtt{sol}$; Verify on input the security parameter 1^λ, a hardness factor h, a puzzle instance $\mathtt{puz}$, and a potential solution $\mathtt{sol}$ outputs 0 or 1. Moreover, while the algorithms Sample and Verify are efficient, it is difficult to compute a solution for a sampled puzzle. More precisely, a puzzle system is g-hard if no adversary can solve the puzzle in less than $g(\cdot)$ steps with more than negligible probability. The authors of [BKZZ16] propose also a stronger notion of puzzles that enjoys the property of *dense samplability*. That is, the puzzles can be sampled by just generating random strings (i.e., the puzzle instances should be dense over $\{0,1\}^{\ell(h,\lambda)}$) for a polynomial ℓ. We consider the same notion of puzzle system with dense samplability of [BKZZ16]. We remark that the notion of dense samplable puzzles considered in [BKZZ16] is equipped with an additional efficient algorithm that generates a puzzle together with its solution, but we do not need this additional requirement in our work. We denote the puzzle space as $\mathcal{PS}_\lambda$, the solution space as $\mathcal{SS}_\lambda$, and the hardness space as $\mathcal{HS}_\lambda$.

Definition 2. *A Dense Samplable Puzzle (DSP) system $\mathsf{PuzSys} = (\mathsf{Sample}, \mathsf{Solve}, \mathsf{Verify})$ enjoys the following properties, denoting with ν a negligible function.*

Completeness. *A puzzle system PuzSys is complete, if for every h in the hardness space $\mathcal{HS}_\lambda$:*

$$\mathrm{Prob}\left[\, \mathtt{puz} \leftarrow \mathsf{Sample}(1^\lambda, h), \mathtt{sol} \leftarrow \mathsf{Solve}(1^\lambda, h, \mathtt{puz}) : \mathsf{Verify}(1^\lambda, h, \mathtt{puz}, \mathtt{sol}) = 0 \,\right] \leq \nu(\lambda).$$

The number of steps that Solve takes to run is monotonically increasing in the hardness factor h and may exponentially depend on λ, while Verify and Sample run in time polynomial in λ.

g-Hardness. *Let $\mathsf{Steps}_B(\cdot)$ be the number of steps (i.e., machine/operation cycles) executed by algorithm B. We say that a puzzle system PuzSys is g-hard for some function g, if for every adversary $\mathcal{A}$ there exists a negligible function ν such that for every auxiliary tape $z \in \{0,1\}^\star$ and for every $h \in \mathcal{HS}_\lambda$ the following holds:*

$$\begin{aligned}\mathrm{Prob}[\mathtt{puz} \leftarrow \mathsf{Sample}(1^\lambda, h), \mathtt{sol} \leftarrow \mathcal{A}(1^\lambda, z, \mathtt{puz}) : \mathsf{Verify}(1^\lambda, h, \mathtt{puz}, \mathtt{sol}) = 1 \wedge \\ \mathsf{Steps}_{\mathcal{A}}(1^\lambda, z, h, \mathtt{puz}) \leq g(\mathsf{Steps}_{\mathsf{Solve}}(1^\lambda, h, \mathtt{puz}))] \leq \nu(\lambda).\end{aligned}$$

Dense Puzzles. *Given $\lambda, h \in \mathbb{Z}^+$ and a polynomial function ℓ, there exists a negligible function ν such that $\Delta[\mathsf{Sample}(1^\lambda, h), \mathsf{U}_{\ell(\lambda,h)})] \leq \nu(\lambda)$ where $\mathsf{U}_{\ell(\lambda,h)}$ stands for the uniform distribution over $\{0,1\}^{\ell(\lambda,h)}$.*

We observe that the properties of density and g-hardness imply that for every adversary $\mathcal{A}$, there exists a negligible function ν such that for every auxiliary tape $z \in \{0,1\}^\star$ and for every $h \in \mathcal{HS}_\lambda$ the following holds:

$$\begin{aligned}\mathrm{Prob}[\mathtt{sol} \leftarrow \mathcal{A}(1^\lambda, z, \eta) : \eta \leftarrow \{0,1\}^{\ell(\lambda,h)} \ \wedge \ \mathsf{Verify}(1^\lambda, h, \eta, \mathtt{sol}) = 1 \wedge \\ \mathsf{Steps}_{\mathcal{A}}(1^\lambda, z, h, \eta) \leq g(\mathsf{Steps}_{\mathsf{Solve}}(1^\lambda, h, \eta))] \leq \nu(\lambda).\end{aligned}$$

Puzzles from the DL assumption In [BKZZ16,BKZZ15] the authors show how to construct puzzles assuming the hardness of the discrete logarithm (DL) problem. In particular, at the end of [BKZZ15, pag. 37] the authors argue that it is possible to obtain a puzzle by randomly sampling an instance of the DL problem. The solution to this puzzle is simply the DL of the instance. This puzzle moreover has the following properties. For every hardness factor $h \in \mathcal{HS}_\lambda$ there exists a negligible function ν such:

1. $\text{Prob}\left[\ \mathsf{puz} \leftarrow \mathsf{Sample}(1^\lambda, h) : g(\mathsf{Steps}_{\mathsf{Solve}}(1^\lambda, h, \mathsf{puz})) \leq \lambda^{\log \lambda}\ \right] \leq \nu(\lambda)$;
2. the worst-case running time of $\mathsf{Solve}(1^\lambda, h, \cdot)$ is $\lambda^{\mathsf{poly}(\log \lambda)}$.

For our NIZK argument of knowledge we need a puzzle that has exactly these hardness parameters. The nice feature of the construction based on DL proposed in [BKZZ15, pag. 37] is that it admits an efficient Σ-protocol. That is, a prover can prove the knowledge of the solution of a puzzle y by simply running the Schnorr protocol. When providing an efficiency analysis of our protocol we will use this puzzle instantiation.

Witness indistinguishability. To formalize the notion of WI we consider a game $\mathsf{ExpAWI}^b_{\Pi,\mathcal{A}}$ between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ in which the instance x and two witnesses w_0 and w_1 for x are chosen by $\mathcal{A}$. The challenger, upon receiving (x, w_0, w_1) starts interacting with $\mathcal{A}$ accordingly to the prover procedure of Π using w_b as a witness. The adversary wins the game if she can guess which of the two witnesses was used by the challenger. We now formally define the WI experiment $\mathsf{ExpAWI}^b_{\Pi,\mathcal{A}}(\lambda, \zeta)$ that is parameterized by a protocol $\Pi = (\mathcal{P}, \mathcal{V})$ for an $\mathcal{NP}$-relation Rel and by a PPT adversary $\mathcal{A}$. We denote with $\mathcal{A}^b_{\mathsf{ExpAWI}}$ the view of $\mathcal{A}$ in the experiment $\mathsf{ExpAWI}^b_{\Pi,\mathcal{A}}(\lambda, \zeta)$. The experiment has as input the security parameter λ and auxiliary information ζ for $\mathcal{A}$.

$\mathsf{ExpAWI}^b_{\Pi,\mathcal{A}}(\lambda, \zeta)$:
1. $\mathcal{A}$ picks an instance x, witnesses w_0 and w_1 such that $(x, w_0), (x, w_1) \in \mathsf{Rel}$, and sends (x, w_0, w_1) to a challenger $\mathcal{C}$.
2. $\mathcal{C}$ interacts with $\mathcal{A}$ as $\mathcal{P}$ would do using the witness w_b.

Definition 3 (Witness Indistinguishability). *An argument system Π is* WI *if for every distinguisher $\mathcal{D}$, there exists a negligible function ν such that for any $\zeta \in \{0,1\}^*$ it holds that*
$$\left|\text{Prob}\left[\ \mathcal{D}(\lambda, \zeta, \mathcal{A}^0_{\mathsf{ExpAWI}}) = 1\ \right] - \text{Prob}\left[\ \mathcal{D}(\lambda, \zeta, \mathcal{A}^1_{\mathsf{ExpAWI}}) = 1\ \right]\right| \leq \nu(\lambda).$$

We also consider the notion of *perfect* WI where $\mathcal{A}$ it is not restricted to be PPT and $\nu(\lambda) = 0$.

