Unconditionally Secure NIZK in the Fine-Grained Setting

Yuyu Wang*[1] and Jiaxin Pan**[2]

[1] University of Electronic Science and Technology of China, Chengdu, China
wangyuyu@uestc.edu.cn

[2] Department of Mathematical Sciences, NTNU - Norwegian University of Science and Technology, Trondheim, Norway
jiaxin.pan@ntnu.no

Abstract. Non-interactive zero-knowledge (NIZK) proof systems are often constructed based on cryptographic assumptions. In this paper, we propose the *first* unconditionally secure NIZK system in the AC^0-fine-grained setting. More precisely, our NIZK system has perfect soundness for all adversaries and unconditional zero-knowledge for AC^0 adversaries, namely, an AC^0 adversary can only break the zero-knowledge property with negligible probability unconditionally. At the core of our construction is an OR-proof system for satisfiability of 1 out of polynomial many statements.

Keywords. Non-interactive zero-knowledge, fine-grained cryptography, AC^0, unconditional security.

1 Introduction

Constructing non-interactive zero-knowledge (NIZK) proof systems [7] is one of the central topics in cryptography, since NIZK is a fundamental primitive that can convince a verifier the validity of a statement with minimum communication round.

Most NIZK systems are constructed based on various cryptographic assumptions, such as Discrete-Logarithm-like (e.g., [10,11]) and Learning With Errors (LWE, e.g., [17]) assumptions. Recent development of succinct NIZK systems [8,16,6,2,9] even base their security on rather strong, non-falsifiable assumptions, such as knowledge assumptions and assuming generic groups. Although there are many cryptanalysis results on assumptions, such as Discrete Logarithm and LWE, it is natural to consider whether it is possible to construct NIZK from much mild assumptions.

* Supported by the National Natural Science Foundation for Young Scientists of China under Grant Number 62002049, the Natural Science Foundation of Sichuan under Grant Number 2023NSFSC0472, the Sichuan Science and Technology Program under Grant Number 2022YFG0037, and the Fundamental Research Funds for the Central Universities under Grant Number ZYGX2020J017.

** Supported by the Research Council of Norway under Project No. 324235.

NIZK based on Mild Assumptions. Very recently, Wang and Pan [19] put forth this direction in the fine-grained setting. Here fine-grained setting (or fine-grained cryptography) [3] means that adversaries can only have bounded resources and honest users have no more resources than adversaries. More precisely, the work of Wang and Pan considers that all parties are in NC^1. In this setting, they obtained a NIZK system under a rather mild assumption, $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L}/\mathsf{poly}$. Their system is very efficient since only simple operations such as AND, OR, and PARITY for bits are involved. The assumption, $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L}/\mathsf{poly}$, also yields the security of proof systems in [5,1,20].

However, in complexity theory, it has not been proven that $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L}/\mathsf{poly}$, although it is widely accepted. It is desirable to further push this direction and study whether it is possible to construct an *unconditionally secure* NIZK system in the fine-grained setting.

We suppose that in the classical setting it seems not possible to have unconditional security for NIZK. The reason is that for proving the zero-knowledge property, the common reference string (CRS) is often related to the simulation trapdoor, and given the CRS an (unbounded) adversary may recover the simulation trapdoor and break the soundness. Meanwhile, it is promising to construct unconditionally secure NIZK in the fine-grained setting, since it restricts the capability of an adversary. However, this will also limit the resources of an honest user, which makes it particularly difficult to instantiate a scheme. Our technical goal is to resolve this tension.

1.1 Our Contributions

We consider the AC^0-fine-grained setting, namely, all adversaries, honest provers, and verifiers are in AC^0. In this setting, we construct the *first* unconditionally secure NIZK proof system for circuit satisfiability (SAT). More precisely, it is perfectly sound and has zero-knowledge against any adversaries in AC^0. Our system only involves simple operations in $GF(2)$ and does not require any cryptographic group operations or assumptions such as Discrete Logarithm and Factoring.

Our NIZK only supports statements verifiable in AC^0 given witnesses, since if a statement circuit is beyond AC^0 then an honest prover in AC^0 cannot decide its truth with the witness. However, we stress that our method is not limited to AC^0 statements. For instance, if we allow polynomial-time honest provers as in [1], our constructions naturally support statement circuits with polynomial-size. Moreover, any polynomial-size statement circuit can be represented as one verifiable in AC^0. Specifically, if a witness contains the bits of all wires in the circuit, then an AC^0 algorithm can efficiently verify the validity of an input/output pair of each gate in parallel and check whether the bit for the final output wire is 1. In this sense, the prover of our NIZK works for any NP statement, given a witness containing "enough information".

Applications of Security against AC^0. Security against AC^0 naturally captures adversaries with limited resources. Moreover, an AC^0-fine-grained NIZK works well in systems requiring "online security", where attacks are valid only if they succeed immediately. For instance, our NIZK with composable zero-knowledge against AC^0 and perfect soundness can be used to protect secrets only valuable in a short period of time. Also, its dual mode enjoys everlasting security. Namely, its perfect zero-knowledge continuously prevents the adversary from learning information on secrets and its soundness guarantees security in a system requiring users to provide proofs in a short time.

Impacts of Our Work. Our work gives us interesting insights to the minimum hardness assumptions required by NIZK and the landscape of AC^0-fine-grained cryptography. Before our work, it seemed that cryptographic assumptions, in particular, those imply public-key encryption (PKE), were necessary for NIZK in the standard model. Putting it in Impagliazzo's view of complexity landscape [14], NIZK seemed to be in the Cryptomania. Examples are Diffie-Hellman-based NIZKs [10,11]. Even in the NC^1-fine-grained setting, NIZK systems [19] require the assumption $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L}/\mathsf{poly}$, which implies PKE schemes [3].

Our work shows that those assumptions implying PKE are not necessary, since in the AC^0-fine-grained setting, it is not known whether there is a PKE scheme yet. [3] Up until now, only "minicrypt primitives" such as one-way function, weak pseudorandom function, secret-key encryption, and collision-resistant hash function are known to exist [12,3] in this setting, and we were not aware of any impossibility or possibility results showing that assumptions implying PKE are necessary for NIZK, in particular, in the AC^0-fine-grained setting, or not. As a further direction left open, we will explore how to extend our techniques in the classical setting and construct a NIZK from weaker assumptions (e.g., Discrete Logarithms) that are not known to imply PKE.

Extensions. While all the aforementioned NIZKs are in the CRS model, we can further extend them to the uniform random string (URS) model, where a trust setup only samples public coins. We also prove that our NIZKs have verifiable correlated key generations [10], which lead to a conversion from our NIZKs to unconditionally secure non-interactive zaps [4] (i.e., non-interactive witness-indistinguishability proof systems in the plain model) [10] against AC^0.

1.2 Technical Overview

In this section, we give more details about our techniques. Our approach is divided into three intermediate steps. We firstly construct a simple NIZK for linear languages, and then compile it to an OR-proof scheme for 1-out-of-ℓ disjunction, where ℓ can be any polynomial. Both schemes run in NC^0, which is a subset of AC^0. Thirdly, we use this OR-proof scheme to construct a NIZK system for circuit SAT.

A main technical hurdle throughout our work is that in the AC^0-fine-grained setting, many standard operations, such as computing the sum of a polynomial

[3] How to construct a provably secure PKE scheme in the AC^0-fine-grained setting is left as an open problem in [3].

number of random elements and multiplication of two random matrices, are not allowed. These operations can be easily performed in NC^1 and thus previous fine-grained NIZKs under complexity assumptions [1,19] are not confronted with this problem. As a result, it is more challenging to construct a NIZK (or any cryptographic scheme, in general) in AC^0, compared to the work of Wang and Pan [19].

NIZK for Linear Languages in AC^0. Our starting point is a simple NIZK that is computable in NC^0 and has perfect soundness and composable zero-knowledge against adversaries in AC^0 under no assumption. The linear languages we consider are of the form

$$\mathsf{L}_{\mathbf{M}} = \{\mathbf{t} : \exists \mathbf{w} \in \{0,1\}^t, \text{ s.t. } \mathbf{t} = \mathbf{M}\mathbf{w}\},$$

where each row vector in $\mathbf{M} \in \{0,1\}^{n \times t}$ is sparse. Here, by sparse we mean that each row vector in $\mathbf{M}$ has only constant Hamming weight. This restriction is inherent, since otherwise even the multiplication of $\mathbf{M}$ and $\mathbf{w}$ cannot be performed in NC^0. [4] However, this is still sufficient for our final NIZK for circuit SAT.

The technique behind our scheme is based on the fact that an AC^0 adversary cannot tell the parity of a random string with the size being the security parameter λ [13,15]. For our purpose, we explain it as the indistinguishability between the following distributions:

$$\underbrace{\{\mathbf{E}_\lambda \widetilde{\mathbf{r}} | \widetilde{\mathbf{r}} \xleftarrow{\$} \{0,1\}^{\lambda-1}\}}_{=D_0} \text{ and } \underbrace{\{\mathbf{E}_\lambda \widetilde{\mathbf{r}} + \mathbf{e}_\lambda^\lambda | \widetilde{\mathbf{r}} \xleftarrow{\$} \{0,1\}^{\lambda-1}}_{=D_1}\},$$

where $\mathbf{e}_\lambda^\lambda \in \{0,1\}^\lambda$ denotes constant vector with the parity being 1 and $\mathbf{E}_\lambda \in \{0,1\}^{\lambda \times (\lambda-1)}$ denotes a fixed constant matrix (see Section 2 for the formal definitions). More specifically, we prove that a vector sampled from D_0 (respectively, D_1) is uniformly distributed conditioned on the parity being 0 (respectively 1). A useful property of $\mathbf{E}_\lambda$ we will exploit is that each row and column vector in it has constant Hamming weight, which implies that multiplication between $\mathbf{E}_\lambda$ and $\widetilde{\mathbf{r}}$ or other matrices can be performed in NC^0.

For the aforementioned linear language $\mathsf{L}_{\mathbf{M}}$, we set the binding CRS as a vector $\mathbf{r}$ sampled from D_1. The prover computes $\mathbf{C} = \mathbf{M}\mathbf{R}$ and $\mathbf{D} = (\mathbf{R}||\mathbf{w}) \begin{pmatrix} \mathbf{E}_\lambda^\top \\ \mathbf{r}^\top \end{pmatrix}$, where $\mathbf{R} \xleftarrow{\$} \{0,1\}^{t \times (\lambda-1)}$, and the verifier accepts iff $(\mathbf{C}||\mathbf{x}) \begin{pmatrix} \mathbf{E}_\lambda^\top \\ \mathbf{r}^\top \end{pmatrix} = \mathbf{M}\mathbf{D}$. For each multiplication of matrices (or vectors) involved, one can see that either the row vectors of the left hand side matrix or the column vectors of the right hand side matrix have only constant Hamming weight. Hence, all the operations can be performed in NC^0. Roughly speaking, soundness follows from the fact that, for a valid proof, either $\mathbf{x}$ being in the span of $\mathbf{M}$ or $\mathbf{r}$ being in the span of $\mathbf{E}_\lambda$ must hold, while all $\mathbf{r} \in D_1$ are outside the span of $\mathbf{E}_\lambda$. To prove zero-knowledge, we switch the binding CRS to a hiding CRS by replacing the distribution of $\mathbf{r}$

[4] An NC^0 circuit cannot compute the inner product of two vectors unless one of them is sparse.

by D_0. In this case, seeing $\mathbf{C}$ and $\mathbf{D}$ simultaneously reveals no information on $\mathbf{w}$ except for $\mathbf{x}$. Due to this CRS switching, we call this zero-knowledge composable, and this change does not modify the view of an AC^0 adversary.

OR-Proof for One Disjunction. Following a fine-grained version of the "OR-proof techniques" [10,18], the above NIZK can be transformed to an OR-proof for the 1-out-of-2 disjunction (namely, satisfiabilty of 1 out of 2 statements). Let $\mathbf{r}$ be a binding CRS sampled from D_1. Assuming the prover knows the witness $\mathbf{w}$ of statement $\mathbf{x}_j$ for some $j \in \{0, 1\}$, it generates a hiding CRS $\mathbf{r}_{1-j}$ with a trapdoor $\widetilde{\mathbf{r}}_{1-j}$ and a binding CRS $\mathbf{r}_j$ such that $\mathbf{r}_j = \mathbf{r} - \mathbf{r}_{1-j}$. Then the prover generates proofs for $\mathbf{x}_j$ and $\mathbf{x}_{1-j}$ with $\mathbf{w}$ and $\widetilde{\mathbf{r}}_{1-j}$ respectively. The verifier receives $\mathbf{r}_0$ and generates $\mathbf{r}_1$ by itself for verification. Soundness follows from the fact that for any pair of $(\mathbf{r}_0, \mathbf{r}_1)$ such that $\mathbf{r} = \mathbf{r}_0 + \mathbf{r}_1$, at least one of $(\mathbf{r}_0, \mathbf{r}_1)$ must be a binding CRS with the parity being 1. Composable zero-knowledge follows from that switching the distribution of $\mathbf{r}$ to D_0 leads both $\mathbf{r}_0$ and $\mathbf{r}_1$ to become hiding CRSs.

OR-Proof for Multiple Disjunctions. While the above construction works for the 1-out-of-2 disjunction, our NIZK for all AC^0 circuit SAT requires 1-out-of-ℓ disjunction for any polynomial ℓ. This is due to the fact that an AC^0 circuit may contain unbounded fan-in AND or OR gates. A natural idea is to let the prover "split" $\mathbf{r}$ into ℓ CRSs $(\mathbf{r}_i)_{i\in[\ell]}$ instead of two, among which one is binding and $\ell - 1$ ones are hiding. However, this will result in workload beyond AC^0 for both the prover and the verifier. Especially, a prover with a witness for the jth statement will have to compute $\mathbf{r}_j = \mathbf{r} - \sum_{i\neq j} \mathbf{r}_i$ and the verifier will have to compute $\mathbf{r}_\ell = \mathbf{r} - \sum_{i=1}^{\ell-1} \mathbf{r}_i$. Neither of them can be performed in AC^0.

To overcome the above problems, we develop a new framework of OR-proof for multiple disjunctions. At the core of our framework is a verifiable "double layer" sampling procedure.

In the first layer, we adopt a distribution, say D'_0, which is the same as D_0 except that it outputs vectors with size ℓ. By running D'_0 for $\lambda - 1$ times, we immediately achieve a matrix in $\{0, 1\}^{\ell\times(\lambda-1)}$, which can be parsed as ℓ random vectors in $\{0, 1\}^{\lambda-1}$ with the sum being a $\mathbf{0}$-vector. In the second layer, we sample ℓ vectors from D_0, while using the vectors generated in the first layer as the internal randomness. This results in ℓ random vectors conditioned on the sum being a $\mathbf{0}$-vector and the parities being 0's. Assuming that the witness for the jth statement is known, we add the jth vector with the original CRS $\mathbf{r}$ from D_1 to obtain a binding CRS and use the rest $\ell - 1$ vectors as the hiding CRSs. Notice that when switching $\mathbf{r}$ to a hiding CRS sampled from D_0, the ℓ split CRSs are all randomly distributed in D_0 conditioned on the sum being $\mathbf{r}$. In this case, information on the index j is information-theoretically hidden, which preserves the zero-knowledge.

For verification, we propose a method to extract a matrix from the internal randomness used in the first layer. We then use the matrix as a witness to prove that the sum of the CRSs generated by the prover is exactly $\mathbf{r}$, via our NIZK for linear languages. In this way, we can convince the verifier that at least one of the CRSs must be binding, and thus soundness can be guaranteed.

In conclusion, the above sampling procedure gives rise to ways to split a CRS into multiple ones and to convince the verifier that some of the resulting CRSs is binding, while all the operations involved can be performed in AC^0. Combining this sampling procedure with our OR-proof for one disjunction, we achieve an OR-proof for multiple disjunction, which plays a key component of our NIZK for circuit SAT.

NIZK for Circuit SAT. We now give an overview on how we construct a NIZK for all statements verifiable in AC^0 (given a witness) by using our NIZK for linear languages and our OR-proof.

