A Simple and Efficient Framework of Proof Systems for NP

Yuyu Wang[1], Chuanjie Su[1], Jiaxin Pan[2,3], and Yu Chen[4,5,6](✉)

[1] University of Electronic Science and Technology of China, Chengdu, China
wangyuyu@uestc.edu.cn, chuanjie.su@hotmail.com
[2] University of Kassel, Kassel, Germany
[3] Department of Mathematical Sciences, NTNU - Norwegian University of Science and Technology, Trondheim, Norway
jiaxin.pan@ntnu.no
[4] School of Cyber Science and Technology, Shandong University, Qingdao 266237, China
[5] State Key Laboratory of Cryptology, P.O. Box 5159, Beijing 100878, China
[6] Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Qingdao 266237, China
yuchen@sdu.edu.cn

Abstract. In this work, we propose a simple framework of constructing efficient non-interactive zero-knowledge proof (NIZK) systems for all NP. Compared to the state-of-the-art construction by Groth, Ostrovsky, and Sahai (J. ACM, 2012), our resulting NIZK system reduces the proof size and proving and verification cost without any trade-off, i.e., neither increasing computation cost, CRS size nor resorting to stronger assumptions.

Furthermore, we extend our framework to construct a batch argument (BARG) system for all NP. Our construction remarkably improves the efficiency of BARG by Waters and Wu (Crypto 2022) without any trade-off.

Keywords: Non-interactive zero-knowledge · batch argument · NP language · pairing-based cryptography

1 Introduction

1.1 Motivation

Zero-knowledge proof systems introduced by Goldwasser, Micali, and Rackoff [25] allow a prover to convince a verifier for the validity of an NP statement, without revealing anything beyond that. As its round-optimal variant, non-interactive zero-knowledge proof (NIZK) allows a prover to convince the verifier by sending out a single message. Due to this nice property, NIZK is a very interesting topic in both practical and theoretical cryptography, and it has been used as an important building block for countless cryptographic primitives and protocols.

NIZKs for all NP were firstly proposed in [7,19] based on the quadratic residuosity assumption and the existence of trapdoor permutation. While these results demonstrate the feasibility of NIZKs for NP under standard assumptions, they are not very efficient.

For better efficiency, Groth, Ostrovsky, and Sahai [29] proposed a framework of efficient pairing-based constructions (GOS-NIZK). It provides constructions with prime-order and composite-order pairings. Their security is based on the Decisional Linear (DLIN) assumption and the subgroup decision assumption, depending on which pairing group they are constructed in. Their constructions have tight security and compact common reference strings (CRS), namely, a CRS contains a constant number of group elements. Moreover, they achieve perfect soundness and computationally zero-knowledge, or computational soundness and perfect zero-knowledge, depending on the CRS. This is referred to as the dual-mode property. Perfect soundness and perfect zero-knowledge can provide "everlasting security" and are interesting for certain applications. For instance, a NIZK with perfect soundness always rejects an invalid proof and can protect messages that are valuable for a limited time and can be published or deleted later. Additionally, the dual mode with perfect zero-knowledge continuously protects secrets and prevents adversaries from breaking soundness at the time where an honest proof is generated, thereby ensuring security by letting the system reject users who have timed out when attempting to generate proofs.

To further improve the efficiency of GOS-NIZK, a sequence of works have been proposed. These works either restrict to algebraic languages (e.g., [15,30,33,34,39]) or base their security on non-falsifiable assumptions (e.g., [17,23,26,28,40]). In this paper, we are interested in constructions based on standard assumptions in pairing groups. By "standard assumptions", we refer to static and falsifiable assumptions, such as the (Matrix) Diffie-Hellman assumption [18]. The reason of using standard assumptions is that they are well-studied and provide more reliable security guarantee. Recently, a notable work by Katsumata et al. [37,38] (KNYY) shortened the proof size of pairing-based NIZKs for all NP based on standard assumptions. Their proof size is additive, e.g., $s + \mathsf{poly}(\lambda)$ rather than $s \cdot \mathsf{poly}(\lambda)$ as in GOS-NIZK, where s is the number of gates. However, its security is based on a particular computational Diffie-Hellman assumption, CDH^*, which is the CDH assumption in a subgroup of $\mathbb{Z}_p^*$ (where prime p is the group order).[1] As noted by the authors themselves, it is unclear how to instantiate their construction under the standard CDH assumption in a pairing group. Also, their construction suffers from non-compact CRSs and significant security loss, and lacks both perfect zero-knowledge and perfect soundness.