Σ-Protocols A Σ-protocol $\Pi = (\mathcal{P}, \mathcal{V})$ is a 3-round public-coin protocol. An execution of Π proceeds with the following 3 moves:

1. $\mathcal{P}$ computes the first message using as input the instance to be proved x with the corresponding witness w, and outputs the first message a with an auxiliary information aux (we denote this action with $(a, \mathsf{aux}) \leftarrow \mathcal{P}(x, w)$).
2. $\mathcal{V}$ upon receiving a, computes and sends a random $c \leftarrow \{0,1\}^l$ with $l \in \mathbb{N}$.
3. $\mathcal{P}$ on input c and aux computes and sends z to $\mathcal{V}$ (we denote this action with $z \leftarrow \mathcal{P}(\mathsf{aux}, c)$).
4. $\mathcal{V}$, on input (x, a, c, z) outputs 1 to accept, 0 to reject (we denote this action with $\mathcal{V}(x, a, c, z) = b$ where $b \in \{0,1\}$ denotes whether $\mathcal{V}$ accepts or not).

Definition 4 (Σ-protocols [CDS94]). *A 3-move protocol Π with challenge length $l \in \mathbb{N}$ is a* Σ-protocol *for a relation* Rel *if it enjoys the following properties:*

1. ***Completeness.*** *If $(x, w) \in \mathsf{Rel}$ then all honest 3-move transcripts for (x, w) are accepting.*
2. ***Special Soundness.*** *There exists an efficient algorithm* $\mathsf{Extract}$ *that, on input two accepting transcripts for x (a, c, z) and (a, c', z') with $c' \neq c$ outputs a witness w such that $(x, w) \in \mathsf{Rel}$.*
3. ***Special Honest-Verifier Zero Knowledge (SHVZK).*** *There exists a* PPT *simulator algorithm* Sim *that takes as input $x \in L_{\mathsf{Rel}}$, security parameter 1^λ and $c \in \{0,1\}^l$ and outputs an accepting transcript for x where c is the challenge (we denote this action with $(a, z) \leftarrow \mathsf{Sim}(x, c)$). Moreover, for all l-bit strings c, the distribution of the output of the simulator on input (x, c) is identical to the distribution of the 3-move honest transcript obtained when $\mathcal{V}$ sends c as challenge and $\mathcal{P}$ runs on common input x and any private input w such that $(x, w) \in \mathsf{Rel}$.*

Following [Fis05] we also define the notion of Σ-protocols with *quasi-unique third round*. This notion requires that it should be infeasible to find another valid third round to an accepting proof (a, c, z), even if one knows the witness. As noted in [Fis05], this property holds for example if the third round z is uniquely determined by x, a, and c as for the protocols by Guillou-Quisquater [GQ88] and Schnorr [Sch89]. More in general, this property holds for the class of Σ-protocol for proving the knowledge of a preimage of a group homomorphism defined in [CD98,Mau15]. The following is a formalization of this property.

Definition 5. *A Σ-protocol $\Pi = (\mathcal{P}, \mathcal{V})$ has* quasi-unique third round *if there exists a negligible function ν such that for any (x, a, c, z, z')*

$\text{Prob}\,[\, \mathcal{V}(x, a, c, z) = \mathcal{V}(x, a, c, z') = 1 \text{ and } z \neq z' \,] \leq \nu(\lambda)$.

Remark 1. Our protocol follows the requirements of [Fis05], hence, we will assume that the input sigma-protocols to our compiler have a quasi-unique third round. However, our result can be made more general using sigma-protocols that enjoy *Strong two-special soundness* as discussed in [Kas22]. Informally the notion of strong-two-special soundness requires the special soundness extractor to extract a witness even in the case where the input transcripts differ only in the last round. We refer to [Kas22] for more details.

2.3 Or-Composition of Σ-Protocols

In this section we recall the or-composition of Σ-protocols proposed in [CDS94]. Let $\Pi_0 = (\mathcal{P}_0, \mathcal{V}_0)$ be a Σ-protocol for the $\mathcal{NP}$-relation Rel_0 and $\Pi_1 = (\mathcal{P}_1, \mathcal{V}_1)$ be a Σ-protocol for the $\mathcal{NP}$-relation Rel_1. Moreover, let Sim_0 be the Special HVZK simulator for Π_0 and Sim_1 be the Special HVZK simulator for Π_1. We consider the following 3-round public coin protocol $\Pi^{\mathtt{OR}} = (\mathcal{P}^{\mathtt{OR}}, \mathcal{V}^{\mathtt{OR}})$ where $\mathcal{P}^{\mathtt{OR}}$ and $\mathcal{V}^{\mathtt{OR}}$ has a common input (x_0, x_1) where $x_0 \in \hat{L}_0$ and $x_1 \in \hat{L}_1$. $\mathcal{P}^{\mathtt{OR}}$ has a private input w_b with $b \in \{0, 1\}$ and $(x_b, w_b) \in \mathsf{Rel}_b$.

1. $\mathcal{P}^{\mathtt{OR}}$ picks $c_{1-b} \leftarrow \{0,1\}^l$, computes $(a_{1-b}, z_{1-b}) \leftarrow \mathsf{Sim}_{1-b}(x_{1-b}, c_{1-b})$ with $l \in \mathbb{N}$, computes $(\mathsf{aux}, a_b) \leftarrow \mathcal{P}_b(x_b, w_b)$ and sends (a_0, a_1) to $\mathcal{V}^{\mathtt{OR}}$.
2. $\mathcal{V}^{\mathtt{OR}}$ upon receiving (a_0, a_1), computes and sends a random string $c \leftarrow \{0,1\}^l$.
3. $\mathcal{P}^{\mathtt{OR}}$, upon receiving c computes $c_b = c \oplus c_{1-b}$ and $z_b \leftarrow \mathcal{P}_b(\mathsf{aux}, c_b)$ and sends (c_0, z_0, c_1, z_1) to $\mathcal{V}^{\mathtt{OR}}$.
4. $\mathcal{V}^{\mathtt{OR}}$, upon receiving (z_0, z_1) checks if $\mathcal{V}_0(x_0, a_0, c_0, z_0) = 1$ and $\mathcal{V}_1(x_1, a_1, c_1, z_1) = 1$ and $c = c_0 \oplus c_1$. If it is, then $\mathcal{V}^{\mathtt{OR}}$ outputs 1, 0 otherwise.

Theorem 2 ([CDS94,GMY06]). *Let Π_0 be a Σ-protocol for the $\mathcal{NP}$-relation Rel_0 and Π_1 be a Σ-protocol for the $\mathcal{NP}$-relation Rel_1 then $\Pi^{\mathtt{OR}}$ is a Σ-protocol that is perfect WI for relation $\mathsf{Rel}_{\mathtt{OR}} = \{((x_0, x_1), w) : ((x_0, w) \in \mathsf{Rel}_{L_0}) \vee ((x_1, w) \in \mathsf{Rel}_{L_1})\}$.*

We recall the above theorem just for completeness even though in this work we use $\Pi^{\mathtt{OR}}$ in a non-blackbox way and we do not rely on its WI property directly. We just find convenient to use the prover and the verifier of $\Pi^{\mathtt{OR}}$ to shorten and make more clear the description of the protocols proposed in this work. Only in the security proof we make non-black box use of $\Pi^{\mathtt{OR}}$ in order to rely on the security of the underling Σ-protocols Π_0 and Π_1.