For a valid witness, we extend it to contain bits of all wires in the statement circuit and commit each bit w_i as $\mathsf{cm}_i = \mathbf{E}_\lambda \mathbf{r}_i + \mathbf{t}\mathsf{w}_i$, where $\mathbf{r}_i$ is a random vector in $\{0,1\}^{\lambda-1}$ and $\mathbf{t} \stackrel{\$}{\leftarrow} D_1$ is in the CRS. For the final output, we commit it as $\mathbf{t}$. Note that the commitment is additively homomorphic and $\mathbf{t}$ is a commitment to 1. For each NOT gate with input commitments $(\mathsf{cm}_{i1}, \mathsf{cm}_{i2})$, we use the NIZK for linear languages to prove that $\mathsf{cm}_{i1} + \mathsf{cm}_{i2} + \mathbf{t}$ is in the span of $\mathbf{E}_\lambda$, i.e., it commits to 0. For each AND gate with input commitments $(\mathsf{cm}_{ij})_{j\in[\ell]}$ and output commitments $\mathsf{cm}_{i(\ell+1)}$, we use an OR-proof for 1-out-of-$(\ell + 1)$ disjunction to prove that either both cm_{ij} and $\mathsf{cm}_{i(\ell+1)}$ commit to 0 for some $j \in [\ell]$ or $\mathsf{cm}_{ij} - \mathbf{t}$ commits to 0 for all $j \in [\ell + 1]$. Proofs for OR gates are generated analogously. Notice that when generating the proof of compliance for each AND (respectively, OR) gate, the prover needs to find the index of the lexicographically first 0-bit (respectively, 1-bit) of its input from the extended witness. While common ways may go beyond AC^0 due to the unbounded fan-in of each gate, we prove that this can indeed be performed in AC^0 by proposing concrete circuits (See Theorem 5 for details).

Due to the perfect soundness of the underlying OR-proof and NIZK for linear languages, if there exist valid proofs for all gates, we can extract a witness leading the circuit to output 1 by computing the parities of all commitments for the input wires of the circuit. Notice that the statement here is information-theoretical, and thus the extraction procedure is not necessarily runnable in AC^0. Moreover, when switching the distribution of $\mathbf{t}$ to D_0, all the commitments are just random vectors with parities being 0 and the proofs of the underlying NIZKs reveal no useful information.

If we only treat statements verifiable in NC^0, which consists only of fan-in 2 gates, rather than AC^0, we can further reduce the proof size by instantiating the underlying OR-proof with our warm-up construction for one disjunction.

Overview of Extensions. Due to the fact that a random string falls into D_0 and D_1 with half-half probability, we can also implement our construction in the URS model by running it for multiple times in parallel. Composable zero-knowledge of the resulting construction follows from that of the original NIZK and statistical soundness follows from the fact that at least one CRS falls into D_1 with overwhelming probability.

Moreover, we can merge each CRS of all our NIZKs into one vector sampled from D_1. In this case, switching a binding CRS to a hiding one can be efficiently done by changing a single bit, and for any two CRSs with the sum being a constant vector where only one entry is 1, at least one of them must be binding. This implies that our NIZKs have verifiable correlated key generation. Based on this observation, we can convert our NIZKs into unconditionally secure non-interactive ZAPs in AC^0, following the conversion technique in [10].

2 Preliminaries

Notations. We note that all arithmetic computations are over $GF(2)$ in this work. Namely, all arithmetic computations are performed with a modulus of 2, and addition and subtraction are equivalent. We write $a \stackrel{\$}{\leftarrow} \mathcal{A}(b)$ (respectively, $a = \mathcal{A}(b)$) to denote the random variable output by a probabilistic (respectively, deterministic) algorithm (or circuit) $\mathcal{A}$ on input b. By $x \stackrel{\$}{\leftarrow} \mathcal{S}$ we denote the process of sampling an element x from a set or distribution $\mathcal{S}$ uniformly at random. By $[n]$ we denote the set $\{1, \cdots, n\}$. By negl we denote an unspecified negligible function.

By $\mathbf{x} \in \{0,1\}^n$ we denote a column vector with size n, and by x_i we denote the ith element of a vector $\mathbf{x}$. By $\mathbf{x}_1 \circ \cdots \circ \mathbf{x}_\ell$ for some ℓ, we denote $(\mathbf{x}_1^\top, \cdots, \mathbf{x}_\ell^\top)^\top$, i.e., the concatenation of $(\mathbf{x}_i)_{i \in [\ell]}$.

For a matrix $\mathbf{A} \in \{0,1\}^{n \times t}$ with rank $t' \leq n$, we denote the sets $\{\mathbf{y} | \exists \mathbf{x} \text{ s.t. } \mathbf{y} = \mathbf{A}\mathbf{x}\}$ and $\{\mathbf{x} \,|\, \mathbf{A}\mathbf{x} = \mathbf{0}\}$ by $\mathrm{Span}(\mathbf{A})$ (i.e., the span of $\mathbf{A}$) and $\mathrm{Ker}(\mathbf{A})$ respectively. By $\mathbf{A}^\perp \in \{0,1\}^{n \times (n-t')}$ we denote a matrix of rank $n - t'$ in $\mathrm{Ker}(\mathbf{A}^\top)$. Note that for any $\mathbf{y} \notin \mathrm{Span}(\mathbf{A})$, we have $\mathbf{y}^\top \mathbf{A}^\perp \neq \mathbf{0}$. By $\overline{\mathbf{A}}$ (respectively, $\underline{\mathbf{A}}$) we denote the upper $(n-1) \times t$ matrix (respectively, lower $1 \times t$ vector) of $\mathbf{A}$.

By $\mathbf{I}_n$ we denote an identity matrix in $\{0,1\}^{n \times n}$. By $\mathbf{e}_n^i$ we denote the column vector in $\{0,1\}^n$ with the ith element being 1 and the other elements being 0. By $\mathbf{0}$ we denote a zero vector or matrix. By $\mathbf{f}_n^i$ we denote the vector in $\{0,1\}^n$ such that the first $i-1$ entries are 0's and the other entries are 1's. By $\mathbf{E}_n$ we denote the following $n \times (n-1)$ matrix, where the entries of the two main diagonals are 1's and the other entries are 0's.

$$\mathbf{E}_n = \begin{pmatrix} 1 & & & \\ 1 & 1 & & \\ & & \ddots & \\ & & & 1\ 1 \\ & & & 1 \end{pmatrix} \in \{0,1\}^{n \times (n-1)}.$$

One can check that $\mathbf{f}_n^1 \in \mathrm{Ker}(\mathbf{E}_n^\top)$ and $\mathbf{E}_n \mathbf{f}_{n-1}^i = \mathbf{e}_n^i + \mathbf{e}_n^n$ for $i \in [n-1]$.

2.1 Circuits in AC^0

We now recall the definitions of function family, NC^0, and AC^0.

Definition 1 (Function family). *A* function family *is a family of (possibly randomized) functions* $\mathcal{F} = \{f_\lambda\}_{\lambda\in\mathbb{N}}$*, where for each* λ*,* f_λ *has a domain* D^f_λ *and a range* R^f_λ*.*

Definition 2 (NC^0)**.** *The class of (non-uniform)* AC^0 *function families is the set of all function families* $\mathcal{F} = \{f_\lambda\}_{\lambda\in\mathbb{N}}$ *for which there is a polynomial* $p(\cdot)$ *and constant* d *such that for each* λ*,* f_λ *can be computed by a (randomized) circuit of size* $p(\lambda)$*, depth* d*, and fan-in 2 using* AND*,* OR*, and* NOT *gates.*

Definition 3 (AC^0)**.** *The class of (non-uniform)* AC^0 *function families is the set of all function families* $\mathcal{F} = \{f_\lambda\}_{\lambda\in\mathbb{N}}$ *for which there is a polynomial* $p(\cdot)$ *and constant* d *such that for each* λ*,* f_λ *can be computed by a (randomized) circuit of size* $p(\lambda)$*, depth* d*, and unbounded fan-in using* AND*,* OR*, and* NOT *gates.*

One can easily see that NC^0 is a subset of AC^0, and for any polynomial $n = n(\lambda)$ and $\mathbf{x}, \mathbf{y} \in \{0,1\}^n$ where either $\mathbf{x}$ or $\mathbf{y}$ has only constant Hamming weight, the inner product of $\mathbf{x}$ and $\mathbf{y}$ is computable in NC^0.

Let $\{\mathsf{PARITY}_\lambda\}_{\lambda\in\mathbb{N}}$ be the function family such that for all $\lambda \in \mathbb{N}$, PARITY_λ on input any $\mathbf{x} \in \{0,1\}^\lambda$ outputs $\sum_{i=1}^{\lambda} x_i$. The following theorem states that any AC^0 circuit has very small correlation with PARITY_λ.

Theorem 1 ([13,15])**.** *For any* $\mathcal{A} = \{a_\lambda\}_{\lambda\in\mathbb{N}} \in \mathsf{AC}^0$ *with size* p *and constant depth* d *and any* $\lambda \in \mathbb{N}$*, we have*

$$\left| \Pr_{\mathbf{x} \xleftarrow{\$} \{0,1\}^\lambda}[a_\lambda(\mathbf{x}) = 1 | \mathsf{PARITY}_\lambda(\mathbf{x}) = 1] - \Pr_{\mathbf{x} \xleftarrow{\$} \{\mathbf{0},\mathbf{1}\}^\lambda}[a_\lambda(\mathbf{x}) = 1 | \mathsf{PARITY}_\lambda(\mathbf{x}) = 0] \right| \leq 2^{-\Omega(\lambda/\log^{d-1}(p))}.$$

One can see that for any polynomial p in λ, $2^{-\Omega(\lambda/\log^{d-1}(p))} = 2^{-\Omega(\lambda/\log^{d-1}(\lambda))}$ is negligible.

2.2 Proof Systems

Definition 4 (Non-interactive zero-knowledge (NIZK) proof). *A* $\mathcal{C}_1$-NIZK *for a family of relations* $\{\mathsf{R}_\lambda\}_{\lambda\in\mathbb{N}}$ *is a function family* $\mathsf{NIZK} = \{\mathsf{Gen}_\lambda, \mathsf{Prove}_\lambda, \mathsf{Ver}_\lambda\}_{\lambda\in\mathbb{N}} \in \mathcal{C}_1$ *with the following properties.*

- Gen_λ *returns a binding CRS* crs*.*
- $\mathsf{Prove}_\lambda(\mathsf{crs}, \mathsf{x}, \mathsf{w})$ *returns a proof* π*.*
- $\mathsf{Ver}_\lambda(\mathsf{crs}, \mathsf{x}, \pi)$ *deterministically returns* 1 *(accept) or* 0 *(reject).*

Completeness *is satisfied if for all* $\lambda \in \mathbb{N}$*, all* (x, w) *such that* $\mathsf{R}_\lambda(\mathsf{x}, \mathsf{w}) = 1$*, all* $\mathsf{crs} \in \mathsf{Gen}_\lambda$*, and all* $\pi \in \mathsf{Prove}_\lambda(\mathsf{crs}, \mathsf{x}, \mathsf{w})$*, we have* $\mathsf{Ver}_\lambda(\mathsf{crs}, \mathsf{x}, \pi) = 1$*.*

$\mathcal{C}_2$-composable zero-knowledge *is satisfied if there exists a simulator* $\{\mathsf{TGen}_\lambda, \mathsf{Sim}_\lambda\}_{\lambda\in\mathbb{N}} \in \mathcal{C}_1$ *such that for any adversary* $\mathcal{A} = \{a_\lambda\}_{\lambda\in\mathbb{N}} \in \mathcal{C}_2$*, we have*

$$|\Pr[1 \xleftarrow{\$} a_\lambda(\mathsf{crs}) | \mathsf{crs} \xleftarrow{\$} \mathsf{Gen}_\lambda] - \Pr[1 \xleftarrow{\$} a_\lambda(\mathsf{crs}) | (\mathsf{crs}, \mathsf{td}) \xleftarrow{\$} \mathsf{TGen}_\lambda]| \leq \mathsf{negl}(\lambda),$$

and for all $\lambda \in \mathbb{N}$ *and all* (x, w) *such that* $\mathsf{R}_\lambda(\mathsf{x}, \mathsf{w}) = 1$*, the following distributions are identical.*

$$\pi \xleftarrow{\$} \mathsf{Prove}_\lambda(\mathsf{crs}, \mathsf{x}, \mathsf{w}) \text{ and } \pi \xleftarrow{\$} \mathsf{Sim}_\lambda(\mathsf{crs}, \mathsf{td}, \mathsf{x}),$$

where $(\mathsf{crs}, \mathsf{td}) \xleftarrow{\$} \mathsf{TGen}_\lambda$*.*

Perfect soundness *is satisfied if for all* $\lambda \in \mathbb{N}$*, all* $\mathsf{crs} \in \mathsf{Gen}_\lambda$*, all* $\mathsf{x} \notin \mathsf{L}_\lambda$*, and all* π*, we have* $\mathsf{Ver}_\lambda(\mathsf{crs}, \mathsf{x}, \pi) = 0$*.*

URS Model. In the above definition, if Gen_λ only returns a public string $\mathsf{crs} \xleftarrow{\$} \{0,1\}^{p(\lambda)}$ uniformly at random for some polynomial p, then we say that NIZK is in the *URS model*.

Non-Interactive Zap. A non-interactive zap is a witness-indistinguishable non-interactive proof system in the plain model, where there is no trusted setup. The definition is as follows.

Definition 5 (Non-interactive zap). *A* $\mathcal{C}_1$-non-interactive zap *for a family of relations* $\{\mathsf{R}_\lambda\}_{\lambda \in \mathbb{N}}$ *is a function family* $\mathsf{ZAP} = \{\mathsf{ZProve}_\lambda, \mathsf{ZVer}_\lambda\}_{\lambda \in \mathbb{N}} \in \mathcal{C}_1$ *with the following properties.*

- $\mathsf{ZProve}_\lambda(\mathsf{x}, \mathsf{w})$ *returns a proof* π*.*
- $\mathsf{ZVer}_\lambda(\mathsf{x}, \pi)$ *deterministically returns* 1 *(accept) or* 0 *(reject).*

Completeness *is satisfied if for all* $\lambda \in \mathbb{N}$ *and all* (x, w) *such that* $\mathsf{R}_\lambda(\mathsf{x}, \mathsf{w}) = 1$*, and all* $\pi \in \mathsf{ZProve}_\lambda(\mathsf{x}, \mathsf{w})$*, we have* $\mathsf{ZVer}_\lambda(\mathsf{x}, \pi) = 1$*.*

$\mathcal{C}_2$-witness indistinguishability *is satisfied if for all* $\lambda \in \mathbb{N}$*, all* $(\mathsf{x}, \mathsf{w}_0, \mathsf{w}_1)$ *such that* $\mathsf{R}_\lambda(\mathsf{x}, \mathsf{w}_0) = \mathsf{R}_\lambda(\mathsf{x}, \mathsf{w}_1) = 1$*, and any adversary* $\mathcal{A} = \{a_\lambda\}_{\lambda \in \mathbb{N}} \in \mathcal{C}_2$*, we have*

$$|\Pr[1 \xleftarrow{\$} a_\lambda(\mathsf{x}, \pi) | \pi \xleftarrow{\$} \mathsf{ZProve}_\lambda(\mathsf{x}, \mathsf{w}_0)] - \Pr[1 \xleftarrow{\$} a_\lambda(\mathsf{x}, \pi) | \pi \xleftarrow{\$} \mathsf{ZProve}_\lambda(\mathsf{x}, \mathsf{w}_1)]| \leq \mathsf{negl}(\lambda).$$

Perfect soundness *is satisfied if for all* $\lambda \in \mathbb{N}$*, all* $\mathsf{x} \notin \mathsf{L}_\lambda$*, and all* π*, we have* $\mathsf{ZVer}_\lambda(\mathsf{x}, \pi) = 0$*.*

3 NIZK for Linear Languages

In this section, we propose an NC^0-NIZK for linear languages with perfect soundness and AC^0-composable zero-knowledge. Before giving our construction, we prove the following lemma, which says that the uniform distribution in and out of the span of $\mathbf{E}_\lambda$ are indistinguishable for an AC^0 adversary.

Lemma 1. *For any* $\mathcal{A} = \{a_\lambda\}_{\lambda \in \mathbb{N}} \in \mathsf{AC}^0$ *and any* $\lambda \in \mathbb{N}$*, we have*

$$\left| \Pr_{\mathbf{r} \xleftarrow{\$} \{0,1\}^{\lambda-1}} [a_\lambda(\mathbf{E}_\lambda \mathbf{r}) = 1] - \Pr_{\mathbf{r} \xleftarrow{\$} \{0,1\}^{\lambda-1}} [a_\lambda(\mathbf{E}_\lambda \mathbf{r} + \mathbf{e}_\lambda^\lambda) = 1] \right| \leq \mathsf{negl}(\lambda).$$

Proof. We first note that for $\mathbf{r} \xleftarrow{\$} \{0,1\}^{\lambda-1}$, the first $\lambda - 1$ bits of $\mathbf{y} = \mathbf{E}_\lambda \mathbf{r} + \mathbf{e}_\lambda^\lambda b$ are uniformly distributed for $b \in \{0,1\}$, due to the fact that $\overline{\mathbf{E}}_\lambda$ is of full rank.