Summing up the above discussion, we ask the following question:

Is it possible to improve the efficiency of GOS-NIZK without any trade-off?

Non-interactive batch arguments. Another line of research focuses on non-interactive batch arguments (BARG), which are sound proof systems amortizing

[1] This particular assumption was not included in their proceeding version [37], but in their later full version [38].

Table 1. Comparison of pairing-based NIZKs for all NP under standard assumptions. GOS12 is the original GOS-NIZK which is only with symmetric pairings, and GOS12* is its variant with asymmetric pairings (see Appendix A). t and s are the number of wires and gates in the statement circuit respectively. In column "**Sound.**" (respectively, "**ZK**"), comp. and perf. mean computation and perfect soundness (respectively, zero-knowledge) respectively. In columns "**Prov. Cost**", "**Ver. Cost**" we measure the numbers of exponentiations and pairings for the proving and verification cost respectively (since they dominate the overall performance of proving and verification). "**Assump.**" means the underlying assumption.

Scheme	**Sound.**	**ZK**	**CRS Size**	**Proof Size**	**Prov. Cost**	**Ver. Cost**	**Assump.**
GOS12 [29] (sym. pair.)	comp. perf.	perf. comp.	$5\|\mathbb{G}\|$	$(9t+6s)\|\mathbb{G}\|$	$15t+12s$	$18(s+t)$	DLIN
GOS12* (asym. pair.)	comp. perf.	perf. comp.	$4\|\mathbb{G}_1\|+4\|\mathbb{G}_2\|$	$(6t+4s)\|\mathbb{G}_1\|+(6t+6s)\|\mathbb{G}_2\|$	$18t+16s$	$12(s+t)$	SXDH
Ours	comp. perf.	perf. comp.	$4\|\mathbb{G}_1\|+4\|\mathbb{G}_2\|$	$(2t+8s)\|\mathbb{G}_1\|+10s\|\mathbb{G}_2\|$	$2t+30s$	$24s$	SXDH

the cost of verification across multiple statements. Specifically, a BARG allows a prover to generate a proof of the validity of multiple statements where the proof size scales sublinearly with the number of statements.

Up until now most works are devoted to constructions in idealized models [2,11,12,28,42,46] or non-standard assumptions [3–6,8,16,23,27,36,41,44]. Recently, Choudhuri et al. [13,14] proposed a construction under both quadratic residue and the subexponentially hard Diffie-Hellman assumption and a construction under the learning with errors assumption. Subsequently, a breakthrough work by Waters and Wu [48] proposed the first BARG (WW-BARG) for all NP over prime-order bilinear maps under the matrix Diffie-Hellman (MDDH) assumption [18]. They also gave a composite-order group version under the subgroup decision assumption. The proof sizes of both constructions are independent of the number of statements. As applications of WW-BARG, they proposed the first succinct non-interactive argument (SNARG) for P with sublinear-sized CRS and the first aggregated signature from standard assumptions over bilinear maps. A recent work by Kalai et al. [35] shows a bootstrapping technique that can generally convert BARGs into ones where CRSs grow polylogarithmically with the number of statements. As a trade-off, the proof sizes grow polylogarithmically as well.

Due to the versatility of BARGs over bilinear maps, it is natural to ask the same question on GOS-NIZK mentioned above for the state-of-the-art BARG by Waters and Wu, i.e. whether we can improve its efficiency without any trade-off. Such an improvement will immediately yields a more efficient BARGs with short CRSs via the bootstrapping technique by Kalai et al. [35].

1.2 Our Contributions

Improvement on GOS-NIZK without Trade-Off. In this work, we improve the efficiency of GOS-NIZK with asymmetric Type-3 pairings by proposing a new and simple framework of constructing efficient NIZKs for NP. We consider Type-3 pairings, since it is the most efficient one among all different types of pairings [20]. Moreover, cryptanalysis [31,32] against symmetric pairing groups with small characteristic curves motivate cryptographic schemes in Type-3 pairings (for instance, [1,10]).

We note that the original GOS-NIZK was proposed with symmetric pairings under the decisional linear (DLIN) assumption. For a fair comparison with our scheme, we give its variant in the asymmetric pairing explicitly in Appendix A. In the rest of this section, we refer GOS-NIZK to be the one in the Type-3 setting, unless stated otherwise.