3 Fischlin's NIZK Argument and its Randomized Version

In [Fis05] the author provides a NIZK AoK, where the AoK extractor relies only on the *observability* of the RO[5]. The main advantage of Fischlin's construction, other than its efficiency, is that the AoK extractor does not need to rewind the malicious prover in order to extract the witness. Following [Fis05], we refer to such an extractor as *online extractor*. As remarked in [Fis05], the arguments of knowledge with online extractors

[5] We observe that even though the author of [Fis05] talks about Proof of Knowledge, they still need to polynomially bound the number of queries that an adversary can make to the random oracle. To avoid any ambiguity, in this work we consider only the notion of AoK since the malicious prover is implicitly bounded by the number of queries that can be made to the RO.

are especially suitable for settings with concurrent executions. Indeed, Fischlin shows an example in which a NIZK argument with *standard* knowledge extractors cannot be used due to the rewinds made by the extractor. As we mentioned, Fischlin's construction works only if the input Σ-protocol has a quasi-unique third round. As recently observed in [Kas22] this excludes some interesting Σ-protocols, like for example the output protocols of the OR-transform of [CDS94]. Indeed, in [Kas22] the authors prove that Fischlin's NIZK argument is not even witness indistinguishable if compiled with a protocol obtained from the OR composition of [CDS94]. Interestingly, [Kas22] shows how to solve this problem by proposing a modification of Fischlin's protocol that works for a more general class of Σ-protocols that, in particular, includes the Σ-protocols output of the transform of [CDS94]. Our starting point is the modified version of Fischlin's protocol, which following [Kas22] we refer to as the *randomized* version of Fischlin's protocol. In this section we denote this randomized version of Fischlin's protocol with $\Pi^{\mathsf{Rand-F}}$, and then we show how to bootstrap $\Pi^{\mathsf{Rand-F}}$ to a NIZK AoK with quasi-polynomial time straight-line simulation and online extraction where simulator and extractor do not program the RO. We refer the reader to Fig. 1 for the formal description of $\Pi^{\mathsf{Rand-F}}$.

Let $\Pi = (\mathcal{P}, \mathcal{V})$ be a Σ-protocol with challenge length t (for t defined below) for the $\mathcal{NP}$-relation $\mathsf{Rel_L}$. The security parameter defines the parameters ℓ, r, t. The integers are related as follows: $r\ell = \lambda$, $t = \lceil \log \lambda \rceil \ell$. Define the following non-interactive argument system $\Pi^{\mathsf{Rand-F}} = (\mathcal{P}^{\mathsf{Fischlin}}, \mathcal{V}^{\mathsf{Fischlin}})$ for relation $\mathsf{Rel_L}$. The RO $\mathcal{O}$ maps to ℓ bits.

Common input: security parameter λ, $\mathcal{NP}$-statement $x \in L$, the parameters b, r, t as defined above.

Input to $\mathcal{P}^{\mathsf{Fischlin}}$: w s.t. $(x, w) \in \mathsf{Rel_L}$.

Proof. The prover $\mathcal{P}^{\mathsf{Fischlin}}$ executes the following steps.

1. Run r times $\mathcal{P}$ (each time using fresh randomness) on input (x, w) thus obtaining $((\mathtt{aux}_1, a_1), (\mathtt{aux}_2, a_2), \ldots, (\mathtt{aux}_r, a_r))$. Let $A = (a_1, a_2, \ldots, a_r)$.
2. For $i = 1, \ldots, r$
   (a) Set $\mathcal{E}_i = \emptyset$.
   (b) Sample $c_i \leftarrow \{0,1\}^t \backslash \mathcal{E}_i$, compute $z \leftarrow \mathcal{P}(\mathtt{aux}_i, c_i)$.
   (c) If $\mathcal{O}(x, A, i, c_i, z_i) \neq 0^b$, update $\mathcal{E}_i = \mathcal{E}_i \cup \{c_i\}$ and go to Step 2.*b*.
3. Output $\pi = (\{a_i, c_i, z_i\}_{i=1,\ldots,r})$.

Verification. The verifier $\mathcal{V}^{\mathsf{Fischlin}}$ on input x and $\pi = (\{a_i, c_i, z_i\}_{i=1,\ldots,r})$ accepts if and only if for $i = 1, \ldots, r$, $\mathcal{V}(x, a_i, c_i, z_i) = 1$ and $\mathcal{O}(x, A, i, c_i, z_i) = 0^\ell$.

Fig. 1: $\Pi^{\mathsf{Rand-F}}$: Randomized version of Fischlin's protocol proposed in [Kas22].

In [Kas22] the authors argue that this protocol has a small completeness error. As remarked in [Fis05], for deterministic verifiers this error can be removed by standard techniques, namely, by letting the prover check on behalf of the verifier that the proof is valid before outputting it; if not, the prover simply sends the witness to the verifier.

4 On the WI of Protocol in Fig. 1

In this section we show how to use the randomized Fischlin protocol to obtain an efficient NIWI argument of knowledge with a straight-line extractor that does not program the RO. To do that, we construct a Σ-protocol $\Pi^{\mathtt{OR}}$ for a relation Rel_0 OR Rel_1 using the compiler of Sec. 2.3 by combining a Σ-protocol Π_0 for Rel_0 with a Σ-protocol Π_1 for Rel_1. For completeness, we formally prove that the randomized Fischlin protocol is an AoK with online extraction even though $\Pi^{\mathtt{OR}}$ does not have a quasi-unique third round. A similar proof can be also found in [Kas22]. Then we prove that if $\Pi^{\mathtt{OR}}$ is perfect HVZK, then the randomized

Fischlin protocol applied on Π^{OR} is perfect WI in the non-programmable random-oracle. For more details on our proof approach, we refer the reader to the proof of Theorem 3. We denote the instantiation of the randomized Fischlin protocol that uses Π^{OR} as input with $\Pi^{\mathcal{WI}} = (\mathcal{P}^{\mathcal{WI}}, \mathcal{V}^{\mathcal{WI}})$ and propose it in Fig. 2. $\Pi^{\mathcal{WI}}$ is similar to the protocol proposed in Fig. 1, with the exception that it takes as input two Σ-protocols Π_0 and Π_1 and combines them using the compiler of [CDS94].

Let $\Pi_0 = (\mathcal{P}_0, \mathcal{V}_0)$ and $\Pi_1 = (\mathcal{P}_1, \mathcal{V}_1)$ be the Σ-protocols described above with challenge length t (for t defined below) for the $\mathcal{NP}$-relation $\mathsf{Rel}_{\mathsf{L}}$. The security parameter defines the parameters ℓ, r, t. The integers are related as follows: $r\ell = \lambda$, $t = \lceil \log \lambda \rceil \ell$. Define the following non-interactive argument system $\Pi^{\mathcal{WI}} = (\mathcal{P}^{\mathcal{WI}}, \mathcal{V}^{\mathcal{WI}})$ for relation $\mathsf{Rel}_{\mathsf{OR}}$. The RO $\mathcal{O}$ maps to ℓ bits.

Common input: security parameter λ, $\mathcal{NP}$-statement $x_0 \in L_0 \vee x_1 \in L_1$, the parameters b, r, S, t as defined above.

Input to $\mathcal{P}^{\mathcal{WI}}$: w_b s.t. $(x_b, w_b) \in \mathsf{Rel}_{\mathsf{b}}$.

Proof. The prover $\mathcal{P}^{\mathcal{WI}}$ executes the following steps.