Moreover, the last bit of $\mathbf{y}$ is uniquely determined by the first $\lambda-1$ ones conditioned on $\mathsf{PARITY}_\lambda(\mathbf{y}) = {\mathbf{f}_\lambda^1}^\top \mathbf{y} = b$. Thus, $\mathbf{y}$ is uniformly distributed conditioned on $\mathsf{PARITY}_\lambda(\mathbf{y}) = b$. Then Lemma 1 follows immediately from Theorem 1. □

Our Construction. Let $\mathbf{M}$ be a matrix from $\{0,1\}^{n\times t}$, where $n = n(\lambda)$, $t = t(\lambda)$, and $t' = t'(\lambda)$ are polynomials in λ and the Hamming weight of each row vector in $\mathbf{M}$ is constant. We define the associated language as

$$\mathsf{L}_\mathbf{M} = \{\mathbf{x} : \exists \mathbf{w} \in \{0,1\}^t,\ \text{s.t. } \mathbf{x} = \mathbf{Mw}\}.$$

For the associated relation $\mathsf{R}_\mathbf{M}$, we have $\mathsf{R}_\mathbf{M}(\mathbf{x}, \mathbf{w}) = 1$ iff $\mathbf{x} = \mathbf{Mw}$. We give the construction of a NIZK LNIZK for $\{\mathsf{L}_\mathbf{M}\}_{\lambda\in\mathbb{N}}$ and its simulator in Figures 1 and 2 respectively.

Gen_λ:	$\mathsf{Prove}_\lambda(\mathsf{crs}, \mathbf{x}, \mathbf{w})$:	$\mathsf{Ver}_\lambda(\mathsf{crs}, \mathbf{x}, \pi)$:
$\widetilde{\mathbf{r}} \xleftarrow{\$} \{0,1\}^{\lambda-1}$	$\mathbf{R} \xleftarrow{\$} \{0,1\}^{t\times(\lambda-1)}$	Return 1 iff
$\mathbf{r} = \mathbf{E}_\lambda \widetilde{\mathbf{r}} + \mathbf{e}_\lambda^\lambda \in \{0,1\}^\lambda$	$\mathbf{C} = \mathbf{MR} \in \{0,1\}^{n\times(\lambda-1)}$	$(\mathbf{C}\|\|\mathbf{x})\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}^\top\end{pmatrix} = \mathbf{MD}$
Return $\mathsf{crs} = \mathbf{r}$	$\mathbf{D} = (\mathbf{R}\|\|\mathbf{w})\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}^\top\end{pmatrix} \in \{0,1\}^{t\times\lambda}$	
	Return $\pi = (\mathbf{C}, \mathbf{D})$	

Fig. 1. Definition of $\mathsf{LNIZK} = \{\mathsf{Gen}_\lambda, \mathsf{Prove}_\lambda, \mathsf{Ver}_\lambda\}_{\lambda\in\mathbb{N}}$.

TGen_λ:	$\mathsf{Sim}_\lambda(\mathsf{crs}, \mathsf{td}, \mathbf{x})$:
$\widetilde{\mathbf{r}} \xleftarrow{\$} \{0,1\}^{\lambda-1}$	$\mathbf{R}' \xleftarrow{\$} \{0,1\}^{t\times(\lambda-1)}$
$\mathbf{r} = \mathbf{E}_\lambda \widetilde{\mathbf{r}}$	$\mathbf{C} = \mathbf{MR}' - \mathbf{x} \cdot \widetilde{\mathbf{r}}^\top$, $\mathbf{D} = \mathbf{R}'\mathbf{E}_\lambda^\top$
Return $\mathsf{crs} = \mathbf{r}$ and $\mathsf{td} = \widetilde{\mathbf{r}}$	Return $\pi = (\mathbf{C}, \mathbf{D})$

Fig. 2. Definition of the simulator $\{\mathsf{TGen}_\lambda, \mathsf{Sim}_\lambda\}_{\lambda\in\mathbb{N}}$ of LNIZK.

Theorem 2. LNIZK *in Figure 1 is an* NC^0*-NIZK with perfect soundness and* AC^0*-composable zero-knowledge.*

Proof. **Complexity.** First, we note that in Figures 1 and 2, the Hamming weight of each row vector in $\mathbf{E}_\lambda$, $\mathbf{M}$, and $\mathbf{x}$ and each column vector in $\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}^\top\end{pmatrix}$ is constant. [5] Thus, the multiplication of matrices involved can be performed in NC^0. Since

[5] Notice that $\mathbf{x}$ can be treated as a matrix with row vectors with Hamming weight at most 1.

addition of a constant number of matrices can be performed in NC^0 as well, we have $\{\mathsf{Gen}_\lambda, \mathsf{Prove}_\lambda, \mathsf{Ver}_\lambda, \mathsf{TGen}_\lambda, \mathsf{Sim}_\lambda\}_{\lambda\in\mathbb{N}} \in \mathsf{NC}^0$.

Completeness. Completeness follows from the fact that for $\mathbf{x} = \mathbf{M}\mathbf{w}$, $\mathbf{C} = \mathbf{M}\mathbf{R}$, and $\mathbf{D} = (\mathbf{R}||\mathbf{w})\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}^\top\end{pmatrix}$, we have

$$(\mathbf{C}||\mathbf{x})\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}^\top\end{pmatrix} = (\mathbf{M}\mathbf{R}||\mathbf{M}\mathbf{w})\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}^\top\end{pmatrix} = \mathbf{M}(\mathbf{R}||\mathbf{w})\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}^\top\end{pmatrix} = \mathbf{M}\mathbf{D}.$$

AC^0-Composable Zero-Knowledge. The indistinguishability between CRSs generated by Gen_λ and TGen_λ follows immediately from Lemma 1.

For $\mathbf{r} = \mathbf{E}_\lambda\widetilde{\mathbf{r}} \in \mathsf{TGen}_\lambda$ and $\mathbf{x} = \mathbf{M}\mathbf{w}$, we have $\mathbf{M}\mathbf{R} = \mathbf{M}(\mathbf{R} + \mathbf{w}\cdot\widetilde{\mathbf{r}}^\top) - \mathbf{x}\cdot\widetilde{\mathbf{r}}^\top$ and

$$(\mathbf{R}||\mathbf{w})\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}^\top\end{pmatrix} = (\mathbf{R}||\mathbf{w})\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \widetilde{\mathbf{r}}^\top\mathbf{E}_\lambda^\top\end{pmatrix} = (\mathbf{R} + \mathbf{w}\cdot\widetilde{\mathbf{r}}^\top)\mathbf{E}_\lambda^\top.$$

Moreover, for $\mathbf{R} \xleftarrow{\$} \{0,1\}^{t\times(\lambda-1)}$, the distribution of $\mathbf{R} + \mathbf{w}\cdot\widetilde{\mathbf{r}}^\top$ is uniformly random in $\{0,1\}^{t\times(\lambda-1)}$. Thus, for any valid statement, the simulator perfectly simulates honest proofs, completing the proof of composable zero-knowledge.

Perfect Soundness. Recall that $\mathbf{f}_\lambda^1$ denotes the vector consisting only of 1's and $\mathbf{f}_\lambda^1 \in \mathrm{Ker}(\mathbf{E}_\lambda^\top)$. When $\mathbf{r}$ is generated as $\mathbf{r} \xleftarrow{\$} \mathsf{Gen}_\lambda$, we have $\mathbf{r} \notin \mathrm{Span}(\mathbf{E}_\lambda)$ since ${\mathbf{f}_\lambda^1}^\top\mathbf{r} = 1$. Moreover, for any valid statement/proof pair $(\mathbf{x}, (\mathbf{C}, \mathbf{D}))$ such that $(\mathbf{C}||\mathbf{x})\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}^\top\end{pmatrix} = \mathbf{M}\mathbf{D}$, we have ${\mathbf{M}^\perp}^\top(\mathbf{C}||\mathbf{x})\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}^\top\end{pmatrix} = \mathbf{0}$, i.e., $\mathbf{E}_\lambda(\mathbf{C}^\top\mathbf{M}^\perp) = \mathbf{r}(\mathbf{x}^\top\mathbf{M}^\perp)$. When $\mathbf{r} \notin \mathrm{Span}(\mathbf{E}_\lambda)$, we must have $\mathbf{x}^\top\mathbf{M}^\perp = \mathbf{0}$, which in turn implies $\mathbf{x} \in \mathsf{L}_\mathbf{M}$, completing the proof of statistical soundness. Notice that in this part, the arguments are information-theoretical and the equations are not necessarily efficiently computable.

Putting all the above together, Theorem 2 immediately follows. □

Remark. By replacing Gen_λ by TGen_λ in LNIZK, we immediately achieve a fine-grained NIZK with perfect zero-knowledge and computational soundness. Similar arguments can also be made for our OR-proofs and NIZK for circuit SAT given in the following sections.

4 NIZK for OR-languages

In this section, we extend the NIZK LNIZK in Section 3 to an OR-proof system. We first give an efficient warm-up construction for 1-out-of-2 disjunction languages, and then show how to extend it to a fully-fledged one for the disjunction of polynomial number of linear languages.

4.1 A Warm-Up Construction

Let $n_0 = n_0(\lambda)$, $n_1 = n_1(\lambda)$, $t_0 = t_0(\lambda)$, and $t_1 = t_1(\lambda)$ be any polynomials in λ. We define the following language

$$\mathsf{L}^{\mathsf{or}}_{(\mathbf{M}_0,\mathbf{M}_1)} = \{(\mathbf{x}_0, \mathbf{x}_1) : \exists\mathbf{w} \text{ s.t. } \mathbf{x}_0 = \mathbf{M}_0\mathbf{w} \vee \mathbf{x}_1 = \mathbf{M}_1\mathbf{w}\},$$

where $\mathbf{M}_i \in \{0,1\}^{n_i \times t_i}$ and the Hamming weight of each row vector in $\mathbf{M}_i$ is constant for $i \in \{0,1\}$. For the associated relation $\mathsf{R}^{\mathsf{or}}_{(\mathbf{M}_0,\mathbf{M}_1)}$, we have $\mathsf{R}^{\mathsf{or}}_{(\mathbf{M}_0,\mathbf{M}_1)}((\mathbf{x}_0,\mathbf{x}_1),\mathbf{w}) = 1$ iff $\mathbf{x}_j = \mathbf{M}_j\mathbf{w}$ for some $j \in \{0,1\}$. The OR-proof and its simulator are given in Figures 3 and 4 respectively. Roughly, the prover splits the original binding CRS $\mathbf{r}$ into a binding one $\mathbf{r}_j$ and a hiding one $\mathbf{r}_{1-j}$ for some $j \in \{0,1\}$, and respectively uses the witness and trapdoor to generate proofs for the two linear statements. The verifer on receiving $\mathbf{r}_0$ recovers $\mathbf{r}_1$ as $\mathbf{r}_1 = \mathbf{r} - \mathbf{r}_0$ and executes the verification procedure.

$\underline{\mathsf{ORGen}_\lambda:}$
$\widetilde{\mathbf{r}} \xleftarrow{\$} \{0,1\}^{\lambda-1}$, $\mathbf{r} = \mathbf{E}_\lambda\widetilde{\mathbf{r}} + \mathbf{e}^\lambda_\lambda \in \{0,1\}^\lambda$
Return $\mathsf{crs} = \mathbf{r}$

$\underline{\mathsf{ORProve}_\lambda(\mathsf{crs}, (\mathbf{x}_0,\mathbf{x}_1), \mathbf{w}):}$
Let $j \in \{0,1\}$ s.t. $\mathbf{x}_j = \mathbf{M}_j\mathbf{w}$
Sample $\widetilde{\mathbf{r}}_{1-j} \xleftarrow{\$} \{0,1\}^{\lambda-1}$ and compute $\mathbf{r}_{1-j} = \mathbf{E}_\lambda\widetilde{\mathbf{r}}_{1-j}$ and $\mathbf{r}_j = \mathbf{r} - \mathbf{r}_{1-j}$
Sample $\mathbf{R}'_{1-j} \xleftarrow{\$} \{0,1\}^{t_{1-j}\times(\lambda-1)}$ and compute

$$\mathbf{C}_{1-j} = \mathbf{M}_{1-j}\mathbf{R}'_{1-j} - \mathbf{x}_{1-j}\cdot\widetilde{\mathbf{r}}^\top_{1-j} \in \{0,1\}^{n_{1-j}\times(\lambda-1)}, \mathbf{D}_{1-j} = \mathbf{R}'_{1-j}\mathbf{E}^\top_\lambda \in \{0,1\}^{t_{1-j}\times\lambda}$$

Sample $\mathbf{R}_j \xleftarrow{\$} \{0,1\}^{t_j\times(\lambda-1)}$ and compute

$$\mathbf{C}_j = \mathbf{M}_j\mathbf{R}_j, \mathbf{D}_j = (\mathbf{R}_j||\mathbf{w})\begin{pmatrix}\mathbf{E}^\top_\lambda \\ \mathbf{r}^\top_j\end{pmatrix}$$

Return $\pi = ((\mathbf{C}_i,\mathbf{D}_i)_{i\in\{0,1\}}, \mathbf{r}_0)$

$\underline{\mathsf{ORVer}_\lambda(\mathsf{crs}, (\mathbf{x}_0,\mathbf{x}_1), \pi):}$
$\mathbf{r}_1 = \mathbf{r} - \mathbf{r}_0$
Return 1 iff $(\mathbf{C}_i||\mathbf{x}_i)\begin{pmatrix}\mathbf{E}^\top_\lambda \\ \mathbf{r}^\top_i\end{pmatrix} = \mathbf{M}_i\mathbf{D}_i$ for all $i \in \{0,1\}$

Fig. 3. Definition of $\mathsf{ORNIZK}_{\mathsf{wm}} = \{\mathsf{ORGen}_\lambda, \mathsf{ORProve}_\lambda, \mathsf{ORVer}_\lambda\}_{\lambda\in\mathbb{N}}$.

Theorem 3. $\mathsf{ORNIZK}_{\mathsf{wm}}$ *in Figure 3 is an* NC^0*-NIZK with perfect soundness and* AC^0*-composable zero-knowledge.*

Proof. **Complexity.** First, we note that in Figures 3 and 4, the Hamming weight of each row vector in $\mathbf{E}_\lambda$, $\mathbf{M}_i$, and $\mathbf{x}_i$ and each column vector in $\begin{pmatrix}\mathbf{E}^\top_\lambda \\ \mathbf{r}^\top_i\end{pmatrix}$ is constant for all $i \in \{0,1\}$. Thus, the multiplication of matrices involved can be performed in NC^0. Also, addition of a constant number of matrices can be performed in NC^0. Hence, we have $\{\mathsf{ORGen}_\lambda, \mathsf{ORProve}_\lambda, \mathsf{ORVer}_\lambda, \mathsf{ORTGen}_\lambda, \mathsf{ORSim}_\lambda\}_{\lambda\in\mathbb{N}} \in \mathsf{NC}^0$.

$\underline{\mathsf{ORTGen}_\lambda}$:
$\widetilde{\mathbf{r}} \xleftarrow{\$} \{0,1\}^{\lambda-1}$, $\mathbf{r} = \mathbf{E}_\lambda \widetilde{\mathbf{r}}$
Return $\mathsf{crs} = \mathbf{r}$ and $\mathsf{td} = \widetilde{\mathbf{r}}$

$\underline{\mathsf{ORSim}_\lambda(\mathsf{crs}, \mathsf{td}, (\mathbf{x}_0, \mathbf{x}_1))}$:
Sample $\widetilde{\mathbf{r}}_0 \xleftarrow{\$} \{0,1\}^\lambda$ and compute $\widetilde{\mathbf{r}}_1 = \widetilde{\mathbf{r}} - \widetilde{\mathbf{r}}_0$, $\mathbf{r}_0 = \mathbf{E}_\lambda \widetilde{\mathbf{r}}_0$, and $\mathbf{r}_1 = \mathbf{E}_\lambda \widetilde{\mathbf{r}}_1$
For all $i \in \{0,1\}$, compute

$$\mathbf{R}'_i \xleftarrow{\$} \{0,1\}^{t_i \times (\lambda-1)}, \mathbf{C}_i = \mathbf{M}_i \mathbf{R}'_i - \mathbf{x}_i \cdot \widetilde{\mathbf{r}}_i^\top, \mathbf{D}_i = \mathbf{R}'_i \mathbf{E}_\lambda^\top$$

Return $\pi = ((\mathbf{C}_i, \mathbf{D}_i)_{i=0,1}, \mathbf{r}_0)$

Fig. 4. Definition of the simulator $\{\mathsf{ORTGen}_\lambda, \mathsf{ORSim}_\lambda\}_{\lambda \in \mathbb{N}}$ of $\mathsf{ORNIZK}_{\mathsf{wm}}$.