By instantiating our scheme based on the SXDH assumption, our resulting NIZK proofs consist of $2t + 8s$ group elements in $\mathbb{G}_1$ and $10s$ elements in $\mathbb{G}_2$, where t and s respectively denote the numbers of wires and gates of the statement circuit (the statement represented by fan-in-2 and unbounded fan-out NAND gates). We denote this as $(2t + 8s, 10s)$. Notice that for each multiple fan-out gate, we only increment the count of wires t by 1 for its output wires, since all output wires of the gate are assigned the same value and serves as input wires for multiple other gates. For proving and verification, we use $2t + 30s$ exponentiations and $24s$ pairings respectively. We note that any circuit can be converted to one with only NAND gates.[2] For GOS-NIZK, its proof size, proving cost, and verification cost are $(6t + 4s, 6t + 6s)$, $18t + 16s$ exponentiations, and $12(t+s)$ pairings respectively, which are strictly larger than ours. This is because t is larger (or even much larger in many cases) than s, since each gate has at least 1 output wire. Indeed, $t-s$ corresponds to the number of input wires (with no gates outputting them) and it cannot be very small. Otherwise, the witness will be very short and an adversary can guess it with large probability. As an instance, for a statement circuit consisting only of fan-in-2 and fan-out-1 gates, we have $t = 2s$ (without counting the final ouput wire) since each time when adding a gate from the bottom to the top in a circuit, s and t increase by 1 and 2 respectively. In this case, our scheme uses $(12s, 10s)$ group elements in the proof, $34s$ exponentiations for proving, and $24s$ pairings for verification, which is much more efficient than GOS-NIZK using $(16s, 18s)$ elements in a proof, $52s$ exponentiations for proving, and $36s$ pairings for verification.

[2] Notice that this conversion does not affect the fairness of the comparison. The reason is that in an original statement circuit consisting of AND, OR, and NOT gates, each AND or OR gate can be represented as the combination of one NAND gate and several NOT gates, while NOT gates are "free" in the sense that they do not increase the proof size in both the GOS-NIZK and ours. Indeed, converting the statement circuits into ones consisting only of NAND gates is unnecessary in practice. This conversion is just used for conceptual simplicity, as we will discuss at the end of Sect. 3. The same arguments can be made for the BARGs mentioned later.

In Table 1, we give comparison of the security quality, CRS size, proof size, proving and verification cost, and the underlying assumptions of our NIZK and the ones by GOS. For being general, we present the schemes and proofs in the technical part with the MDDH assumption [18], which is an algebraic generalization of the DLIN and SXDH assumptions. The security of all the instantiations are tight. For the experimental results on the cost and proof size, which are consistent with our comparison in Table 1, we refer the reader to Sect. 5.

Table 2. Comparison of pairing-based BARGs for all NP. WW22 is WW-BARG in the asymmetric pairing, and WW22* is its symmetric pairing version. m denotes the number of statement instances. t and s denote the number of wires and gates in the relation circuit respectively. We assume that all provers take as input m statements. All the instantiations satisfy somewhere argument of knowledge. In columns "**Prov. Cost**", "**Ver. Cost**", we measure the numbers of multiplications and pairings for the proving and verification cost respectively (since they dominate the overall performance of proving and verification). "**Assump.**" means the underlying assumption.

Scheme	CRS Size	Proof Size	Prov. Cost	Ver. Cost	Assump.
WW22 [48] (asym. pair.)	$(4+2m^2)\lvert\mathbb{G}_1\rvert+(4+2m^2)\lvert\mathbb{G}_2\rvert$	$(4t+4s)\lvert\mathbb{G}_1\rvert+(4t+4s)\lvert\mathbb{G}_2\rvert$	$4m^2t+4m(m-1)s$	$24t+32s$	SXDH
WW22* [48] (sym. pair.)	$(1+m^2)\lvert\mathbb{G}\rvert$	$(2t+s)\lvert\mathbb{G}\rvert$	$m^2t+\frac{m(m-1)}{2}s$	$2t+3s$	Subgroup decision
Ours (asym. pair.)	$(4+2m^2)\lvert\mathbb{G}_1\rvert+(4+2m^2)\lvert\mathbb{G}_2\rvert$	$(2t+6s)\lvert\mathbb{G}_1\rvert+(2t+6s)\lvert\mathbb{G}_2\rvert$	$4mt+6m(m-1)s$	$40s$	SXDH
Ours (sym. pair.)	$(1+m^2)\lvert\mathbb{G}\rvert$	$(t+2s)\lvert\mathbb{G}\rvert$	$mt+m(m-1)s$	$4s$	Subgroup decision