1. On input (x_b, w_b) run r times $\mathcal{P}_b$ (each time using fresh randomness) on input (x_b, w_b) thus obtaining $((\mathtt{aux}_1^b, a_1^b), (\mathtt{aux}_2^b, a_2^b), \ldots, (\mathtt{aux}_r^b, \ldots, a_r^b))$.
2. For $i = 1, \ldots, r$, pick $c_i^{1-b} \leftarrow \{0,1\}^t$ and compute $(a_i^{1-b}, z_i^{1-b}) \leftarrow \mathsf{Sim}(x_{1-b}, c_i^{1-b})$.
   Let $A = ((a_1^0, a_1^1), (a_2^0, a_2^1), \ldots, (a_r^0, a_r^1)$.
3. For $i = 1, \ldots, r$:
   (a) Set $\mathcal{E}_i = \emptyset$.
   (b) Sample $c_i \leftarrow \{0,1\}^t \backslash \mathcal{E}_i$, compute $z_i^b \leftarrow \mathcal{P}_b(\mathtt{aux}_i, c_i^b)$.
   (c) If $\mathcal{O}(x_0, x_1, A, i, c_i, c_i^0, c_i^1, z_i^0, z_i^1) \neq 0^b$ where $c_i = c_i^0 \oplus c_i^1$, update $\mathcal{E}_i = \mathcal{E}_i \cup \{c_i\}$ and go to Step 3.b
4. Output $\pi = (\{a_i^0, a_i^1, c_i^0, c_i^1, z_i^0, z_i^1\}_{i=1,\ldots,r})$.

Verification. The verifier $\mathcal{V}^{\mathcal{WI}}$ accepts if and only if for $i = 1, \ldots, r$: $\mathcal{V}_0(x_0, a_i^0, c_i^0, z_i^0) = 1$, $\mathcal{V}_1(x_1, a_i^1, c_i^1, z_i^1) = 1$, $c_i = c_i^0 \oplus c_i^1$ and $\mathcal{O}(x_0, x_1, A, i, c_i, c_i^0, c_i^1, z_i^0, z_i^1) = 0^\ell$.

Fig. 2: $\Pi^{\mathcal{WI}}$: Our NIWI AoK with an on-line extractor that does not program the RO.

Theorem 3. *Let Π_0 be a Σ-protocol for the $\mathcal{NP}$-relation Rel_0 and Π_1 be a Σ-protocol for the $\mathcal{NP}$-relation Rel_1 such that both Π_0 and Π_1 have a quasi-unique third round, then $\Pi^{\mathcal{WI}}$ is perfect WI for the $\mathcal{NP}$-relation $\mathsf{Rel}_{\mathsf{OR}}$ and is an AoK with an online extractor that does not program the RO.*

Proof. **Completeness.** It follows exactly the completeness of Theorem 6.2 of [Kas22].

Witness Indistinguishability. We prove that if the transform of [CDS94] showed in Sec. 2.3 on input Π_0 and Π_1 outputs a new Σ-protocol that is perfect WI for the $\mathcal{NP}$-relation $\mathsf{Rel}_{\mathsf{OR}}$ then $\Pi^{\mathcal{WI}}$ is WI for $\mathsf{Rel}_{\mathsf{OR}}$ as well.

We assume by contradiction that $\Pi^{\mathcal{WI}}$ is not WI and then we construct an adversary $\mathcal{A}^{\mathsf{SHVZK}}$ that breaks the Special HVZK of either Π_0 or Π_1. More formally, let $\mathcal{A}^b_{\mathsf{ExpAWI}}$ be the view of the adversary in the experiment $\mathsf{ExpAWI}^b_{\Pi^{\mathcal{WI}},\mathcal{A}}(\lambda, \zeta)$. By contradiction, we have that there exists a distinguisher $\mathcal{D}$, and a non-negligible function δ such that

$$\left|\text{Prob}\left[\ \mathcal{D}(\lambda, \zeta, \mathcal{A}^0_{\mathsf{ExpAWI}}) = 1\ \right] - \text{Prob}\left[\ \mathcal{D}(\lambda, \zeta, \mathcal{A}^1_{\mathsf{ExpAWI}}) = 1\ \right]\right| = \delta(\lambda).$$

Let b, r, S, t be the parameters defined in the description of $\Pi^{\mathcal{WI}}$ in Fig. 2, we now define the hybrid experiment $\mathcal{H}_i$ and we denote with $\mathcal{A}_{\mathcal{H}_i}$ the view of the adversary in the experiment $\mathcal{H}_i$ with $i \in \{1, \ldots, r\}$.

The hybrid experiment $\mathcal{H}_i$ is formally described in Fig. 3 and takes as input the security parameter λ, the auxiliary information ζ for $\mathcal{A}$ and contains b, r, S and t.

Upon receiving (x, w_0, w_1) from $\mathcal{A}$ execute the following steps.

1. Run i times $\mathcal{P}^{\mathtt{OR}}$ (each time using fresh randomness) on input (x_0, x_1, w_1) thus obtaining $((\mathtt{aux}_1^1, a_1^1), \ldots, (\mathtt{aux}_i^1, a_i^1))$ and $((a_1^0, z_1^0), \ldots, (a_i^0, z_i^0))$.
2. Run $r - i$ times $\mathcal{P}^{\mathtt{OR}}$ (each time using fresh randomness) on input (x_0, x_1, w_0) thus obtaining $((\mathtt{aux}_{i+1}^0, a_{i+1}^0), \ldots, (\mathtt{aux}_r^0, a_r^0))$ and $((a_{i+1}^1, z_{i+1}^1), \ldots, (a_r^1, z_r^1))$. Let $A = ((a_1^0, a_1^1), \ldots, (a_r^0, a_r^1)$.
3. For $j = 1, \ldots, i$;
   Pick a random $c_j \in \{0,1\}^t$ such that $\mathcal{O}(x_0, x_1, A, j, c_j, c_j^0, c_j^1, z_j^0, z_j^1) = 0^\ell$ where $c_j = c_j^0 \oplus c_j^1$ and $z_j^1 \leftarrow \mathcal{P}_1(\mathtt{aux}_j^1, c_j^1)$.
   If such a c_j does not exist, abort the computation.
4. For $j = i + 1, \ldots, r$:
   Pick a random $c_j \in \{0,1\}^t$ such that $\mathcal{O}(x_0, x_1, A, j, c_j, c_j^0, c_j^1, z_j^0, z_j^1) = 0^\ell$ where $c_j = c_j^0 \oplus c_j^1$ and $z_j^0 \leftarrow \mathcal{P}_0(\mathtt{aux}_j^0, c_j^0)$.
5. Send $\pi = (\{a_i^0, a_i^1, c_i^0, c_i^1, z_i^0, z_i^1\}_{i=1,\ldots,r})$ to $\mathcal{A}$.

Fig. 3: Hybrid experiment $\mathcal{H}_i$, with $i \in \{1, \ldots, r\}$.

We observe that $\mathsf{ExpAWI}^0_{\Pi^{\mathcal{WI}},\mathcal{A}}(\lambda, \zeta) = \mathcal{H}_0(\lambda, \zeta)$ and that $\mathsf{ExpAWI}^1_{\Pi^{\mathcal{WI}},\mathcal{A}}(\lambda, \zeta) = \mathcal{H}_r(\lambda, \zeta)$. Therefore, by contradiction, there must be a value $i \in \{1, \ldots r\}$ such that $\left|\mathrm{Prob}\left[\, \mathcal{D}(\lambda, \zeta, \mathcal{A}_{\mathcal{H}_{i-1}}) = 1 \,\right] - \mathrm{Prob}\left[\, \mathcal{D}(\lambda, \zeta, \mathcal{A}_{\mathcal{H}_i}) = 1 \,\right]\right| = \delta(\lambda)$. To reach a contradiction we consider the additional intermediate hybrid experiment $\mathcal{H}_{\mathsf{int}}$ of Fig. 4. Informally, the difference between $\mathcal{H}_{i-1}$ and $\mathcal{H}_{\mathsf{int}}$ is that the honest prover procedure $\mathcal{P}_1$ is used in $\mathcal{H}_{\mathsf{int}}$ to compute (a_i^1, z_i^1) instead of the simulated one. Instead, the difference between $\mathcal{H}_{\mathsf{int}}$ and $\mathcal{H}_i$ is that in $\mathcal{H}_i$ the messages (a_i^0, z_i^0) are computed using the Special HVZK simulator of Π_0 instead of the honest prover procedure. The following lemma completes the proof of the theorem.