Completeness. Completeness follows from the fact that for $\mathbf{x}_j = \mathbf{M}_j \mathbf{w}$, $\mathbf{C}_j = \mathbf{M}_j \mathbf{R}_j$, and $\mathbf{D}_j = (\mathbf{R}_j || \mathbf{w}) \begin{pmatrix} \mathbf{E}_\lambda^\top \\ \mathbf{r}_j^\top \end{pmatrix}$, we have

$$(\mathbf{C}_j || \mathbf{x}_j) \begin{pmatrix} \mathbf{E}_\lambda^\top \\ \mathbf{r}_j^\top \end{pmatrix} = (\mathbf{M}_j \mathbf{R}_j || \mathbf{M}_j \mathbf{w}) \begin{pmatrix} \mathbf{E}_\lambda^\top \\ \mathbf{r}_j^\top \end{pmatrix} = \mathbf{M}_j \mathbf{D}_j,$$

and for $\mathbf{C}_{1-j} = \mathbf{M}\mathbf{R}'_{1-j} - \mathbf{x}_{1-j} \cdot \widetilde{\mathbf{r}}_{1-j}^\top$ and $\mathbf{D}_{1-j} = \mathbf{R}'_{1-j} \mathbf{E}_\lambda^\top$, we have

$$\begin{aligned}(\mathbf{C}_{1-j} || \mathbf{x}_{1-j}) \begin{pmatrix} \mathbf{E}_\lambda^\top \\ \mathbf{r}_{1-j}^\top \end{pmatrix} =& ((\mathbf{M}\mathbf{R}'_{1-j} - \mathbf{x}_{1-j} \cdot \widetilde{\mathbf{r}}_{1-j}^\top) || \mathbf{x}_{1-j}) \begin{pmatrix} \mathbf{E}_\lambda^\top \\ \mathbf{r}_{1-j}^\top \end{pmatrix} \\ =& \mathbf{M}\mathbf{R}'_{1-j} \mathbf{E}_\lambda = \mathbf{M}\mathbf{D}_{1-j}.\end{aligned}$$

AC^0**-Composable Zero-Knowledge.** The indistinguishability between CRSs generated by Gen_λ and TGen_λ follows immediately from Lemma 1.

When the CRS is generated as $\mathbf{r} = \mathbf{E}_\lambda \widetilde{\mathbf{r}}$ where $\widetilde{\mathbf{r}} \xleftarrow{\$} \{0,1\}^{\lambda-1}$, $\mathbf{r}_0$ and $\mathbf{r}_1$ generated by both $\mathsf{ORProve}_\lambda$ and ORSim_λ are uniformly distributed in $\mathrm{Span}(\mathbf{E}_\lambda)$, conditioned on $\mathbf{r} = \mathbf{r}_0 + \mathbf{r}_1$. Moreover, we have

$$\mathbf{M}_j \mathbf{R}_j = \mathbf{M}_j (\mathbf{R}_j + \mathbf{w} \cdot \widetilde{\mathbf{r}}^\top) - \mathbf{x}_j \cdot \widetilde{\mathbf{r}}^\top$$

and

$$(\mathbf{R}_j || \mathbf{w}) \begin{pmatrix} \mathbf{E}_\lambda^\top \\ \widetilde{\mathbf{r}}_j^\top \mathbf{E}_\lambda^\top \end{pmatrix} = (\mathbf{R}_j + \mathbf{w} \cdot \widetilde{\mathbf{r}}_j^\top) \mathbf{E}_\lambda^\top$$

for $\mathbf{x}_j = \mathbf{M}_j \mathbf{w}$. Since the distribution of $\mathbf{R}_j + \mathbf{w} \cdot \widetilde{\mathbf{r}}_j^\top$ for $\mathbf{R}_j \xleftarrow{\$} \{0,1\}^{t_j \times (\lambda-1)}$ is uniform in $\{0,1\}^{t_j \times (\lambda-1)}$, the simulator perfectly simulates honest proofs, completing the proof of composable zero-knowledge.

Perfect Soundness. Recall that $\mathbf{f}_\lambda^1$ denotes the vector consisting only of 1's and $\mathbf{f}_\lambda^1 \in \mathrm{Ker}(\mathbf{E}_\lambda^\top)$. For $\mathbf{r} \in \mathsf{Gen}_\lambda$, we have ${\mathbf{f}_\lambda^1}^\top \mathbf{r} = 1$, i.e., $\mathbf{r} \notin \mathrm{Span}(\mathbf{E}_\lambda)$. Hence, for a valid statement/proof pair (x, π) where $\mathsf{x} = (\mathbf{x}_0, \mathbf{x}_1)$ and $\pi = ((\mathbf{C}_i, \mathbf{D}_i)_{i\in\{0,1\}}, \mathbf{r}_0)$, we must have $\mathbf{r}_j \notin \mathrm{Span}(\mathbf{E}_\lambda)$ and $(\mathbf{C}_j||\mathbf{x}_j)\begin{pmatrix}\mathbf{E}_\lambda^\top\\ \mathbf{r}_j^\top\end{pmatrix} = \mathbf{M}_j\mathbf{D}_j$ for some $j \in \{0,1\}$, where $\mathbf{r}_1 = \mathbf{r} - \mathbf{r}_0$. For such j, we have $(\mathbf{M}_j^\perp)^\top(\mathbf{C}_j||\mathbf{x}_j)\begin{pmatrix}\mathbf{E}_\lambda^\top\\ \mathbf{r}_j^\top\end{pmatrix} = \mathbf{0}$, i.e., $\mathbf{r}_j(\mathbf{x}_j^\top\mathbf{M}_j^\perp) = \mathbf{E}_\lambda(\mathbf{C}_j^\top\mathbf{M}_j^\perp)$. Since $\mathbf{r}_j \notin \mathrm{Span}(\mathbf{E}_\lambda)$, we must have $\mathbf{x}_j^\top\mathbf{M}_j^\perp = 0$, which in turn implies $\mathsf{x} \in \mathsf{L}^{\mathsf{or}}_{\mathbf{M}_0,\mathbf{M}_1}$, completing the proof of perfect soundness. Notice that this part of arguments is information-theoretical and thus the equations are not necessarily computable in AC^0.

Putting all the above together, Theorem 3 immediately follows. $\square$

Remark. As discussed in Section 1.2, the above construction can not be naturally extended to 1-out-of-ℓ disjunction for any polynomial ℓ, due to the fact that an AC^0 algorithm cannot compute the sum of a polynomial number of random vectors (even conditioned on the parity being fixed). Specifically, if we extend the construction in a straightforward way, the prover and the verifier will have to compute $\mathbf{r}_j = \mathbf{r} - \sum_{i\neq j}\mathbf{r}_i$ and $\mathbf{r}_\ell = \mathbf{r} - \sum_{i=1}^{\ell-1}\mathbf{r}_i$ respectively, while neither can be performed in AC^0. In the next section, we propose a new method to overcome this problem.

4.2 A Fully-Fledged Construction

We now extend the warm-up OR-proof to a fully-fledged one for 1-out-of-ℓ disjunction.

Let $\ell = \ell(\lambda)$, $(n_i = n_i(\lambda))_{i\in[\ell]}$, $(t_i = t_i(\lambda))_{i\in[\ell]}$ be any polynomials in λ. We define the following languages:

$$\mathsf{L}_{\mathbf{E}_\ell} = \{\mathbf{Y} : \exists \mathbf{W} \in \{0,1\}^{(\ell-1)\times\lambda},\ \text{s.t. } \mathbf{Y} = \mathbf{E}_\ell\mathbf{W}\}.$$

and

$$\mathsf{L}^{\mathsf{or}}_{(\mathbf{M}_i)_{i\in[\ell]}} = \{(\mathbf{x}_i)_{i=1}^{\ell} : \exists \mathbf{w} \in \{0,1\}^{t_i},\ \text{s.t. } \bigvee_{i\in[\ell]} \mathbf{x}_i = \mathbf{M}_i\mathbf{w}\},$$

where $\mathbf{M}_i \in \{0,1\}^{n_i\times t_i}$ and the Hamming weight of each row vector in $\mathbf{M}_i$ is constant for $i \in [\ell]$. One can easily see that $\{\mathsf{L}_{\mathbf{E}_\ell}\}_{\lambda\in\mathbb{N}}$ is supported by our NIZK for linear languages given in Section 3, since $\mathsf{L}_{\mathbf{E}_\ell}$ is equivalent to the following linear language:

$$\mathsf{L}'_{\mathbf{E}_\ell} = \{(\mathbf{y}_i)_{i\in[\ell]} : \exists \mathbf{w} \in \{0,1\}^{(\ell-1)\lambda},\ \text{s.t. } \mathbf{y}_1 \circ \cdots \circ \mathbf{y}_\ell = \mathbf{M}\mathbf{w}\}$$

where $\mathbf{Y} = (\mathbf{y}_i)_{i\in[\ell]}$ and

$$\mathbf{M} = \begin{pmatrix}\mathbf{E}_\ell & \mathbf{0} & \cdots & \mathbf{0}\\ \mathbf{0} & \ddots & \ddots & \vdots\\ \vdots & \ddots & \ddots & \mathbf{0}\\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{E}_\ell\end{pmatrix} \in \{0,1\}^{\ell\cdot\lambda\times(\ell-1)\lambda}$$

contains $\mathbf{E}_\lambda$'s in the main diagonal and $\mathbf{0}$ in the other positions. Here recall that $\mathbf{y}_1 \circ \cdots \circ \mathbf{y}_\ell$ denotes the concatenation of $(\mathbf{y}_i)_{i\in[\ell]}$. It is easy to see that the Hamming weight of each row vector in $\mathbf{M}$ is constant.

Let $\mathsf{LNIZK} = \{\mathsf{Gen}_\lambda, \mathsf{Prove}_\lambda, \mathsf{Ver}_\lambda\}_{\lambda\in\mathbb{N}}$ be a NIZK with a simulator $\{\mathsf{TGen}_\lambda, \mathsf{Sim}_\lambda\}_{\lambda\in\mathbb{N}}$ for $\{\mathsf{L}_{\mathbf{E}_\ell}\}_{\lambda\in\mathbb{N}}$, we give an OR-proof for $\{\mathsf{L}^{\mathsf{or}}_{(\mathbf{M}_i)_{i\in[\ell]}}\}_{\lambda\in\mathbb{N}}$ and its simulator in Figures 5 and 6 respectively.

Roughly, we adopt a verifiable sampling procedure with double layers to split the original CRS into $\ell-1$ hiding CRSs and one binding CRS. In the first layer, we sample ℓ vectors with a trapdoor $\mathbf{S}$, and in the second layer, we in turn use the ℓ vectors as trapdoors to sample ℓ random hiding CRSs with the sum being 0, and add one of them with $\mathbf{r}$ to make it binding. Later, we use a NIZK for linear languages to prove that the sum of the ℓ CRSs is $\mathbf{r}$, where the witness can be extracted from $\mathbf{S}$. In this way, a verifier in AC^0 can check that at least one of the split CRSs is binding, without learning any useful information.

Theorem 4. *If* LNIZK *is an* NC^0*-NIZK with perfect soundness and* AC^0*-composable zero-knowledge, then* ORNIZK *in Figure 5 is an* NC^0*-NIZK with perfect soundness and* AC^0*-composable zero-knowledge.*

Proof. **Complexity.** First, we note that in Figures 5 and 6, the Hamming weight of each row vector in $\mathbf{E}_\lambda$, $\mathbf{E}_{\ell-1}$, $\mathbf{M}_i$, and $\mathbf{x}_i$ and each column vector in $\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}_i^\top\end{pmatrix}$ is constant for all $i \in [\ell]$. Thus, the multiplication of matrices involved can be performed in NC^0. Since addition of a constant number of matrices and running LNIZK and its simulator can be performed in NC^0 as well, we have $\{\mathsf{ORGen}_\lambda, \mathsf{ORProve}_\lambda, \mathsf{ORVer}_\lambda, \mathsf{ORTGen}_\lambda, \mathsf{ORSim}_\lambda\}_{\lambda\in\mathbb{N}} \in \mathsf{NC}^0$.

Completeness. For $\mathbf{x}_j = \mathbf{M}_j\mathbf{w}$, $\mathbf{C}_j = \mathbf{M}_j\mathbf{R}_j$, and $\mathbf{D}_j = (\mathbf{R}_j||\mathbf{w})\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}_j^\top\end{pmatrix}$, we have

$$(\mathbf{C}_j||\mathbf{x}_j)\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}_j^\top\end{pmatrix} = (\mathbf{M}_j\mathbf{R}_j||\mathbf{M}_j\mathbf{w})\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}_j^\top\end{pmatrix} = \mathbf{M}_j\mathbf{D}_j.$$

For $(\mathbf{r}_i)_{i\in[\ell]} = \mathbf{R} = \mathbf{E}_\lambda\widetilde{\mathbf{R}} + \mathbf{r}\cdot{\mathbf{e}_\ell^j}^\top$, we have $\mathbf{r}_i = \mathbf{E}_\lambda\widetilde{\mathbf{r}}_i$ for all $i \in [\ell]\backslash\{j\}$. Then, for $\mathbf{C}_i = \mathbf{M}\mathbf{R}'_i - \mathbf{x}_i\cdot\widetilde{\mathbf{r}}_i^\top$ and $\mathbf{D}_i = \mathbf{R}'_i\mathbf{E}_\lambda^\top$ where $i \in [\ell]\backslash\{j\}$, we have

$$(\mathbf{C}_i||\mathbf{x}_i)\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}_i^\top\end{pmatrix} =((\mathbf{M}\mathbf{R}'_i - \mathbf{x}_i\cdot\widetilde{\mathbf{r}}_i^\top)||\mathbf{x}_i)\begin{pmatrix}\mathbf{E}_\lambda^\top \\ \mathbf{r}_i^\top\end{pmatrix} = \mathbf{M}\mathbf{R}'_i\mathbf{E}_\lambda = \mathbf{M}\mathbf{D}_i.$$

Moreover, since $\mathbf{E}_\ell\mathbf{f}_{\ell-1}^j = \mathbf{e}_\ell^j + \mathbf{e}_\ell^\ell$, for $\widetilde{\mathbf{R}}^\top = \mathbf{E}_\ell\mathbf{S}$ and $\mathbf{R} = \mathbf{E}_\lambda\widetilde{\mathbf{R}} + \mathbf{r}\cdot{\mathbf{e}_\ell^j}^\top$, we have

$$\begin{aligned}\mathbf{R}^\top =&\widetilde{\mathbf{R}}^\top\mathbf{E}_\lambda^\top + \mathbf{e}_\ell^j\cdot\mathbf{r}^\top \\ =&\mathbf{E}_\ell\mathbf{S}\mathbf{E}_\lambda^\top + \mathbf{e}_\ell^j\cdot\mathbf{r}^\top \\ =&\mathbf{E}_\ell\mathbf{S}\mathbf{E}_\lambda^\top + (\mathbf{e}_\ell^\ell\mathbf{r}^\top + \mathbf{e}_\ell^j\cdot\mathbf{r}^\top) + \mathbf{e}_\ell^\ell\cdot\mathbf{r}^\top \\ =&\mathbf{E}_\ell\mathbf{S}\mathbf{E}_\lambda^\top + \mathbf{E}_\ell\mathbf{f}_{\ell-1}^j\mathbf{r}^\top + \mathbf{e}_\ell^\ell\mathbf{r}^\top \\ =&\mathbf{E}_\ell(\mathbf{S}\mathbf{E}_\lambda^\top + \mathbf{f}_{\ell-1}^j\mathbf{r}^\top) + \mathbf{e}_\ell^\ell\mathbf{r}^\top,\end{aligned}$$

$\underline{\mathsf{ORGen}_\lambda:}$
$\widetilde{\mathbf{r}} \xleftarrow{\$} \{0,1\}^{\lambda-1}$, $\mathbf{r} = \mathbf{E}_\lambda \widetilde{\mathbf{r}} + \mathbf{e}_\lambda^\lambda \in \{0,1\}^\lambda$, $\mathsf{crs} \xleftarrow{\$} \mathsf{Gen}_\lambda$
Return $\mathsf{crs}_{\mathsf{or}} = (\mathsf{crs}, \mathbf{r})$