Given that our construction improves the proving and verification costs of state-of-the-art constructions without any trade-off, it is recommended that any applications of GOS-NIZK utilize our construction as a drop-in replacement. Shorter proofs are always better, particularly in distributed settings. In such scenarios, proofs may need to be stored permanently and can significantly impact bandwidth usage. Therefore, even a constant rate of communication cost holds significant importance. This is exemplified by ZKB++ [9], which successfully reduces the proof size of ZKBoo [24] in the random oracle model by a factor of 2. Also, similar to GOS-NIZK, via the generic construction in [29], our NIZK can be converted into a (more efficient) non-interactive zap, which has witness-indistinguishability and uses no CRS. As far as we know, this is the most efficient non-interactive zap based on standard assumptions by now. It provides perfect subversion-resistance and is important for distributed systems where trusted CRS is not desirable. Moreover, it can be converted into a leakage-resilient NIZK via the generic construction by Garg-Jain-Sahai [21], which in turn implies a (more efficient) fully leakage-resilient signature.

Extension to BARG. We further extend our framework to improve the efficiency of WW-BARG without making compromises. Similar to our NIZK, we present our BARG with the MDDH assumptions. Under the SXDH assumption, we obtain a BARG with each proof consisting of $(6s + 2t, 6s + 2t)$ elements. It is shorter than that in WW-BARG with $(4s + 4t, 4s + 4t)$ elements. Transplanting our BARG into composite-order bilinear groups derives a BARG with the proof size $2s + t$, while the proof size of the composite-order construction by Waters and Wu is $2t + s$. Moreover, our proving and verifying costs are less than WW-BARG in both the prime-order and composite-order groups.

In Table 2, we give comparison of our constructions and the ones by Waters and Wu. All the instantiations in the table satisfy the (tight) security of somewhere extractability argument of knowledge (see Definition 7), which in turn implies non-adaptive soundness, namely, soundness for statements independent of the CRS. For the experimental results on the cost and proof size, which are consistent with our comparison in Table 2, we refer the reader to Sect. 5.

Similar to our NIZK construction, we recommend using our BARG construction as a drop-in replacement for WW-BARG in any of its applications. For instance, it provides the most efficient SNARG for P with optimal succinctness on CRS and proof sizes, through conversions by Waters-Wu and Kalai et al. [35].

1.3 Technical Overview

Let $\mathsf{C}(\mathsf{x}, \cdot)$ be a statement circuit represented by NAND gates, where x is the statement hardwired in C. We briefly recall that, in the GOS-NIZK, to prove the existence of a witness w such that $\mathsf{C}(\mathsf{x}, \mathsf{w}) = 1$, a prover first extends the witness to contain the bits for all wires of $\mathsf{C}(\mathsf{x}, \cdot)$. Then it hides all bits in w with an additively homomorphic commitment and makes the commitment for the final output wire a fixed one corresponding to 1. In this way anyone can check it. Since for each gate $G_\ell = (d_1, d_2, d_3)$, $((\mathsf{w}_{d_1}, \mathsf{w}_{d_2}), \mathsf{w}_{d_3})$ is a valid input/output tuple if and only if

$$\mathsf{w}_{d_1} + \mathsf{w}_{d_2} + 2\mathsf{w}_{d_3} - 2 \in \{0, 1\}. \quad (1)$$

Here by $G_\ell = (d_1, d_2, d_3)$ we mean that the left and right input wires of the gate G_ℓ are indexed as d_1 and d_2 respectively, while the output wire of G_ℓ is indexed as d_3. The prover can use an OR-proof system to prove that the plaintexts of all the commitments satisfy such a relation. Additionally, the prover has to prove the validity of each wire, namely, each commitment commits to a bit (rather than some other value).

Our approach of NIZK for all NP. In our construction, we also commit to the value of each wire and prove that the committed values are valid for each gate. Different to the GOS-NIZK, we adopt the following consistency relation to improve the efficiency:

$$(-1 + \mathsf{w}_{d_1} + \mathsf{w}_{d_3} = 0 \wedge -1 + \mathsf{w}_{d_2} = 0) \ \vee \ (-1 + \mathsf{w}_{d_3} = 0 \wedge \mathsf{w}_{d_2} = 0). \quad (2)$$