Upon receiving (x, w_0, w_1) from $\mathcal{A}$ execute the following steps.

1. Run $i - 1$ times $\mathcal{P}_1$ (each time using fresh randomness) on input (x_1, w_1) thus obtaining $((\mathtt{aux}_1^1, a_1^1), \ldots, (\mathtt{aux}_{i-1}^1, a_{i-1}^1))$.
2. For $j = 1, \ldots, i - 1$ pick $c_j^0 \leftarrow \{0,1\}^t$ and compute $(a_j^0, z_j^0) \leftarrow \mathsf{Sim}_0(x_0, c_j^0)$.
3. Run $\mathcal{P}_1$ in input (x_1, w_1) thus obtaining $(a_i^1, \mathtt{aux}_i^1)$ and run $\mathcal{P}_0$ in input (x_0, w_0) thus obtaining $(a_i^0, \mathtt{aux}_i^0)$.
4. Run $r - i$ times $\mathcal{P}_0$ (each time using fresh randomness) on input (x_0, w_0) thus obtaining $((\mathtt{aux}_{i+1}^0, a_{i+1}^0), \ldots, (\mathtt{aux}_r^0, a_r^0))$.
5. For $j = i + 1, \ldots, r$ pick $c_j^1 \leftarrow \{0,1\}^t$ and compute $(a_j^1, z_j^1) \leftarrow \mathsf{Sim}_1(x_1, c_j^1)$. Let $A = ((a_1^0, a_1^1), (a_2^0, a_2^1), \ldots, (a_r^0, a_r^1)$.
6. For $j = 1, \ldots, i - 1$: pick a random $c_j \in \{0,1\}^t$ such that $\mathcal{O}(x_0, x_1, A, j, c_j, c_j^0, c_j^1, z_j^0, z_j^1) = 0^\ell$ where $c_j = c_j^0 \oplus c_j^1$ and $z_j^1 \leftarrow \mathcal{P}_1(\mathtt{aux}_j^1, c_j^1)$; if such a c_j does not exist, then abort the computation.
7. Pick a random $c_i \in \{0,1\}^t$ such that $\mathcal{O}(x_0, x_1, A, i, c_i, c_i^0, c_i^1, z_i^0, z_i^1) = 0^\ell$ where $c_i^0 \leftarrow \{0,1\}^t$, $c_i = c_i^0 \oplus c_i^1$, $z_i^1 \leftarrow \mathcal{P}_1(\mathtt{aux}_i^1, c_i^1)$ and $z_i^0 \leftarrow \mathcal{P}_0(\mathtt{aux}_i^0, c_i^0)$
   If such a c_i does not exist abort the computation.
8. For $j = i + 1, \ldots, r$: pick a random $c_j \in \{0,1\}^t$ such that $\mathcal{O}(x_0, x_1, A, j, c_j, c_j^0, c_j^1, z_j^0, z_j^1) = 0^\ell$ where $c_j = c_j^0 \oplus c_j^1$ and $z_j^0 \leftarrow \mathcal{P}_0(\mathtt{aux}_j^0, c_j^0)$.
9. Send $\pi = (\{a_i^0, a_i^1, c_i^0, c_i^1, z_i^0, z_i^1\}_{i=1,\ldots,r})$ to $\mathcal{A}$.

Fig. 4: Hybrid experiment $\mathcal{H}_{\mathsf{int}}$.

Lemma 1. *There exists a negligible function ν such that for every distinguisher $\mathcal{D}$*

$|\text{Prob}[\ \mathcal{D}(\lambda,\zeta,\mathcal{A}_{\mathcal{H}_{i-1}})=1\ ]-\text{Prob}[\ \mathcal{D}(\lambda,\zeta,\mathcal{A}_{\mathcal{H}_{\mathsf{int}}})=1\ ]|\leq\nu(\lambda)$ *and*
$|\text{Prob}[\ \mathcal{D}(\lambda,\zeta,\mathcal{A}_{\mathcal{H}_{\mathsf{int}}}=1\ ]-\text{Prob}[\ \mathcal{D}(\lambda,\zeta,\mathcal{A}_{\mathcal{H}_i})=1\ ]|\leq\nu(\lambda)$

Proof. Assume by contradiction that one of the above two conditions does not hold. Therefore there exists a non-negligible function δ such that[6]

$|\text{Prob}[\ \mathcal{D}(\lambda,\zeta,\mathcal{A}_{\mathcal{H}_{i-1}})=1\ ]-\text{Prob}[\ \mathcal{D}(\lambda,\zeta,\mathcal{A}_{\mathcal{H}_{\mathsf{int}}})=1\ ]|=\delta(\lambda)$ In this case, we can construct an adversary $\mathcal{A}^{\mathsf{SHVZK}}$ that breaks the Special HVZK of Π_1. Let $\mathcal{C}^{\mathsf{SHVZK}}$ be the challenger of the Special HVZK security game, $\mathcal{A}^{\mathsf{SHVZK}}$ internally runs $\mathcal{A}$ and executes the following steps.

1. Upon receiving (x_0,x_1,w_0,w_1) from $\mathcal{A}$, pick $\mathsf{c}\leftarrow\{0,1\}^t$ and send (x_1,w_1,c) to $\mathcal{C}^{\mathsf{SHVZK}}$.
2. Upon receiving (a,z) from $\mathcal{C}^{\mathsf{SHVZK}}$, run $i-1$ times $\mathcal{P}_1$ (each time using fresh randomness) on input (x_1,w_1) thus obtaining $((\mathsf{aux}_1^1,a_1^1),\ldots,(\mathsf{aux}_{i-1}^1,a_{i-1}^1))$. For $j=1,\ldots,i-1$ pick $c_j^0\leftarrow\{0,1\}^t$ and compute $(a_j^0,z_j^0)\leftarrow\mathsf{Sim}_0(x_0,c_j^0)$.
3. Run $\mathcal{P}_0$ in input (x_0,w_0) thus obtaining (a_i^0,aux_i^0), set $a_i^1=\mathsf{a}$.
4. Run $r-i$ times $\mathcal{P}_0$ (each time using fresh randomness) on input (x_0,w_0) thus obtaining $((\mathsf{aux}_{i+1}^0,a_{i+1}^0),\ldots,(\mathsf{aux}_r^0,a_r^0))$.
5. For $j=i+1,\ldots,r$ pick $c_j^1\leftarrow\{0,1\}^t$ and compute $(a_j^1,z_j^1)\leftarrow\mathsf{Sim}_1(x_1,c_j^1)$.
6. Define $A=((a_1^0,a_1^1),(a_2^0,a_2^1),\ldots,(a_r^0,a_r^1))$.
7. For $j=1,\ldots,i-1$:
   Pick a random $c_j\in\{0,1\}^t$ such that $\mathcal{O}(x_0,x_1,A,j,c_j,c_j^0,c_j^1,z_j^0,z_j^1)=0^\ell$ where $c_j=c_j^0\oplus c_j^1$ and $z_j^1\leftarrow\mathcal{P}_1(\mathsf{aux}_j^1,c_j^1)$.
   If such a c_j does not exist, abort the computation.
8. Pick a random $c_i\in\{0,1\}^t$ such that $\mathcal{O}(x_0,x_1,A,i,c_i,c_i^0,c_i^1,z_i^0,z_i^1)=0^\ell$ where $c_i=\mathsf{c}\oplus c_i^0$, $z_i^0\leftarrow\mathcal{P}_0(\mathsf{aux}_i^0,c_i^0)$ and $z_i^1=\mathsf{z}$.
   If such a c_i does not exist, abort the computation.
9. For $j=i+1,\ldots,r$:
   Pick a random $c_j\in\{0,1\}^t$ such that $\mathcal{O}(x_0,x_1,A,j,c_j,c_j^0,c_j^1,z_j^0,z_j^1)=0^\ell$ where $c_j=c_j^0\oplus c_j^1$ and $z_j^0\leftarrow\mathcal{P}_0(\mathsf{aux}_j^0,c_j^0)$.
   If such a c_j does not exist, abort the computation.
10. Send $\pi=(\{a_i^0,a_i^1,c_i^0,c_i^1,z_i^0,z_i^1\}_{i=1,\ldots,r})$ to $\mathcal{A}$.
11. Output what $\mathcal{A}$ outputs.