$\underline{\mathsf{ORProve}_\lambda(\mathsf{crs}_{\mathsf{or}}, (\mathbf{x}_i)_{i\in[\ell]}, \mathbf{w}):}$
Let $\mathbf{x}_j = \mathbf{M}_j \mathbf{w}$ for some $j \in [\ell]$
Sample $\mathbf{S} \xleftarrow{\$} \{0,1\}^{(\ell-1)\times(\lambda-1)}$
Compute $\widetilde{\mathbf{R}}^\top = \mathbf{E}_\ell \mathbf{S} \in \{0,1\}^{\ell\times(\lambda-1)}$ and $\mathbf{R} = \mathbf{E}_\lambda \widetilde{\mathbf{R}} + \mathbf{r} \cdot {\mathbf{e}_\ell^j}^\top \in \{0,1\}^{\lambda\times\ell}$
Parse $\mathbf{R} = (\mathbf{r}_i)_{i\in[\ell]}$ and $\widetilde{\mathbf{R}} = (\widetilde{\mathbf{r}}_i)_{i\in[\ell]}$
For all $i \in [\ell]\backslash\{j\}$, sample $\mathbf{R}'_i \xleftarrow{\$} \{0,1\}^{t_i\times(\lambda-1)}$ and compute

$$\mathbf{r}_i = \mathbf{E}_\lambda \widetilde{\mathbf{r}}_i \in \{0,1\}^\lambda, \mathbf{C}_i = \mathbf{M}_i \mathbf{R}'_i - \mathbf{x}_i \cdot \widetilde{\mathbf{r}}_i^\top \in \{0,1\}^{n_i\times(\lambda-1)}, \mathbf{D}_i = \mathbf{R}'_i \mathbf{E}_\lambda^\top \in \{0,1\}^{t_i\times\lambda}$$

Sample $\mathbf{R}_j \xleftarrow{\$} \{0,1\}^{t_j\times(\lambda-1)}$ and compute

$$\mathbf{C}_j = \mathbf{M}_j \mathbf{R}_j \text{ and } \mathbf{D}_j = (\mathbf{R}_j || \mathbf{w}) \begin{pmatrix} \mathbf{E}_\lambda^\top \\ \mathbf{r}_j^\top \end{pmatrix}$$

Compute

$$\pi \xleftarrow{\$} \mathsf{Prove}_\lambda(\mathsf{crs}, \mathbf{R}^\top - \mathbf{e}_\ell^\ell \mathbf{r}^\top, \mathbf{S}\mathbf{E}_\lambda^\top + \mathbf{f}_{\ell-1}^j \mathbf{r}^\top)$$

Return $\pi_{\mathsf{or}} = ((\mathbf{C}_i, \mathbf{D}_i)_{i\in[\ell]}, \mathbf{R}, \pi)$

$\underline{\mathsf{ORVer}_\lambda(\mathsf{crs}, (\mathbf{x}_i)_{i\in[\ell]}, \pi_{\mathsf{or}}):}$
Parse $\mathbf{R} = (\mathbf{r}_i)_{i\in[\ell]}$
Return 1 iff $(\mathbf{C}_i || \mathbf{x}_i) \begin{pmatrix} \mathbf{E}_\lambda^\top \\ \mathbf{r}_i^\top \end{pmatrix} = \mathbf{M}_i \mathbf{D}_i$ for all $i \in [\ell]$ and

$$\mathsf{Ver}_\lambda(\mathsf{crs}, \mathbf{R}^\top - \mathbf{e}_\ell^\ell \mathbf{r}^\top, \pi) = 1$$

Fig. 5. Definition of $\mathsf{ORNIZK} = \{\mathsf{ORGen}_\lambda, \mathsf{ORProve}_\lambda, \mathsf{ORVer}_\lambda\}_{\lambda\in\mathbb{N}}$. Recall that by $\mathbf{f}_{\ell-1}^j \in \{0,1\}^{\ell-1}$ we denote the vector such that the first $j-1$ entries are 0's and the last $\ell - j$ ones are 1's.

i.e., $\mathbf{R}^\top - \mathbf{e}_\ell^\ell \mathbf{r}^\top = \mathbf{E}_\ell(\mathbf{S}\mathbf{E}_\lambda^\top + \mathbf{f}_{\ell-1}^j \mathbf{r}^\top)$. Then the completeness of ORNIZK follows immediately from that of LNIZK.

AC^0-Composable Zero-Knowledge. The indistinguishability between CRSs generated by ORGen_λ and ORTGen_λ follows immediately from the composable zero-knowledge of LNIZK and Lemma 1.

Next we define a modified prover $\mathsf{ORProve}_\lambda{}'$, which is exactly the same as $\mathsf{ORProve}_\lambda$ except that π is generated as $\pi \xleftarrow{\$} \mathsf{Sim}_\lambda(\mathsf{crs}, \mathsf{td}, \mathbf{R}^\top - \mathbf{e}_\ell^\ell \mathbf{r}^\top)$. The following distributions are identical due to the composable zero-knowledge of ORNIZK.

$$\Pi \xleftarrow{\$} \mathsf{ORProve}_\lambda(\mathsf{crs}_{\mathsf{or}}, (\mathbf{x}_i)_{i\in[\ell]}, \mathbf{w}) \text{ and } \Pi \xleftarrow{\$} \mathsf{ORProve}_\lambda{}'(\mathsf{crs}_{\mathsf{or}}, (\mathbf{x}_i)_{i\in[\ell]}, \mathbf{w}),$$

$\underline{\mathsf{ORTGen}_\lambda}$:
$\widetilde{\mathbf{r}} \xleftarrow{\$} \{0,1\}^{\lambda-1}$, $\mathbf{r} = \mathbf{E}_\lambda \widetilde{\mathbf{r}} \in \{0,1\}^\lambda$, $(\mathsf{crs}, \mathsf{td}) \xleftarrow{\$} \mathsf{TGen}_\lambda$
Return $\mathsf{crs}_{\mathsf{or}} = ((\mathsf{crs}, \mathbf{r}), \mathsf{td}_{\mathsf{or}} = (\mathsf{td}, \widetilde{\mathbf{r}}))$

$\underline{\mathsf{ORSim}_\lambda(\mathsf{crs}, \mathsf{td}, (\mathbf{x}_i)_{i \in [\ell]})}$:
Sample $\mathbf{S} \xleftarrow{\$} \{0,1\}^{(\ell-1)\times(\lambda-1)}$
Compute $\widetilde{\mathbf{R}}^\top = \mathbf{E}_\ell \mathbf{S} \in \{0,1\}^{\ell\times(\lambda-1)}$ and $\mathbf{R} = \mathbf{E}_\lambda \widetilde{\mathbf{R}} + \mathbf{r} \cdot \mathbf{e}_\ell^{\ell\,\top} \in \{0,1\}^{\ell \times \lambda}$
Parse $\mathbf{R} = (\mathbf{r}_i)_{i\in[\ell]}$ and $\widetilde{\mathbf{R}} = (\widetilde{\mathbf{r}}_i)_{i\in[\ell]}$
For $i \in [\ell]$, sample $\mathbf{R}'_i \xleftarrow{\$} \{0,1\}^{t_i \times (\lambda-1)}$ and compute $\mathbf{D}_i = \mathbf{R}'_i \mathbf{E}_\lambda^\top$
Compute $\mathbf{C}_i = \mathbf{M}_i \mathbf{R}'_i - \mathbf{x}_i \cdot \widetilde{\mathbf{r}}_i^\top$ for $i \in [\ell-1]$ and $\mathbf{C}_\ell = \mathbf{M}_\ell \mathbf{R}'_\ell - \mathbf{x}_\ell \cdot (\widetilde{\mathbf{r}}_\ell + \widetilde{\mathbf{r}})^\top$
Compute
$$\pi \xleftarrow{\$} \mathsf{Sim}_\lambda(\mathsf{crs}, \mathsf{td}, \mathbf{R}^\top - \mathbf{e}_\ell^\ell \mathbf{r}^\top)$$
Return $\pi_{\mathsf{or}} = ((\mathbf{C}_i, \mathbf{D}_i)_{i\in[\ell]}, \mathbf{R}, \pi)$

Fig. 6. Definition of the simulator $\{\mathsf{ORTGen}_\lambda, \mathsf{ORSim}_\lambda\}_{\lambda\in\mathbb{N}}$ of ORNIZK.

for $(\mathsf{crs}_{\mathsf{or}}, \mathsf{td}_{\mathsf{or}}) \xleftarrow{\$} \mathsf{ORTGen}_\lambda$ and any $((\mathbf{x}_i)_{i\in[\ell]}, \mathbf{w})$ such that $\mathbf{x}_j = \mathbf{M}_j \mathbf{w}$ for some $j \in [\ell]$.

Next we note that for $\mathbf{S} \xleftarrow{\$} \{0,1\}^{(\ell-1)\times(\lambda-1)}$, $\widetilde{\mathbf{R}}^\top = \mathbf{E}_\ell \mathbf{S}$ is uniformly distributed conditioned on $\sum_{i=1}^{\ell} \widetilde{\mathbf{r}}_i = \mathbf{0}$ for $\widetilde{\mathbf{R}}^\top = (\widetilde{\mathbf{r}}_i)_{i\in[\ell]}$. The reason is that $(\widetilde{\mathbf{r}}_i)_{i\in[\ell-1]}$ are randomly distributed (since $\overline{\mathbf{E}}_\ell$ is of full rank) and $\widetilde{\mathbf{r}}_\ell$ is uniquely determined conditioned on $\sum_{i=1}^{\ell} \widetilde{\mathbf{r}}_i = \mathbf{0}$. Thus, for any $\mathbf{r} = \mathbf{E}_\lambda \widetilde{\mathbf{r}}$ where $\widetilde{\mathbf{r}} \in \{0,1\}^{\lambda-1}$, both $\widetilde{\mathbf{R}} + \widetilde{\mathbf{r}} \cdot \mathbf{e}_\ell^{j\,\top}$ and $\widetilde{\mathbf{R}} + \widetilde{\mathbf{r}} \cdot \mathbf{e}_\ell^{\ell\,\top}$ are uniformly distributed conditioned on the sum of the column vectors being $\widetilde{\mathbf{r}}$. In this case, the distributions of $\mathbf{R} = \mathbf{E}_\lambda \widetilde{\mathbf{R}} + \mathbf{r} \cdot \mathbf{e}_\ell^{j\,\top}$ and $\mathbf{R} = \mathbf{E}_\lambda \widetilde{\mathbf{R}} + \mathbf{r} \cdot \mathbf{e}_\ell^{\ell\,\top}$ (generated by $\mathsf{ORProve}_\lambda$ and ORSim_λ respectively) are identical as well. Moreover, we have

$$\mathbf{M}_j \mathbf{R}_j = \mathbf{M}_j (\mathbf{R}_j + \mathbf{w} \cdot \widetilde{\mathbf{r}}_j^\top) - \mathbf{x}_j \cdot \widetilde{\mathbf{r}}_j^\top$$

and

$$(\mathbf{R}_j || \mathbf{w}) \begin{pmatrix} \mathbf{E}_\lambda^\top \\ \widetilde{\mathbf{r}}_j^\top \mathbf{E}_\lambda^\top \end{pmatrix} = (\mathbf{R}_j + \mathbf{w} \cdot \widetilde{\mathbf{r}}_j^\top) \mathbf{E}_\lambda^\top$$

for $\mathbf{x}_j = \mathbf{M}_j \mathbf{w}$. Since the distribution of $\mathbf{R}_j + \mathbf{w} \cdot \widetilde{\mathbf{r}}_j^\top$ for $\mathbf{R}_j \xleftarrow{\$} \{0,1\}^{t_j \times (\lambda-1)}$ is uniform in $\{0,1\}^{t_j\times(\lambda-1)}$, the following distributions are identical.

$$\Pi \xleftarrow{\$} \mathsf{ORProve}_\lambda{}'(\mathsf{crs}_{\mathsf{or}}, (\mathbf{x}_i)_{i\in[\ell]}, \mathbf{w}) \text{ and } \Pi \xleftarrow{\$} \mathsf{ORSim}_\lambda(\mathsf{crs}_{\mathsf{or}}, \mathsf{td}_{\mathsf{or}}, (\mathbf{x}_i)_{i\in[\ell]}),$$

for $(\mathsf{crs}_{\mathsf{or}}, \mathsf{td}_{\mathsf{or}}) \xleftarrow{\$} \mathsf{ORTGen}_\lambda$ and any $((\mathbf{x}_i)_{i\in[\ell]}, \mathbf{w})$ such that $\mathbf{x}_j = \mathbf{M}_j \mathbf{w}$ for some $j \in [\ell]$, completing the proof of composable zero-knowledge.

Perfect Soundness. Due to the perfect soundness of LNIZK, for a valid proof $\pi_{\mathsf{or}} = ((\mathbf{C}_i, \mathbf{D}_i)_{i=0,1}, \mathbf{R}, \pi)$, we have $\mathbf{R}^\top = \mathbf{E}_\ell \mathbf{W} + \mathbf{e}_\ell^\ell \mathbf{r}^\top$ for some $\mathbf{W} \in \{0,1\}^{(\ell-1)\times\lambda}$.

Hence, we have

$$\sum_{i=1}^{\ell} \mathbf{r}_i^\top = {\mathbf{f}_\ell^1}^\top \mathbf{R}^\top = {\mathbf{f}_\ell^1}^\top (\mathbf{E}_\ell \mathbf{W} + \mathbf{e}_\ell^\ell \mathbf{r}^\top) = {\mathbf{f}_\ell^1}^\top \mathbf{e}_\ell^\ell \mathbf{r}^\top = \mathbf{r}^\top.$$

Here, recall that $\mathbf{f}_\ell^1$ denotes a vector in $\{0,1\}^\ell$ consisting only of 1's and $\mathbf{f}_\ell^1 \in \mathrm{Span}(\mathbf{E}_\ell^\top)$. Since we have $\mathbf{r} \notin \mathrm{Span}(\mathbf{E}_\lambda)$ in any CRS generated by Gen_λ, we must have $\mathbf{r}_j \notin \mathrm{Span}(\mathbf{E}_\lambda)$ for some $j \in [\ell]$. For such $j \in [\ell]$, we have $(\mathbf{C}_j || \mathbf{x}_j) \begin{pmatrix} \mathbf{E}_\lambda^\top \\ \mathbf{r}_j^\top \end{pmatrix} = \mathbf{M}_j \mathbf{D}_j$, i.e., $(\mathbf{M}_j^\perp)^\top (\mathbf{C}_j || \mathbf{x}_j) \begin{pmatrix} \mathbf{E}_\lambda^\top \\ \mathbf{r}_j^\top \end{pmatrix} = \mathbf{0}$. Hence, $\mathbf{r}_j(\mathbf{x}_j^\top \mathbf{M}_j^\perp) = \mathbf{E}_\lambda(\mathbf{C}_j^\top \mathbf{M}_j^\perp)$ must hold. Since $\mathbf{r}_j \notin \mathrm{Span}(\mathbf{E}_\lambda)$, we must have $\mathbf{x}_j^\top \mathbf{M}_j^\perp = 0$, which implies $\mathsf{x} \in \mathsf{L}^{\mathsf{or}}_{(\mathbf{M}_i)_{i\in[\ell]}}$, completing the proof of perfect soundness. Notice that this part of arguments is information-theoretical and thus the equations are not necessarily computable in AC^0.

Putting all the above together, Theorem 4 immediately follows. □

Remark on the CRS. When instantiating LNIZK in ORNIZK with our NIZK given in Section 3, both crs and $\mathbf{r}$ in $\mathsf{crs}_{\mathsf{or}}$ are uniformly distributed conditioned on the parities being 1. Hence, we can reduce the length of $\mathsf{crs}_{\mathsf{or}}$ by merging crs and $\mathbf{r}$ in $\mathsf{crs}_{\mathsf{or}}$ as a single vector in $\mathrm{Span}(\mathbf{E}_\lambda)$.

5 NIZK for Circuit SAT

In this section, we propose a fine-grained NIZK for AC^0 circuit SAT running in AC^0 and secure against adversaries in AC^0.

Before giving our construction, we prove the following theorem, which is necessary to show that our NIZK can be executed in AC^0.

Theorem 5. *There exists a family of circuits* $\{\mathsf{ZeroF}_\lambda\}_{\lambda\in\mathbb{N}} \in \mathsf{AC}^0$ *(respectively,* $\{\mathsf{OneF}_\lambda\}_{\lambda\in\mathbb{N}} \in \mathsf{AC}^0$*) such that* ZeroF_λ *(respectively,* OneF_λ*) on input a bit-string* $(b_1, \ldots, b_n)$ *(for some polynomial* $n = n(\lambda)$*) outputs the index* i^* *of the lexicographically first* 0*-bit (respectively,* 1*-bit) of* $(b_i)_{i\in[n]}$.