We now observe that if the reduction does not abort, then we have the following. If the messages (a,z) have been computed by $\mathcal{C}^{\mathsf{SHVZK}}$ using the Special HVZK simulator Sim_1 then the output of $\mathcal{A}^{\mathsf{SHVZK}}$ corresponds to the output of $\mathcal{A}$ in $\mathcal{H}_{i-1}$. If instead (a,z) are computed using $\mathcal{P}_1$ on input the witness w_1 for x_1, then the output of the reduction corresponds to the output of $\mathcal{A}$ in $\mathcal{H}_{\mathsf{int}}$. The proof ends with the observation that the reduction aborts only with negligible probability (the same arguments used to prove the correctness of the scheme can be used here). Therefore $\mathcal{A}^{\mathsf{SHVZK}}$ breaks the security of the Special HVZK of Π_1 thus reaching a contradiction. □

Argument of knowledge. Here we follow in large part the proof of Theorem 2 of [Fis05]. We show a knowledge extractor $\mathsf{Ext}(x,\pi,\mathcal{Q}_{\mathcal{O}}(\mathcal{A}))$ that except with negligible probability over the choice of $\mathcal{O}$, outputs a witness w_b such that $(x_b,w_b)\in\mathsf{Rel}_b$ for an accepted proof $\pi=(\{a_i^0,a_i^1,c_i^0,c_i^1,z_i^0,z_i^1\}_{i=1,\ldots,r})$ with respect to (x_0,x_1). Since $\Pi^{\mathtt{OR}}$ is special-sound, then the algorithm Ext just needs to look into $\mathcal{Q}_{\mathcal{O}}(\mathcal{A})$ for a query $\mathcal{O}(x_0,x_1,A,i,c_i,c_i^0,c_i^1,z_i^0,z_i^1)$ and for another query $\mathcal{O}(x_0,x_1,A,i,\tilde{c}_i,\tilde{c}_i^0,\tilde{c}_i^1,\tilde{z}_i^0,\tilde{z}_i^1)$ such that $\mathcal{V}^{\mathtt{OR}}(x_0,x_1,a_i^0,a_i^1,\tilde{c}_i^0,\tilde{c}_i^1,\tilde{z}_i^0,\tilde{z}_i^1)=1$ and $c_i\neq\tilde{c}_i$. Indeed, from the special soundness of $\Pi^{\mathtt{OR}}$, this yields an efficient extraction procedure of either the witness for x_0 or x_1. If there are no such queries then Ext simply outputs $\perp$. We now need to bound the probability that such two queries do not exist, but still, $\mathcal{V}^{\mathcal{WI}}$ accepts π with non-negligible probability. Consider the set of tuples (x_0,x_1,A) such that $\mathcal{A}$ queries $\mathcal{O}$ about $(x_0,x_1,A,i,c_i^0,c_i^1,z_i^0,z_i^1)$ for some $i,c_i^0,c_i^1,z_i^0,z_i^1$ and such that $\mathcal{V}^{\mathtt{OR}}(x_0,x_1,a_i^0,a_i^1,c_i^0,c_i^1,z_i^0,z_i^1)=1$ (we can neglect tuples with invalid proofs since they are not useful to the prover to compute an accepting proof for

[6] The proof for the other case follows using the same arguments but in that case, we break the Special HVZK of Π_0 instead of Π_1.

the protocol). Let $Q = |\mathcal{Q}_{\mathcal{O}}(\mathcal{A})|$, then there are at most $Q+1$ of these tuples (x_0, x_1, A). Fix one of the tuples for the moment, say, (x_0, x_1, A). By contradiction, for this tuple and any i, $\mathcal{A}$ never queries $\mathcal{O}$ about two values $(x_0, x_1, A, i, c_i, c_i^0, c_i^1, z_i^0, z_i^1)$, $(x_0, x_1, a_i^0, a_i^1, \tilde{c}_i^0, \tilde{c}_i^1, \tilde{z}_i^0, \tilde{z}_i^1)$ with $c_i \neq \tilde{c}_i$ which $\mathcal{V}^{\mathsf{OR}}$ would accept (we note that if $c_i \neq \tilde{c}_i$ then either $c_i^0 \neq \tilde{c}_i^1$ or $c_i^1 \neq \tilde{c}_i^1$). Similarly, we can assume that $\mathcal{A}$ never queries about $(x_0, x_1, A, i, c_i, c_i^0, c_i^1, z_i^0, z_i^1)$, $(x_0, x_1, a_i^0, a_i^1, \tilde{c}_i^0, \tilde{c}_i^1, \tilde{z}_i^0, \tilde{z}_i^1)$ with either $z_i^0 \neq \tilde{z}_i^0$ or $z_i^1 \neq \tilde{z}_i^1$, otherwise this would contradict the property of unique responses of either Π_0 or Π_1. This allows us to assign a set of unique values $s_1, \ldots, s_r$ to (x_0, x_1, A) such that s_i equals $\mathcal{O}(x_0, x_1, A, i, c_i, c_i^0, c_i^1, z_i^0, z_i^1)$ if $\mathcal{A}$ queries about any such tuple. Conclusively, the values $s_1, \ldots, s_r$ assigned to (x_0, x_1, A) are all random and independent. Given such an assignment we calculate the probability that the values $s_1, \ldots, s_r$ are all equal to 0^ℓ. This is equivalent to the probability that r, ℓ-bit strings uniformly random sampled, are all equal to 0^ℓ. Such probability is $2^{-r\ell}$. Hence, having set $\lambda = r\ell$, we have that the probability that the extractor fails it at most $\mathcal{Q}_{\mathcal{O}}(\mathcal{A})/2^\lambda$.

5 Our Efficient NIZK Argument of Knowledge

In order to construct our NIZK argument of knowledge with quasi-polynomial simulation and online extraction (still without requiring simulator and extractor to program the RO) for the $\mathcal{NP}$ relation $\mathsf{Rel}_{\mathsf{L}}$ $\Pi^{\mathsf{NIZK}} = (\mathcal{P}^{\mathsf{NIZK}}, \mathcal{V}^{\mathsf{NIZK}})$, we make use of the following tools.

- A dense samplable puzzle system $\mathsf{PuzSys} = (\mathsf{Sample}, \mathsf{Solve}, \mathsf{Verify})$ such that for every hardness factor $h \in \mathcal{HS}_\lambda$ there exists a negligible function ν such that the following holds:
  1. Prob $\left[\mathsf{puz} \leftarrow \mathsf{Sample}(1^\lambda, h) : g(\mathsf{Steps}_{\mathsf{Solve}}(1^\lambda, h, \mathsf{puz})) \leq \lambda^{\log \lambda} \right] \leq \nu(\lambda)$;
  2. the worst-case running time of $\mathsf{Solve}(1^\lambda, h, \cdot)$ is $\lambda^{\mathsf{poly}(\log \lambda)}$.[7]
- $\Pi^{\mathcal{WI}} = (\mathcal{P}^{\mathcal{WI}}, \mathcal{V}^{\mathcal{WI}})$: a perfect NIWI AoK with online extractor for the $\mathcal{NP}$-relation $\mathsf{Rel}_{\mathcal{WI}} = \{((x, \mathsf{puz}), w) : (x, w) \in \mathsf{Rel}_{\mathsf{L}} \text{ or } \mathsf{Verify}(1^\lambda, h, \mathsf{puz}, w) = 1\}$.

$\mathcal{P}^{\mathsf{NIZK}}$ and $\mathcal{V}^{\mathsf{NIZK}}$ have also access to a RO (that will not be programmed by the simulator and by the extractor) $\mathcal{O} : \{0,1\}^* \to \{0,1\}^{\ell(\lambda,h)}$ where $\ell(\lambda, h)$ is a function of the security and the hardness parameters of PuzSys. We need to relate the output length of the random oracle to the parameters of PuzSys because $\mathcal{O}$ is used to generate a puzzle in our construction. More details are given in the security proof.

Common input: security parameter λ, $\mathcal{NP}$-statement $x \in L$.
Input to $\mathcal{P}^{\mathsf{NIZK}}$: w s.t. $(x, w) \in \mathsf{Rel}_{\mathsf{L}}$.

Proof. $\mathcal{P}^{\mathsf{NIZK}}$ computes the puzzle puz for PuzSys by querying the random oracle $\mathcal{O}$ on input x: $\mathsf{puz} \leftarrow \mathcal{O}(x)$. $\mathcal{P}^{\mathsf{NIZK}}$ defines $x_{\mathcal{WI}} = (x, \mathsf{puz})$, $w_{\mathcal{WI}} = w$ and runs $\mathcal{P}^{\mathcal{WI}}$ on input $(x_{\mathcal{WI}}, w_{\mathcal{WI}})$ thus obtaining $\pi_{\mathcal{WI}}$ which is sent to $\mathcal{V}^{\mathsf{NIZK}}$.
Verification. $\mathcal{V}^{\mathsf{NIZK}}$ queries $\mathcal{O}$ with x thus obtaining puz and defines $x_{\mathcal{WI}} = (x, \mathsf{puz})$. $\mathcal{V}^{\mathsf{NIZK}}$ now outputs what $\mathcal{V}^{\mathcal{WI}}$ outputs on input $(x_{\mathcal{WI}}, \pi_{\mathcal{WI}})$.

Fig. 5: Our NIZK AoK with straight-line simulator and online extractor.

Theorem 4. *If $\Pi^{\mathcal{WI}} = (\mathcal{P}^{\mathcal{WI}}, \mathcal{V}^{\mathcal{WI}})$ is a non-interactive perfect WI AoK with online extractor for the $\mathcal{NP}$-relation $\mathsf{Rel}_{\mathcal{WI}}$ that does not program the RO and PuzSys is a dense samplable puzzle system according to Definition 2, then Π^{NIZK} is a straight-line concurrent perfectly $\lambda^{\mathsf{poly}(\log \lambda)}$-simulatable argument of knowledge with online extraction where neither the zero-knowledge simulator nor the AoK extractor needs to program the RO.*

[7] This is the same puzzle used in Theorem 7 of [BKZZ16].

Proof. **Completeness.** It follows from the completeness of $\Pi^{\mathcal{WI}}$ and PuzSys.
Quasi-polynomial time perfect simulation. Let $\mathcal{V}^{\mathsf{NIZK}}$ be an arbitrary verifier. We construct a simulator Sim that runs in $\lambda^{\mathsf{poly}(\log\lambda)}$ time, such that the distributions $\left\{(\langle\mathcal{P}^{\mathsf{NIZK}}(w),\mathcal{V}^{\mathsf{NIZK}\star}(z)\rangle(x))\right\}_{z\in\{0,1\}^*,x\in L}$ for arbitrary w s.t. $(x,w)\in\mathsf{Rel}_{\mathsf{L}}$ and $\left\{(\langle\mathsf{Sim},\mathcal{V}^{\mathsf{NIZK}\star}(z)\rangle(x))\right\}_{z\in\{0,1\}^*,x\in L}$ are perfectly indistinguishable. The simulator Sim is described in Fig. 6. Note that Sim can compute the solution of the puzzle $\mathtt{puz}$ in time $\lambda^{\mathsf{poly}(\log\lambda)}$ and $\Pi^{\mathcal{WI}}$ is perfect WI.

$\mathsf{Sim}(1^\lambda,x)$
- Compute the puzzle $\mathtt{puz}$ for PuzSys by querying the random oracle $\mathcal{O}$ on input x: $\mathtt{puz}\leftarrow\mathcal{O}(x)$ and compute $\mathtt{sol}\leftarrow\mathsf{Solve}(1^\lambda,h,\mathtt{puz})$.
- Define $x_{\mathcal{WI}}=(x,\mathtt{puz})$, $w_{\mathcal{WI}}=\mathtt{sol}$ and run $\mathcal{P}^{\mathcal{WI}}$ on input $(x_{\mathcal{WI}},w_{\mathcal{WI}})$ thus obtaining $\pi_{\mathcal{WI}}$.
- Send $\pi_{\mathcal{WI}}$ to $\mathcal{V}^{\mathsf{NIZK}\star}$ and output what $\mathcal{V}^{\mathsf{NIZK}\star}$ outputs.

Fig. 6: Quasi-polynomial time simulator Sim.

Argument of knowledge. By assumption there exist a PPT AoK online extractor Ext for $\Pi^{\mathcal{WI}}$ such that the following holds for any *ppt* algorithm $\mathcal{P}^{\mathcal{WI}\star}$. Let $\mathcal{O}$ be a random oracle, $(x_{\mathcal{WI}},\pi_{\mathcal{WI}})\leftarrow\mathcal{P}^{\mathcal{WI}\star}(\lambda)$ and $\mathcal{Q}_{\mathcal{O}}(\mathcal{P}^{\mathcal{WI}\star})$ be the sequence of queries of $\mathcal{P}^{\mathcal{WI}\star}$ to $\mathcal{O}$ and $\mathcal{O}$'s answers. Let $w_{\mathcal{WI}}\leftarrow\mathsf{Ext}(x,\pi,\mathcal{Q}_{\mathcal{O}}(\mathcal{P}^{\mathcal{WI}\star}))$. Then there exists a negligible function ν such that

Prob $\left[\ (x_{\mathcal{WI}},w_{\mathcal{WI}})\notin\mathsf{Rel}_{\mathcal{WI}}\text{ and }\mathcal{V}^{\mathcal{WI}}(x_{\mathcal{WI}},\pi_{\mathcal{WI}})=1\ \right]\leq\nu(\lambda)$.

The AoK PPT extractor $\mathsf{E}^{\mathsf{NIZK}}$ for Π^{NIZK} simply internally runs Ext and outputs what Ext would output. We now observe that $\mathsf{E}^{\mathsf{NIZK}}$ could fail only because the output of the internal extractor Ext is a solution $\mathtt{sol}$ for $\mathtt{puz}=\mathcal{O}(x)$. Let us assume by contradiction that this happens. That is, let $\mathcal{O}$ be a random oracle, $(x,\pi_{\mathsf{NIZK}})\leftarrow\mathcal{P}^{\mathsf{NIZK}\star}(\lambda)$, $\mathcal{Q}_{\mathcal{O}}(\mathcal{P}^{\mathsf{NIZK}\star})$ be the sequence of queries of $\mathcal{P}^{\mathsf{NIZK}\star}$ to $\mathcal{O}$ and $\mathcal{O}$'s answers and let $\eta\leftarrow\mathsf{Ext}(x,\pi,\mathcal{Q}_{\mathcal{O}}(\mathcal{P}^{\mathsf{NIZK}\star}))$, then

Prob $\left[\ \mathsf{Verify}(1^\lambda,h,\mathcal{O}(x),\eta)=1\text{ and }\mathcal{V}^{\mathsf{NIZK}}(x,\pi_{\mathsf{NIZK}})=1\ \right]=\delta(\lambda)$.