Proof. We first define ZeroF_λ as in Figure 7.

$\mathsf{ZeroF}_\lambda(b_1, \ldots, b_n)$:
For each $i \in [n]$, we compute $\mathbf{x}_i = \mathbf{i} \cdot (1 - b_i)$ in parallel
For each $i \in [n]$, we compute $\mathbf{y}_i = \mathbf{x}_i \cdot (1 - \mathsf{OR}_{1\le k\le(1-i), 1\le j\le \ell}(x_{k,j}))$
Compute $\mathbf{y}_{i^*} = \mathsf{OR}_{1\le i\le n}(y_{i,1}) \| \ldots \| \mathsf{OR}_{1\le i\le n}(y_{i,\ell})$

Fig. 7. Definition of ZeroF_λ. By $\mathbf{i} \in \{0,1\}^\ell$ we denote the bit-string representing the index i, where we assume that the bit-representation of n has ℓ bits. By $y_{i,j}$ we denote the j-th bit of $\mathbf{y}_i$.

Complexity. The first step can be done by running the NOT and AND gates in parallel with depth 2. The second step can be done by running the NOT, OR, and AND gates in parallel with depth 3. The third step can be done in parallel by running the OR gates with depth 1. Hence, ZeroF_λ can be performed in AC^0 with constant depth 6 by using unbounded fan-in AND, OR, and NOT gates.

Correctness. We now show that ZeroF_λ correctly finds the index of the lexicographically first 0-bit of its input. Via the first step, we can obtain a sequence of strings $(\mathbf{x}_i)_{i\in[n]}$ such that $\mathbf{x}_i = \mathbf{i}$ if $b_i = 0$ and $\mathbf{x}_i = \mathbf{0}$ otherwise. This step is to pick up indices corresponding to 0-bits.

The second step is to cancel all the indices larger than i^*, where i^* is the index of the first 0-bit in $(b_1, \ldots, b_n)$. Specifically, we use the OR gate to compute $\mathbf{y}_i$ such that $\mathbf{y}_i = \mathbf{x}_i$ if all $\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}$ are 0^ℓ, and $\mathbf{y}_i = 0^\ell$ otherwise.

After the second step, we have obtained $(\mathbf{y}_i)_{i\in[n]}$ such that $\mathbf{y}_{i^*} = \mathbf{i}^*$ and $\mathbf{y}_i = \mathbf{0}$ for all $i \neq i^*$, where i^* is the index of the first 0-bit in $(b_1, \ldots, b_n)$. Then we can conclude that the final step outputs each bit of $\mathbf{y}_{i^*} = \mathbf{i}^*$ correctly by using the OR gate.

Construction of OneF_λ. One can see that by generating $\mathbf{x}_i$ as $\mathbf{x}_i = \mathbf{i} \cdot b_i$ instead of $\mathbf{x}_i = \mathbf{i} \cdot (1 - b_i)$, we immediately obtain a circuit OneF_λ running in AC^0 and outputting the first 1-bit of a bit string.

Putting all the above together, Theorem 5 immediately follows. □

An Example for ZeroF_λ. For ease of understanding, we now give an example of the running procedure of ZeroF_λ. Assuming that the string is 10100. In the first step, the circuit outputs $000-010-000-100-101$ by using the NOT and AND gates. In the second step, for each block, the circuit checks wether all its left bits are 0 by using the NOT and OR gates. We can see that the check only works for the block 010. Hence, the circuit now outputs $000-010-000-000-000$. In the third step, the circuit outputs $(\mathsf{OR}(0,0,0,0,0), \mathsf{OR}(0,1,0,0,0), \mathsf{OR}(0,0,0,0,0)) = 010 = 2$, which is exactly the index of the first $b_i = 0$.

Construction of Our NIZK. We now define the following languages

$$\mathsf{L}_\lambda = \{\mathbf{x} : \exists \mathbf{w} \in \{0,1\}^{\lambda-1},\ \text{s.t. } \mathbf{x} = \mathbf{E}_\lambda \mathbf{w}\}$$

and

$$\begin{aligned}\mathsf{L}_\lambda^{\mathsf{or}} = \{(\mathbf{x}_i)_{i\in[\ell]} : &\exists \mathbf{w} \in \{0,1\}^{2\lambda} \ \text{ s.t. } \bigvee_{i\in[\ell]} \mathbf{x}_i = \mathbf{M}_1 \mathbf{w} \\ &\text{or } \exists \mathbf{w} \in \{0,1\}^{(\ell+1)\cdot\lambda} \ \text{ s.t. } \ \mathbf{x}_{(\ell+1)} = \mathbf{M}_2 \mathbf{w}\}\end{aligned}$$

where

$$\mathbf{M}_1 = \begin{pmatrix} \mathbf{E}_\lambda & \mathbf{0} \\ \mathbf{0} & \mathbf{E}_\lambda \end{pmatrix} \in \{0,1\}^{2\lambda \times 2(\lambda-1)}$$

and

$$\mathbf{M}_2 = \begin{pmatrix} \mathbf{E}_\lambda & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{E}_\lambda \end{pmatrix} \in \{0,1\}^{(\ell+1)\cdot\lambda\times(\ell+1)\cdot(\lambda-1)},$$

i.e., $\mathbf{M}_1$ and $\mathbf{M}_2$ contain $\mathbf{E}_\lambda$'s in the main diagonal and $\mathbf{0}$ in the other positions. One can see that $\{\mathsf{L}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\{\mathsf{L}^{\mathsf{or}}_\lambda\}_{\lambda\in\mathbb{N}}$ are supported by our NIZK for linear languages in Section 3 and our OR-proof given in Section 4.2 respectively.

Let $\{\mathsf{L}^{\mathsf{AC}^0}_\lambda\}_{\lambda\in\mathbb{N}}$ be any family of languages such that for all $\mathsf{x} \in \mathsf{L}^{\mathsf{AC}^0}_\lambda$, we can run $\mathsf{R}^{\mathsf{AC}^0}_\lambda(\mathsf{x},\cdot)$ in AC^0, where $\mathsf{R}^{\mathsf{AC}^0}_\lambda(\mathsf{x},\cdot)$ is the associated relation. [6] Without loss of generality, we assume that all the AND and OR gates have fan-in of some polynomial $\ell = \ell(\lambda)$. Let $\mathsf{LNIZK} = \{\mathsf{Gen}_\lambda, \mathsf{Prove}_\lambda, \mathsf{Ver}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathsf{ORNIZK} = \{\mathsf{ORGen}_\lambda, \mathsf{ORProve}_\lambda, \mathsf{ORVer}_\lambda\}_{\lambda\in\mathbb{N}}$ be NIZKs with simulators $\{\mathsf{TGen}_\lambda, \mathsf{Sim}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\{\mathsf{ORTGen}_\lambda, \mathsf{ORSim}_\lambda\}_{\lambda\in\mathbb{N}}$ for $\{\mathsf{L}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\{\mathsf{L}^{\mathsf{or}}_\lambda\}_{\lambda\in\mathbb{N}}$ respectively. We give our NIZK for $\{\mathsf{L}^{\mathsf{AC}^0}_\lambda\}_{\lambda\in\mathbb{N}}$ and its simulator in Figures 8 and 9 respectively.

Theorem 6. *If* LNIZK *and* ORNIZK *are* NC^0*-NIZKs with perfect soundness and* AC^0*-composable zero-knowledge, then* ACNIZK *is an* AC^0*-NIZK with perfect soundness and* AC^0*-composable zero-knowledge.*

Proof. **Complexity.** First, we note that the Hamming weight of each row vector in $\mathbf{E}_\lambda$, $\mathbf{M}_1$, and $\mathbf{M}_2$ is constant. Thus, the multiplication of matrices involved in Figures 8 and 9 and running NIZK and ORNIZK and their simulators can be performed in NC^0. Also, addition of a constant number of matrices can be performed in NC^0, and extending the witness to contain the bits of all wires can be performed in AC^0. Moreover, finding the lexicographically first $j \in [\ell]$ such that $\mathsf{w}_{ij} = 0$ (respectively $\mathsf{w}_{ij} = 1$) for each AND (respectively, OR) gate can also be performed in AC^0 according to Theorem 5. As a result, we have $\{\mathsf{ACGen}_\lambda, \mathsf{ACProve}_\lambda, \mathsf{ACVer}_\lambda, \mathsf{ACTGen}_\lambda, \mathsf{ACSim}_\lambda\}_{\lambda\in\mathbb{N}} \in \mathsf{AC}^0$. Notice that after extending the witness, the prover can generate commitments and run ORNIZK for each wire and gate in parallel and the verifier can check the proofs in parallel.

Completeness. Let $(\mathsf{w}_{i1}, \mathsf{w}_{i2})$ be an input/output pair of a NOT gate and $(\mathsf{cm}_{ib} = \mathbf{E}_\lambda\mathbf{r}_{ib} + \mathbf{t}\mathsf{w}_{ib})_{b\in[2]}$ be the corresponding commitments, we must have

$$\mathsf{cm}_{i1} + \mathsf{cm}_{i2} + \mathbf{t} = \mathbf{E}_\lambda(\mathbf{r}_{i1} + \mathbf{r}_{i2}) + \mathbf{t}(\mathsf{w}_{i1} + \mathsf{w}_{i2} + 1) = \mathbf{E}_\lambda(\mathbf{r}_{i1} + \mathbf{r}_{i2}).$$

Let $((\mathsf{w}_{ij})_{j\in[\ell]}, \mathsf{w}_{i(\ell+1)})$ be a valid input/output pair of an AND or OR gate in the statement circuit and $(\mathsf{cm}_{ij} = \mathbf{E}_\lambda\mathbf{r}_{ij} + \mathbf{t}\mathsf{w}_{ij})_{j\in[\ell+1]}$ be the corresponding commitments.

[6] We can assume that each $\mathsf{R}^{\mathsf{AC}^0}_\lambda(\mathsf{x},\cdot)$ consists only of AND and OR gates, since by De Morgan Rules, we can move all NOT gates to just the inputs and the resulting circuit is still in AC^0. However, this may cause loss on efficiency.

$\underline{\mathsf{ACGen}_\lambda:}$
$\mathsf{crs} \xleftarrow{\$} \mathsf{Gen}_\lambda$, $\mathsf{crs_{or}} \xleftarrow{\$} \mathsf{ORGen}_\lambda$, $\widetilde{\mathbf{r}} \xleftarrow{\$} \{0,1\}^{\lambda-1}$, $\mathbf{t} = \mathbf{E}_\lambda \widetilde{\mathbf{r}} + \mathbf{e}_\lambda^\lambda$
Return $\mathsf{CRS} = (\mathsf{crs}, \mathsf{crs_{or}}, \mathbf{t})$

$\underline{\mathsf{ACProve}_\lambda(\mathsf{CRS}, \mathsf{x}, \mathsf{w}):}$
Extend w to $(\mathsf{w}_1, \cdots, \mathsf{w_{out}})$ containing the bits of all wires in the circuit $\mathsf{R}_\lambda^{\mathsf{AC}^0}(\mathsf{x}, \cdot)$
Compute $\mathbf{r}_i \xleftarrow{\$} \{0,1\}^{\lambda-1}$ and $\mathsf{cm}_i = \mathbf{E}_\lambda \mathbf{r}_i + \mathbf{t}\mathsf{w}_i$ for each bit w_i
Set $\mathbf{r_{out}} = \mathbf{0}$ and $\mathsf{cm_{out}} = \mathbf{e}_\lambda^\lambda$ for the output wire
For each NOT gate with input commitment $\mathsf{cm}_{i1} = \mathbf{E}_\lambda \mathbf{r}_{i1} + \mathbf{t}\mathbf{w}_{i1}$ and output commitment $\mathsf{cm}_{i2} = \mathbf{E}_\lambda \mathbf{r}_{i2} + \mathbf{t}\mathbf{w}_{i2}$, compute $\pi_i \xleftarrow{\$} \mathsf{Prove}_\lambda(\mathsf{crs}, \mathsf{x}_i, \mathbf{r}_{i1} + \mathbf{r}_{i2})$ where $\mathsf{x}_i = \mathsf{cm}_{i1} + \mathsf{cm}_{i2} + \mathbf{t}$
For each AND or OR gate with input commitments $(\mathsf{cm}_{ij} = \mathbf{E}_\lambda \mathbf{r}_{ij} + \mathbf{t}\mathsf{w}_{ij})_{j\in[\ell]}$ and the output commitment $\mathsf{cm}_{i(\ell+1)} = \mathbf{E}_\lambda \mathbf{r}_{i(\ell+1)} + \mathbf{t}\mathsf{w}_{i(\ell+1)}$,

- if the gate is an AND gate,
  - if $\mathsf{w}_{ij} = 1$ for all $j \in [\ell+1]$, set $\mathbf{r} = \mathbf{r}_1 \circ \cdots \circ \mathbf{r}_{\ell+1}$
  - otherwise, find the lexicographically first $j \in [\ell]$ such that $\mathsf{w}_{ij} = 0$ and set $\mathbf{r} = \mathbf{r}_i \circ \mathbf{r}_{\ell+1}$
  - compute $\pi_i \xleftarrow{\$} \mathsf{ORProve}_\lambda(\mathsf{crs_{or}}, (\mathsf{x}_{ij})_{j\in[\ell+1]}, \mathbf{r})$ where $\mathsf{x}_{ij} = \mathsf{cm}_{ij} \circ \mathsf{cm}_{i(\ell+1)}$ for all $j \in [\ell]$ and $\mathsf{x}_{i(\ell+1)} = (\mathsf{cm}_{i1} - \mathbf{t}) \circ \cdots \circ (\mathsf{cm}_{i(\ell+1)} - \mathbf{t})$
- if the gate is an OR gate,
  - if $\mathsf{w}_{ij} = 0$ for all $j \in [\ell+1]$, set $\mathbf{r} = \mathbf{r}_1 \circ \cdots \circ \mathbf{r}_{\ell+1}$
  - otherwise, find the lexicographically first $j \in [\ell]$ such that $\mathsf{w}_{ij} = 1$ and set $\mathbf{r} = \mathbf{r}_i \circ \mathbf{r}_{\ell+1}$
  - compute $\pi_i \xleftarrow{\$} \mathsf{ORProve}_\lambda(\mathsf{crs_{or}}, (\mathsf{x}_{ij})_{j\in[\ell+1]}, \mathbf{r})$ where $\mathsf{x}_{ij} = (\mathsf{cm}_{ij} - \mathbf{t}) \circ (\mathsf{cm}_{i(\ell+1)} - \mathbf{t})$ for all $j \in [\ell]$ and $\mathsf{x}_{i(\ell+1)} = \mathsf{cm}_{i1} \circ \cdots \circ \mathsf{cm}_{i(\ell+1)}$

Return Π consisting of all the commitments and proofs

$\underline{\mathsf{ACVer}_\lambda(\mathsf{CRS}, \mathsf{x}, \Pi):}$
Check that all wires have a corresponding commitment and $\mathsf{cm_{out}} = \mathbf{t}$
Check that all NAND gates have a valid NIZK proof of compliance
Return 1 iff all checks pass

Fig. 8. Definition of $\mathsf{ACNIZK} = \{\mathsf{ACGen}_\lambda, \mathsf{ACProve}_\lambda, \mathsf{ACVer}_\lambda\}_{\lambda\in\mathbb{N}}$. Recall that for any vectors $(\mathbf{x}_i)_{i\in[\ell]}$, by $\mathbf{x}_1 \circ \cdots \circ \mathbf{x}_\ell$ we denote $(\mathbf{x}_1^\top, \cdots, \mathbf{x}_\ell^\top)^\top$.