But this would contradict the fact that PuzSys cannot be solved in less than $\lambda^{\log\lambda}$ steps. Indeed the extractor $\mathsf{E}^{\mathsf{NIZK}}$ outputs in PPT a solution to a puzzle sampled from an uniform distribution (since $\mathcal{O}$ is modelled as a random oracle). We observe that even though in the above proof the ZK simulator needs to run in quasi-polynomial time we can still rely on the WI property of $\Pi^{\mathcal{WI}}$ since the WI of $\Pi^{\mathcal{WI}}$ holds against all-powerful adversary.

5.1 Complexity Analysis

To give a concrete idea of the efficiency of our protocol we need to define the two Σ-protocols used as input of $\Pi^{\mathcal{WI}}$ (see Fig 2). Thus, we need a Σ-protocol $\Pi^{\mathtt{puz}}$ for the $\mathcal{NP}$-relation $\mathsf{Rel}_{\mathtt{puz}}=\{(\mathtt{puz},\mathtt{sol}):\mathsf{Verify}(1^\lambda,h,\mathtt{puz},w)=1)\}$ and a Σ-protocol Π^{L} for the $\mathcal{NP}$-relation $\mathsf{Rel}_{\mathsf{L}}$. We need to use a dense samplable puzzle system that admits a Σ-protocol. As discussed in Sec. 2.2, in [BKZZ15] the authors propose a dense samplable puzzle system where the puzzle is an instance of the DL problem. So we can instantiate $\Pi^{\mathtt{puz}}$ with the well known protocol of Schnorr [Sch89].

If we parametrize Fig. 2 with parameters $\ell=9,t=9,r=13$ then we obtain Following the analysis of we obtain an online-extractor that fails with probability at most $\mathcal{Q}2^{-117}$ where the total number of hash function evaluations is at most 2^9r and the number of executions of $\Pi^{\mathtt{OR}}$ is $r=13$, where $\Pi^{\mathtt{OR}}$ denotes the output of the or-composition proposed in Sec. 2.3 using $\Pi^{\mathtt{puz}}$ and Π^{L} as input. To give a practical example, let us consider the complexity of Π^{NIZK} when $\mathsf{Rel}_{\mathsf{L}}$ is the discrete log $\mathcal{NP}$ relation $\{((\mathcal{G},q,g,Y),y):g^y=Y\}$ where $\mathcal{G}$ is a group of prime-order q. We construct $\Pi_{\mathcal{WI}}$ using two instantiations of the Schnorr protocol

thus obtaining a Σ-protocol that requires 3 exponentiations to be executed (cf. [CPS+16]). So our protocol Π^{NIZK} requires 39 exponentiations and $2^9 r \approx 6000$ hash evaluations.

References

BCC+24. Christian Badertscher, Matteo Campanelli, Michele Ciampi, Luigi Russo, and Luisa Siniscalchi. Universally composable SNARKs with transparent setup without programmable random oracle. Cryptology ePrint Archive, Paper 2024/1549, 2024.

BFM88. Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 103–112, 1988.

BKZZ15. Foteini Baldimtsi, Aggelos Kiayias, Thomas Zacharias, and Bingsheng Zhang. Indistinguishable proofs of work or knowledge. Cryptology ePrint Archive, Paper 2015/1230, 2015. `https://eprint.iacr.org/2015/1230`.

BKZZ16. Foteini Baldimtsi, Aggelos Kiayias, Thomas Zacharias, and Bingsheng Zhang. Indistinguishable proofs of work or knowledge. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part II*, volume 10032 of *Lecture Notes in Computer Science*, pages 902–933, 2016.

BP04. Boaz Barak and Rafael Pass. On the possibility of one-message weak zero-knowledge. In *Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004, Cambridge, MA, USA, February 19-21, 2004, Proceedings*, pages 121–132, 2004.

BR93. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993.*, pages 62–73, 1993.

CD98. Ronald Cramer and Ivan Damgård. Zero-knowledge proofs for finite field arithmetic; or: Can zero-knowledge be for free? In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, volume 1462 of *Lecture Notes in Computer Science*, pages 424–441. Springer, 1998.

CDS94. Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings*, pages 174–187, 1994.

CJS14. Ran Canetti, Abhishek Jain, and Alessandra Scafuro. Practical UC security with a global random oracle. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*, pages 597–608, 2014.

CL24. Yi-Hsiu Chen and Yehuda Lindell. Optimizing and implementing fischlin's transform for uc-secure zero knowledge. *IACR Commun. Cryptol.*, 1(2):11, 2024.

CPS+16. Michele Ciampi, Giuseppe Persiano, Alessandra Scafuro, Luisa Siniscalchi, and Ivan Visconti. Improved or-composition of sigma-protocols. In Eyal Kushilevitz and Tal Malkin, editors, *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II*, volume 9563 of *Lecture Notes in Computer Science*, pages 112–141. Springer, 2016.

CPSV16. Michele Ciampi, Giuseppe Persiano, Luisa Siniscalchi, and Ivan Visconti. A transform for NIZK almost as efficient and general as the fiat-shamir transform without programmable random oracles. In *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II*, pages 83–111, 2016.

Fis05. Marc Fischlin. Communication-efficient non-interactive proofs of knowledge with online extractors. In *CRYPTO*, volume 3621 of *Lecture Notes in Computer Science*, pages 152–168. Springer, 2005.

FLS90. Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume I*, pages 308–317. IEEE Computer Society, 1990.

FS86. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, pages 186–194, 1986.

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

GMY06. Juan A. Garay, Philip MacKenzie, and Ke Yang. Strengthening zero-knowledge protocols using signatures. *Journal of Cryptology*, 19(2):169–209, 2006.

GQ88. Louis C. Guillou and Jean-Jacques Quisquater. A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. In Christoph G. Günther, editor, *Advances in Cryptology - EUROCRYPT '88, Workshop on the Theory and Application of of Cryptographic Techniques, Davos, Switzerland, May 25-27, 1988, Proceedings*, volume 330 of *Lecture Notes in Computer Science*, pages 123–128. Springer, 1988.

Kas22. Yashvanth Kondi and abhi shelat. Improved straight-line extraction in the random oracle model with applications to signature aggregation. Cryptology ePrint Archive, Paper 2022/393, 2022.

Lin15. Yehuda Lindell. An efficient transform from sigma protocols to NIZK with a CRS and non-programmable random oracle. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part I*, pages 93–109, 2015.

Mau15. Ueli Maurer. Zero-knowledge proofs of knowledge for group homomorphisms. *Designs, Codes and Cryptography*, pages 1–14, 2015.

Pas03a. Rafael Pass. On deniability in the common reference string and random oracle model. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, volume 2729 of *Lecture Notes in Computer Science*, pages 316–337. Springer, 2003.

Pas03b. Rafael Pass. Simulation in quasi-polynomial time, and its application to protocol composition. In Eli Biham, editor, *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, volume 2656 of *Lecture Notes in Computer Science*, pages 160–176. Springer, 2003.

Pas04a. Rafael Pass. Alternative variants of zero-knowledge proofs. Master's thesis, Kungliga Tekniska Högskolan, 2004. Licentiate Thesis Stockholm, Sweden.

Pas04b. Rafael Pass. Bounded-concurrent secure multi-party computation with a dishonest majority. In László Babai, editor, *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*, pages 232–241. ACM, 2004.

Sch89. C.P. Schnorr. Efficient identification and signatures for smart cards. In Gilles Brassard, editor, *Advances in Cryptology — CRYPTO' 89 Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 239–252. Springer New York, 1989.