If the gate is an AND gate, we must have $\mathsf{w}_{ij} = 0 \wedge \mathsf{w}_{i(\ell+1)} = 0$ for some $j \in [\ell]$ or $\mathsf{w}_{ij} = 1$ for all $j \in [\ell+1]$, which implies

$$\mathsf{cm}_{ij} \circ \mathsf{cm}_{i(\ell+1)} = \mathbf{M}_1(\mathbf{r}_{ij} \circ \mathbf{r}_{\ell+1})$$

for some $j \in [\ell]$ or

$$(\mathsf{cm}_{i1} - \mathbf{t}) \circ \cdots \circ (\mathsf{cm}_{i(\ell+1)} - \mathbf{t}) = \mathbf{M}_2(\mathbf{r}_1 \circ \cdots \circ \mathbf{r}_{\ell+1}).$$

If the gate is an OR gate, we must have $\mathsf{w}_{ij} = 1 \wedge \mathsf{w}_{i(\ell+1)} = 1$ for some $i \in [\ell]$ or $\mathsf{w}_{ij} = 0$ for all $j \in [\ell+1]$, which implies

$$(\mathsf{cm}_{ij} - \mathbf{t}) \circ (\mathsf{cm}_{i(\ell+1)} - \mathbf{t}) = \mathbf{M}_1(\mathbf{r}_i \circ \mathbf{r}_{\ell+1})$$

$\underline{\mathsf{ACTGen}_\lambda}$:
$(\mathsf{crs}, \mathsf{td}) \xleftarrow{\$} \mathsf{TGen}_\lambda(\lambda)$, $(\mathsf{crs}_{\mathsf{or}}, \mathsf{td}_{\mathsf{or}}) \xleftarrow{\$} \mathsf{ORTGen}_\lambda(\lambda)$, $\widetilde{\mathbf{r}} \xleftarrow{\$} \{0,1\}^{\lambda-1}$, $\mathbf{t} = \mathbf{E}_\lambda \widetilde{\mathbf{r}}$
Return $\mathsf{CRS} = (\mathsf{crs}, \mathsf{crs}_{\mathsf{or}}, \mathbf{t})$ and $\mathsf{TD} = (\mathsf{td}, \mathsf{td}_{\mathsf{or}})$

$\underline{\mathsf{ACSim}_\lambda(\mathsf{CRS}, \mathsf{TD}, \mathsf{x})}$:
Compute $\mathbf{r}_i \xleftarrow{\$} \{0,1\}^{\lambda-1}$ and $\mathsf{cm}_i = \mathbf{E}_\lambda \mathbf{r}_i$ for each wire in the circuit $\mathsf{R}_\lambda^{\mathsf{AC}^0}(\mathsf{x}, \cdot)$
For each NOT gate with input commitment cm_{i1} and output commitment cm_{i2}, run $\pi_i \xleftarrow{\$} \mathsf{Sim}_\lambda(\mathsf{crs}, \mathsf{td}, \mathsf{x}_i)$ where $\mathsf{x}_i = \mathsf{cm}_{i1} + \mathsf{cm}_{i2} + \mathbf{t}$
For each AND or NOT gate with input commitments $(\mathsf{cm}_{ij})_{j \in [\ell]}$ and the output commitment $\mathsf{cm}_{i(\ell+1)}$, run $\pi_i \xleftarrow{\$} \mathsf{ORSim}_\lambda(\mathsf{crs}_{\mathsf{or}}, \mathsf{td}_{\mathsf{or}}, (\mathsf{x}_{ij})_{j \in [\ell+1]})$, where
 – $\mathsf{x}_{ij} = \mathsf{cm}_{ij} \circ \mathsf{cm}_{i(\ell+1)}$ for all $j \in [\ell]$ and $\mathsf{x}_{\ell+1} = (\mathsf{cm}_{i1} - \mathbf{t}) \circ \cdots \circ (\mathsf{cm}_{i(\ell+1)} - \mathbf{t})$ if the gate is an AND gate
 – $\mathsf{x}_{ij} = (\mathsf{cm}_{ij} - \mathbf{t}) \circ (\mathsf{cm}_{i(\ell+1)} - \mathbf{t})$ for all $j \in [\ell]$ and $\mathsf{x}_{\ell+1} = \mathsf{cm}_{i1} \circ \cdots \circ \mathsf{cm}_{i(\ell+1)}$ if the gate is an OR gate
Return Π consisting of all the commitments and proofs

Fig. 9. Definition of the simulator $\{\mathsf{ACTGen}_\lambda, \mathsf{ACSim}_\lambda\}_{\lambda \in \mathbb{N}}$ of ACNIZK.

for some $i \in [\ell]$ or

$$\mathsf{cm}_{i1} \circ \cdots \circ \mathsf{cm}_{i(\ell+1)} = \mathbf{M}_2(\mathbf{r}_1 \circ \cdots \circ \mathbf{r}_{\ell+1}).$$

Then the completeness of ACNIZK follows from that of LNIZK and that of ORNIZK.

AC^0-Composable Zero-Knowledge. The indistinguishability of CRSs generated by ACGen_λ and ACTGen_λ follows immediately from Lemma 1 and the composable zero-knowledge of LNIZK and ORNIZK.

Next we define a modified prover $\mathsf{ACProve}'_\lambda$, which is exactly the same as $\mathsf{ACProve}_\lambda$ except that for each NOT gate, π_i is generated as

$$\pi_i \xleftarrow{\$} \mathsf{Sim}_\lambda(\mathsf{crs}, \mathsf{td}, \mathsf{x}_i),$$

and for each AND or OR gate, π_i is generated as

$$\pi_i \xleftarrow{\$} \mathsf{ORSim}_\lambda(\mathsf{crs}_{\mathsf{or}}, \mathsf{td}_{\mathsf{or}}, (\mathsf{x}_{ij})_{j \in [\ell+1]}).$$

The following distributions are identical due to the composable zero-knowledge of LNIZK and ORNIZK.

$$\Pi \xleftarrow{\$} \mathsf{ACProve}_\lambda(\mathsf{CRS}, \mathsf{x}, \mathsf{w}) \text{ and } \Pi \xleftarrow{\$} \mathsf{ACProve}'_\lambda(\mathsf{CRS}, \mathsf{x}, \mathsf{w}),$$

for $(\mathsf{CRS}, \mathsf{TD}) \xleftarrow{\$} \mathsf{TGen}_\lambda$ and any (x, w) such that $\mathsf{R}_\lambda^{\mathsf{AC}^0}(\mathsf{x}, \mathsf{w}) = 1$.

Moreover, since the distribution of $\mathsf{cm}_i = \mathbf{E}_\lambda \mathbf{r}_i$ is identical to that of $\mathsf{cm}_i = \mathbf{E}_\lambda \mathbf{r}_i + \mathbf{t}\mathsf{w}_i$ for $\mathbf{r}_i \xleftarrow{\$} \{0,1\}^\lambda$ when $\mathbf{t} \in \mathrm{Span}(\mathbf{E}_\lambda)$, the distributions of

$$\Pi \xleftarrow{\$} \mathsf{ACProve}'_\lambda(\mathsf{CRS}, \mathsf{x}, \mathsf{w}) \text{ and } \Pi \xleftarrow{\$} \mathsf{ACSim}_\lambda(\mathsf{CRS}, \mathsf{TD}, \mathsf{x}),$$

where $(\mathsf{CRS}, \mathsf{TD}) \xleftarrow{\$} \mathsf{ACTGen}_\lambda$ and $\mathsf{R}_\lambda^{\mathsf{AC}^0}(\mathsf{x}, \mathsf{w}) = 1$, are identical as well, completing the proof of composable zero-knowledge.

Perfect Soundness. Due to the perfect soundness of LNIZK and ORNIZK, for each NOT gate with input/output commitments $(\mathsf{cm}_{i0}, \mathsf{cm}_{i1})$, we have $\mathsf{cm}_{i0} + \mathsf{cm}_{i1} = \mathbf{t}$. For each AND gate with input commitments $(\mathsf{cm}_{ij})_{i\in[\ell]}$ and an output commitment $\mathsf{cm}_{i(\ell+1)}$ in a valid proof, we have

$$\mathsf{x}_{ij} = (\mathsf{cm}_{ij} \circ \mathsf{cm}_{i(\ell+1)}) \in \mathrm{Span}(\mathbf{M}_1)$$

for some $j \in [\ell]$ or

$$\mathsf{x}_k = (\mathsf{cm}_{i1} - \mathbf{t}) \circ \cdots \circ (\mathsf{cm}_{i(\ell+1)} - \mathbf{t}) \in \mathrm{Span}(\mathbf{M}_2).$$

Similarly, for each OR gate, we have

$$\mathsf{x}_{ij} = (\mathsf{cm}_{ij} - \mathbf{t} \circ \mathsf{cm}_{i(\ell+1)} - \mathbf{t}) \in \mathrm{Span}(\mathbf{M}_1)$$

for some $j \in [\ell]$ or

$$\mathsf{x}_k = \mathsf{cm}_{i1} \circ \cdots \circ \mathsf{cm}_{i(\ell+1)} \in \mathrm{Span}(\mathbf{M}_2).$$

Recall that $\mathbf{f}_\lambda^1$ denotes a vector in $\{0,1\}^\lambda$ consisting only of 1's and $\mathbf{f}_\lambda^1 \in \mathrm{Ker}(\mathbf{E}_\lambda^\top)$. For $\mathbf{t} = \mathbf{E}_\lambda \widetilde{\mathbf{r}} + \mathbf{e}_\lambda^\lambda$ where $\widetilde{\mathbf{r}} \in \{0,1\}^{\lambda-1}$, we have ${\mathbf{f}_\lambda^1}^\top \mathbf{t} = 1$. For a NOT gate, we must have

$${\mathbf{f}_\lambda^1}^\top \mathsf{cm}_{i1} + {\mathbf{f}_\lambda^1}^\top \mathsf{cm}_{i2} + 1 = 0.$$

For an AND gate, we must have

$${\mathbf{f}_\lambda^1}^\top \mathsf{cm}_{ij} = 0 \text{ and } {\mathbf{f}_\lambda^1}^\top \mathsf{cm}_{i(\ell+1)} = 0 \text{ for some } j \in [\ell]$$

or

$${\mathbf{f}_\lambda^1}^\top \mathsf{cm}_{ij} = 1 \text{ for all } j \in [\ell+1].$$

For an OR gate, we must have

$${\mathbf{f}_\lambda^1}^\top \mathsf{cm}_{ij} = 1 \text{ and } {\mathbf{f}_\lambda^1}^\top \mathsf{cm}_{i(\ell+1)} = 1 \text{ for some } j \in [\ell]$$

or

$${\mathbf{f}_\lambda^1}^\top \mathsf{cm}_{ij} = 0 \text{ for all } j \in [\ell+1].$$

For the output wire, we have

$${\mathbf{f}_\lambda^1}^\top \mathsf{cm}_{\mathsf{out}} = {\mathbf{f}_\lambda^1}^\top \mathbf{t} = 1.$$

As a result, we can extract valid values of all the wires with the final output being 1, completing the proof of perfect soundness. Notice that the extraction procedure is not necessarily in AC^0 since the arguments in this part are information-theoretical.

Putting all the above together, Theorem 6 immediately follows. □

Remark. If we only treat statement circuits in NC^0, we can further reduce the proof size by instantiating the underlying OR-proof with our warm-up construction for one disjunction given in Section 4.1.

Similar to previous fine-grained NIZKs [1,19], our construction also works in the "inefficient prover setting". Namely, if we allow the prover to run in polynomial-time, we immediately have an unconditionally secure NIZK for all NP against AC^0 adversaries.

Extension to NIZK in the URS model. As remarked in Section 4.2, the CRS of the underlying OR-proof can be generated as a single vector uniformly distributed conditioned on the parity being 1. For ACNIZK, we can further merge $\mathsf{crs_{or}}$ and $\mathbf{t}$ in the same way. Moreover, by running ACNIZK in parallel for the same statement and generating each CRS as a uniformly random string, we immediately achieve a NIZK with perfect soundness and composable zero-knowledge in the URS model. The reason is that a random string is a binding and a hiding CRS with "half-half" probability. Composable zero-knowledge of the resulting scheme follows immediately from Lemma 1, and statistical soundness follows from that at least one string is a binding CRS with overwhelming probability.

6 Non-Interactive Zap

In this section, we show that our NIZKs have verifiable correlated key generation and exploit the framework in [10] to convert our NIZKs into non-interactive zaps.

6.1 Verifiable Correlated Key Generation

The definition of verifiable correlated key generation is as follows.

Definition 6 (Verifiable correlated key generation). *A $\mathcal{C}_1$-NIZK* $\mathsf{NIZK} = \{\mathsf{Gen}_\lambda, \mathsf{Prove}_\lambda, \mathsf{Ver}_\lambda\}_{\lambda\in\mathbb{N}}$ *with a simulator* $\{\mathsf{TGen}_\lambda, \mathsf{Sim}_\lambda\}_{\lambda\in\mathbb{N}}$ *has* verifiable correlated key generation *if there exists a function family* $\{\mathsf{Convert}_\lambda, \mathsf{Check}_\lambda\}_{\lambda\in\mathbb{N}} \in \mathcal{C}_1$ *such that*

1. *the distribution of* $\mathsf{Convert}_\lambda(\mathsf{crs}_0)$ *is identical to that of* crs_1*, where* $\mathsf{crs}_0 \xleftarrow{\$} \mathsf{Gen}_\lambda$ *and* $(\mathsf{crs}_1, \mathsf{td}_1) \xleftarrow{\$} \mathsf{TGen}_\lambda$*,*
2. $\mathsf{Check}_\lambda(\mathsf{crs}_0, \mathsf{Convert}_\lambda(\mathsf{crs}_0)) = 1$ *for all* $\mathsf{crs}_0 \in \mathsf{Gen}_\lambda$*, and*
3. *for any* crs_0 *and* crs_1 *(not necessarily in the support of* Gen_λ *or* TGen_λ*) such that* $\mathsf{Check}_\lambda(\mathsf{crs}_0, \mathsf{crs}_1) = 1$*, we have* $\mathsf{crs}_0 \in \mathsf{Gen}_\lambda$ *or* $\mathsf{crs}_1 \in \mathsf{Gen}_\lambda$*.*

Lemma 2. LNIZK *in Section 3 (see Figure 1) has verifiable correlated key generation.*

Proof. For LNIZK where a binding (respectively, hiding) CRS consists only of a vector uniformly sampled conditioned on the parity being 1 (respectively, 0), we define $\{\mathsf{Check}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\{\mathsf{Convert}_\lambda\}_{\lambda\in\mathbb{N}}$ as in Figure 10.

First we note that $\{\mathsf{Convert}_\lambda\}_{\lambda\in\mathbb{N}} \in \mathsf{NC}^0$ and $\{\mathsf{Check}_\lambda\}_{\lambda\in\mathbb{N}} \in \mathsf{NC}^0$ since they only involve addition of two vectors.

$\mathsf{Convert}_\lambda(\mathbf{r}_0)$:	$\mathsf{Check}_\lambda(\mathbf{r}_0, \mathbf{r}_1)$:
$\mathbf{r}_1 = \mathbf{r}_0 + \mathbf{e}_\lambda^\lambda$	Return 1 iff $\mathbf{e}_\lambda^\lambda = \mathbf{r}_0 + \mathbf{r}_1$

Fig. 10. Definition of $\{\mathsf{Convert}_\lambda, \mathsf{Check}_\lambda\}_{\lambda\in\mathbb{N}}$.

For $\mathbf{r}_0 \overset{\$}{\leftarrow} \mathsf{Gen}_\lambda$ and $\mathbf{r}_1 \overset{\$}{\leftarrow} \mathsf{TGen}_\lambda$, the distributions of $\mathbf{r}_0 + \mathbf{e}_\lambda^\lambda$ and $\mathbf{r}_1$ are identical. Hence, the first condition in Definition 6 is satisfied. The second condition is satisfied since for $\mathbf{r}_1 = \mathbf{r}_0 + \mathbf{e}_\lambda^\lambda$, we have $\mathbf{r}_0 + \mathbf{r}_1 = \mathbf{r}_0 + (\mathbf{r}_0 + \mathbf{e}_\lambda^\lambda) = \mathbf{e}_\lambda^\lambda$. For $\mathbf{r}_0$ and $\mathbf{r}_1$ such that $\mathbf{e}_\lambda^\lambda = \mathbf{r}_0 + \mathbf{r}_1$, we must have $\mathsf{PARITY}_\lambda(\mathbf{r}_0) = 1$ or $\mathsf{PARITY}_\lambda(\mathbf{r}_1) = 1$, i.e., $\mathbf{r}_0 \in \mathsf{Gen}_\lambda$ or $\mathbf{r}_1 \in \mathsf{Gen}_\lambda$. Hence, the third condition is also satisfied, completing the proof of Lemma 2. □

As remarked in Sections 4.2 and 5, the CRSs of our OR-proof and our NIZK for circuit SAT can be generated in exactly the same way as those of LNIZK. Hence, we have the following corollary.

Corollary 1. *ORNIZK in Section 4.2 (see Figure 5) and ACNIZK in Section 5 (see Figure 8) have verifiable correlated key generation.*

6.2 Construction of Non-Interactive Zap

We now show how to convert our NIZKs with verifiable correlated key generation to non-interactive zaps by using the technique in [10].

Let $\{\mathsf{L}_\lambda^{\mathsf{AC}^0}\}_{\lambda\in\mathbb{N}}$ be any family of languages such that for all $\lambda \in \mathbb{N}$ and all $\mathsf{x} \in \mathsf{L}_\lambda^{\mathsf{AC}^0}$, we can run $\mathsf{R}_\lambda^{\mathsf{AC}^0}(\mathsf{x}, \cdot)$ in AC^0, where $\mathsf{R}_\lambda^{\mathsf{AC}^0}$ is the associated relation. Let $\mathsf{NIZK} = \{\mathsf{Gen}_\lambda, \mathsf{Prove}_\lambda, \mathsf{Ver}_\lambda\}_{\lambda\in\mathbb{N}}$ be a NIZK with a simulator $\{\mathsf{TGen}_\lambda, \mathsf{Sim}_\lambda\}_{\lambda\in\mathbb{N}}$ and verifiable correlated key converting and checking algorithms $\{\mathsf{Check}_\lambda, \mathsf{Convert}_\lambda\}_{\lambda\in\mathbb{N}}$ for $\{\mathsf{L}_\lambda^{\mathsf{AC}^0}\}_{\lambda\in\mathbb{N}}$. We give a non-interactive zap $\mathsf{ZAP} = \{\mathsf{ZProve}_\lambda, \mathsf{ZVer}_\lambda\}_{\lambda\in\mathbb{N}}$ for $\{\mathsf{L}_\lambda^{\mathsf{AC}^0}\}_{\lambda\in\mathbb{N}}$ in Figure 11.

$\mathsf{ZProve}_\lambda(\mathsf{x}, \mathsf{w})$:	$\mathsf{ZVer}_\lambda(\mathsf{x}, \pi)$:
$(\mathsf{crs}_0, \mathsf{td}_0) \overset{\$}{\leftarrow} \mathsf{TGen}_\lambda$, $\mathsf{crs}_1 = \mathsf{Convert}_\lambda(\mathsf{crs}_0)$	Return 1 iff
$\pi_0 \overset{\$}{\leftarrow} \mathsf{Prove}_\lambda(\mathsf{crs}_0, \mathsf{x}, \mathsf{w})$	$\mathsf{Check}_\lambda(\mathsf{crs}_0, \mathsf{crs}_1) = 1$
$\pi_1 \overset{\$}{\leftarrow} \mathsf{Prove}_\lambda(\mathsf{crs}_1, \mathsf{x}, \mathsf{w})$	$\mathsf{Ver}_\lambda(\mathsf{crs}_0, \mathsf{x}, \pi_0) = 1$
Return $\pi = (\mathsf{crs}_0, \mathsf{crs}_1, \pi_0, \pi_1)$	$\mathsf{Ver}_\lambda(\mathsf{crs}_1, \mathsf{x}, \pi_1) = 1$

Fig. 11. Definition of $\mathsf{ZAP} = \{\mathsf{ZProve}_\lambda, \mathsf{ZVer}_\lambda\}_{\lambda\in\mathbb{N}}$ for $\{\mathsf{L}_\lambda^{\mathsf{AC}^0}\}_{\lambda\in\mathbb{N}}$.

Theorem 7. *If* NIZK *is an* AC^0*-NIZK with* AC^0*-composable zero-knowledge, perfect soundness, and verifiable correlated key generation, then* ZAP *is an* AC^0*-non-interactive zap with perfect soundness and* AC^0*-witness indistinguishability.*

We refer the reader to Appendix A for the security proof.

By instantiating the underlying NIZK with our NIZK in Section 5, we obtain an AC^0-non-interactive zap for AC^0-circuit SAT with AC^0-witness indistinguishability.

Acknowledgement

We are grateful to the anonymous reviewers of ASIACRYPT 2022 for the helpful feedback.

References

1. Ball, M., Dachman-Soled, D., Kulkarni, M.: New techniques for zero-knowledge: Leveraging inefficient provers to reduce assumptions, interaction, and trust. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part III. LNCS, vol. 12172, pp. 674–703. Springer, Heidelberg (Aug 2020) 2, 4, 24
2. Danezis, G., Fournet, C., Groth, J., Kohlweiss, M.: Square span programs with applications to succinct NIZK arguments. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014, Part I. LNCS, vol. 8873, pp. 532–550. Springer, Heidelberg (Dec 2014) 1
3. Degwekar, A., Vaikuntanathan, V., Vasudevan, P.N.: Fine-grained cryptography. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part III. LNCS, vol. 9816, pp. 533–562. Springer, Heidelberg (Aug 2016) 2, 3
4. Dwork, C., Naor, M.: Zaps and their applications. In: 41st FOCS. pp. 283–293. IEEE Computer Society Press (Nov 2000) 3
5. Egashira, S., Wang, Y., Tanaka, K.: Fine-grained cryptography revisited. J. Cryptol. 34(3), 23 (2021) 2
6. Gennaro, R., Gentry, C., Parno, B., Raykova, M.: Quadratic span programs and succinct NIZKs without PCPs. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 626–645. Springer, Heidelberg (May 2013) 1
7. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM Journal on Computing 18(1), 186–208 (1989) 1
8. Groth, J.: Short non-interactive zero-knowledge proofs. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 341–358. Springer, Heidelberg (Dec 2010) 1
9. Groth, J.: On the size of pairing-based non-interactive arguments. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 305–326. Springer, Heidelberg (May 2016) 1
10. Groth, J., Ostrovsky, R., Sahai, A.: New techniques for noninteractive zero-knowledge. J. ACM 59(3), 11:1–11:35 (2012) 1, 3, 5, 7, 24, 25
11. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (Apr 2008) 1, 3
12. Håstad, J.: One-way permutations in NC0. Inf. Process. Lett. 26(3), 153–155 (1987) 3
13. Håstad, J.: On the correlation of parity and small-depth circuits. SIAM J. Comput. 43(5), 1699–1708 (2014) 4, 8

14. Impagliazzo, R.: A personal view of average-case complexity. In: Computational Complexity Conference. pp. 134–147. IEEE Computer Society (1995) 3
15. Impagliazzo, R., Matthews, W., Paturi, R.: A satisfiability algorithm for ac$ˆ0$. CoRR abs/1107.3127 (2011) 4, 8
16. Lipmaa, H.: Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 169–189. Springer, Heidelberg (Mar 2012) 1
17. Peikert, C., Shiehian, S.: Noninteractive zero knowledge for NP from (plain) learning with errors. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 89–114. Springer, Heidelberg (Aug 2019) 1
18. Ràfols, C.: Stretching groth-sahai: NIZK proofs of partial satisfiability. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 247–276. Springer, Heidelberg (Mar 2015) 5
19. Wang, Y., Pan, J.: Non-interactive zero-knowledge proofs with fine-grained security. In: Eurocrypt 2022, Trondheim, Norway, 2022, Proceedings. Lecture Notes in Computer Science (2022), https://eprint.iacr.org/2022/548 2, 3, 4, 24, 29
20. Wang, Y., Pan, J., Chen, Y.: Fine-grained secure attribute-based encryption. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part IV. LNCS, vol. 12828, pp. 179–207. Springer, Heidelberg, Virtual Event (Aug 2021) 2

Appendix

A Proof of Theorem 7

We prove Theorem 7 in this section.

Proof. **Complexity.** our ZAP runs in AC^0, since the underlying NIZK runs in AC^0.

Completeness. The completeness of ZAP follows immediately from that of NIZK and the fact that $\mathsf{Check}_\lambda(\mathsf{crs}_0, \mathsf{Convert}_\lambda(\mathsf{crs}_0)) = 1$ for all $\mathsf{crs}_0 \in \mathsf{Gen}_\lambda$ (see Definition 6).

Perfect Soundness. Due to the verifiable correlated key generation of NIZK, we have $\mathsf{crs}_0 \in \mathsf{Gen}_\lambda$ or $\mathsf{crs}_1 \in \mathsf{Gen}_\lambda$ for a valid proof $\pi = (\mathsf{crs}_0, \mathsf{crs}_1, \pi_0, \pi_1)$. Hence, the perfect soundness of ZAP follows immediately from that of NIZK.

AC^0-Witness Indistinguishability. We prove the witness indistinguishability of ZAP by a sequence of games as in Figure 12.

$\mathsf{ZProve}_\lambda(\mathsf{x}, \mathsf{w}_0)$, $\mathsf{ZProve}_\lambda(\mathsf{x}, \mathsf{w}_0, \mathsf{w}_1)$, $\mathsf{ZProve}_\lambda(\mathsf{x}, \mathsf{w}_1)$:

G_0, G_1, G_2, G_3, G_4

$(\mathsf{crs}_0, \mathsf{td}_0) \xleftarrow{\$} \mathsf{TGen}_\lambda$, $\mathsf{crs}_0 \xleftarrow{\$} \mathsf{Gen}_\lambda$, $(\mathsf{crs}_0, \mathsf{td}_0) \xleftarrow{\$} \mathsf{TGen}_\lambda$

$\mathsf{crs}_1 = \mathsf{Convert}_\lambda(\mathsf{crs}_0)$

$\pi_0 \xleftarrow{\$} \mathsf{Prove}_\lambda(\mathsf{crs}_0, \mathsf{x}, \mathsf{w}_0)$, $\pi_0 \xleftarrow{\$} \mathsf{Prove}_\lambda(\mathsf{crs}_0, \mathsf{x}, \mathsf{w}_1)$

$\pi_1 \xleftarrow{\$} \mathsf{Prove}_\lambda(\mathsf{crs}_1, \mathsf{x}, \mathsf{w}_0)$, $\pi_1 \xleftarrow{\$} \mathsf{Prove}_\lambda(\mathsf{crs}_1, \mathsf{x}, \mathsf{w}_1)$

Return $\pi = (\mathsf{crs}_0, \mathsf{crs}_1, \pi_0, \pi_1)$

Fig. 12. Modifications on ZProve_λ in the intermediate games.

Let $\mathcal{A} = \{a_\lambda\}_{\lambda \in \mathbb{N}} \in \mathsf{AC}^0$ be an adversary against the witness indistinguishability of ZAP. It receives a proof π generated by the (modified) prover in each game as defined in Figure 12. Below by ε_i we denote the probability that a_λ outputs 1 in Game G_i for $i = 0, \cdots, 4$.

Games G_0 and G_1. G_0 is the real game where a_λ receives $\pi = (\mathsf{crs}_0, \mathsf{crs}_1, \pi_0, \pi_1) \xleftarrow{\$} \mathsf{ZProve}_\lambda(\mathsf{x}, \mathsf{w}_0)$. G_1 is the same as G_0 except that π_0 is generated as $\pi_0 \xleftarrow{\$} \mathsf{Prove}_\lambda(\mathsf{crs}_0, \mathsf{x}, \mathsf{w}_1)$ instead of $\pi_0 \xleftarrow{\$} \mathsf{Prove}_\lambda(\mathsf{crs}_0, \mathsf{x}, \mathsf{w}_0)$.

Lemma 3. $\varepsilon_0 = \varepsilon_1$.

Proof. Lemma 3 follows immediately from the composable zero knowledge of NIZK. □

Game G_2. This is the same as G_1 except that crs_0 is generated as $\mathsf{crs}_0 \xleftarrow{\$} \mathsf{Gen}_\lambda$ instead of $(\mathsf{crs}_0, \mathsf{td}_0) \xleftarrow{\$} \mathsf{TGen}_\lambda$.

Lemma 4. *There exists an adversary* $\mathcal{B}_1 = \{b^1_\lambda\}_{\lambda\in\mathbb{N}} \in \mathsf{AC}^0$ *such that* b^1_λ *breaks the composable zero-knowledge of* NIZK *with probability* $|\varepsilon_2 - \varepsilon_1|$.

Proof. We build the distinguisher b^1_λ as follows.

b^1_λ runs as in G_1 except that now it takes crs_0 as input from the composable zero-knowledge game of NIZK. crs_0 can be generated as $(\mathsf{crs}_0, \mathsf{td}_0) \xleftarrow{\$} \mathsf{TGen}_\lambda$ or $\mathsf{crs}_0 \xleftarrow{\$} \mathsf{Gen}_\lambda$. When a_λ outputs $\beta \in \{0, 1\}$, b^1_λ outputs β as well.

If crs_0 is generated as $(\mathsf{crs}_0, \mathsf{td}_0) \xleftarrow{\$} \mathsf{TGen}_\lambda$ (respectively, $\mathsf{crs}_0 \xleftarrow{\$} \mathsf{Gen}_\lambda$), the view of a_λ is the same as its view in G_1 (respectively, G_2). Hence, the probability that b^1_λ breaks the fine-grained matrix linear assumption is $|\varepsilon_2 - \varepsilon_1|$.

Moreover, since all operations in b^1_λ are performed in AC^0, we have $\mathcal{B}_1 = \{b^1_\lambda\}_{\lambda\in\mathbb{N}} \in \mathsf{AC}^0$, completing this part of proof. ☐

Game G_3. G_3 is the same as G_2 except that π_1 is generated as $\pi_1 \xleftarrow{\$} \mathsf{Prove}_\lambda(\mathsf{crs}_1, \mathsf{x}, \mathsf{w}_1)$ instead of $\pi_1 \xleftarrow{\$} \mathsf{Prove}_\lambda(\mathsf{crs}_1, \mathsf{x}, \mathsf{w}_0)$.

Lemma 5. $\varepsilon_3 = \varepsilon_2$.

Proof. By the verifiable correlated key generation, the distribution of $\mathsf{Convert}_\lambda(\mathsf{crs}_0)$ is the same as crs_1 for $\mathsf{crs}_0 \xleftarrow{\$} \mathsf{Gen}_\lambda$ and $(\mathsf{crs}_1, \mathsf{td}_1) \xleftarrow{\$} \mathsf{TGen}_\lambda$. Then Lemma 5 follows from the composable zero-knowledge of NIZK. ☐

Game G_4. G_4 is the same as G_3 except that crs_0 is generated as $(\mathsf{crs}_0, \mathsf{td}_0) \xleftarrow{\$} \mathsf{TGen}_\lambda$ instead of $\mathsf{crs}_0 \xleftarrow{\$} \mathsf{Gen}_\lambda$.

Lemma 6. *There exists an adversary* $\mathcal{B}_2 = \{b^2_\lambda\}_{\lambda\in\mathbb{N}} \in \mathsf{AC}^0$ *such that* b^2_λ *breaks the composable zero-knowledge of* NIZK *with probability* $|\varepsilon_4 - \varepsilon_3|$.

Proof. We build the distinguisher b^2_λ as follows.

b^2_λ runs as in G_3 except that crs_0 is taken as input from its composable zero-knowledge challenger, namely, crs_0 can be generated as $\mathsf{crs}_0 \xleftarrow{\$} \mathsf{Gen}_\lambda$ or $(\mathsf{crs}_0, \mathsf{td}_0) \xleftarrow{\$} \mathsf{TGen}_\lambda$. When a_λ outputs $\beta \in \{0, 1\}$, b^2_λ outputs β as well.

If crs_0 is generated as $\mathsf{crs}_0 \xleftarrow{\$} \mathsf{Gen}_\lambda$ (respectively, $(\mathsf{crs}_0, \mathsf{td}_0) \xleftarrow{\$} \mathsf{TGen}_\lambda$), the view of a_λ is the same as its view in G_3 (respectively, G_4). Hence, the probability that b^2_λ breaks the composable zero-knowledge of NIZK is $|\varepsilon_4 - \varepsilon_3|$.

Moreover, since all operations in b^2_λ are performed in AC^0, we have $\mathcal{B}_2 = \{b^2_\lambda\}_{\lambda\in\mathbb{N}} \in \mathsf{AC}^0$, and this completes the proof. ☐

Putting all the above together, Theorem 7 immediately follows. ☐

Remark on Non-Interactive Zap for NP. Similar to the work of Wang and Pan [19], our transformation from NIZK to the non-interactive zap also works for polynomial-time provers, namely, we have an unconditionally secure non-interactive zap for all NP against AC^0 adversaries if we allow polynomial-time provers. In our transformation, generating a zap proof (see Figure 11) involves two proofs of the underlying NIZK. In this case, we have to show that the above reductions run in AC^0, i.e., we need to ensure that they can generate proofs of the underlying NIZK in AC^0. This is possible for our NIZK in Figure 8. More precisely, to generate a NIZK proof for an NP statement, AC^0-reductions can perform all the steps except for extending the witness (since the commitments and OR-proofs can be generated in parallel). Extending the witness is not necessary, since the extended witness can be hard-wired in an AC^0-reduction beforehand, due to the fact that any statement x and its two witnesses w_0 and w_1 are a-prior fixed in the hybrid games.