One can check that when the computations are over $GF(2)$, Relation (2) holds if and only if the input/output pair is binary. Only proving this relation of committed values for each gate can be done by using a simple OR-proof, and this indeed yields shorter proof size in total, compared with the GOS-NIZK. However, when considering a large field, only satisfying this relation may seem meaningless. Specifically, when $\mathsf{w}_{d_2} = 1$, w_{d_1} and w_{d_3} might be large numbers with sum "happening to be" 1, and when $\mathsf{w}_{d_2} = 0$, the situation seems worse: there is no restriction on w_{d_1} at all. Hence, without proving the wires are binary, a valid proof for such a relation does not necessarily mean the validity of a statement. A natural approach is to additionally generate proofs of wire validity for $\mathsf{w}_{d_1}, \mathsf{w}_{d_3} \in \{0,1\}$. However, this results in longer proofs than the GOS-NIZK. To overcome this, we develop a new method for soundness without additional wire validity checking procedure.

A new witness-extraction strategy. To maintain both security and efficiency, we propose a new *witness-extraction strategy* for proving soundness, which does not require additional wire validity checks when adopting Relation (2). Specifically, this strategy helps us extract a witness from any valid proof only proving that committed values satisfy Relation (2) for each gate. The strategy uses two phases.

In the first phase, given a valid proof, we use a trapdoor to decrypt all commitments. The decryption result for the final output wire must be 1, and those for other wires could be any value (not necessarily in $\{0,1\}$). The soundness of the underlying OR-proof system only guarantees that all the decryption results satisfy Relation (2) for each gate.

In the next phase, we start to pick up useful values from the decryption results. This procedure starts from the final output wire to the input wires. Let $((\mathsf{w}_{d_1}, \mathsf{w}_{d_2}), \mathsf{w}_{d_3})$ be the decryption results for the final gate G_t. We must have $\mathsf{w}_{d_3} = 1$, and $\mathsf{w}_{d_2} \in \{0,1\}$ according to Relation (2). If $\mathsf{w}_{d_2} = 1$, $\mathsf{w}_{d_1} = 0$ must hold, and we set $((\mathsf{w}_{d_1}, \mathsf{w}_{d_2}), \mathsf{w}_{d_3})$ as the input/output values for G_t. The problem is that when $\mathsf{w}_{d_2} = 0$, w_{d_1} could be any large value. Our trick is not to assign any number to this wire and leave it blank for now. The point is that no matter which in $\{0,1\}$ will be assigned to the left-input wire, as long as $\mathsf{w}_{d_2} = 0$ and $\mathsf{w}_{d_3} = 1$, $((\mathsf{w}_{d_1}, \mathsf{w}_{d_2}), \mathsf{w}_{d_3})$ will be a valid pair for G_t. Next, for each gate where we have previously assigned a value in $\{0,1\}$ to its output wire, we assign values to its input-wire(s) in a similar way. By doing this recursively from the bottom to the top of the circuit, we eventually obtain values for part of the input wires of the whole statement circuit. Now notice that these values will lead the circuit to output 1 anyway, no matter what the rest of the input wires (left as blank) will be. By setting these rest of the input wires as, say 0, we obtain a value witness.

For better understanding, we give an example of the witness-extraction procedure for the statement circuit in Fig. 1. In the decryption result of a valid proof, the final output must be 1, and the right inputs of all gates must be in $\{0,1\}$ according to Relation (2). Without loss of generality, we assume that the right inputs of (G_1, G_2, G_4, G_5) are $(0,1,1,0)$ respectively. Here we do not care about the right input of G_3 since it does not affect the final output as we will see. Then we extract the witness from the bottom to the top. For G_5, we leave its left input as blank. Then for G_4, its left input must be 1 according to Relation (2). Next, according to the same rule, we leave the left input of G_1 as blank and set the left input of G_2 as 0. One can see that by now, we have found a path (remarked as red wires in Fig. 1) leading the whole circuit to output 1. By setting the rest of the input wires assigned $\perp$ as 0, we immediately obtain a valid witness, which is 000001. One can check that it leads the circuit to output 1. For the full details, we refer the reader to Sect. 3.

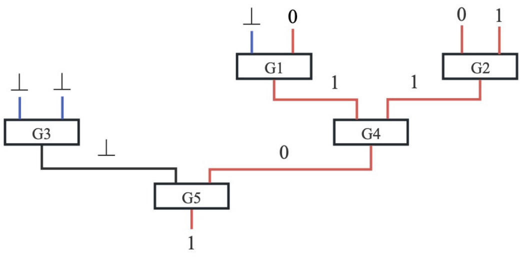

Fig. 1. An instance of the witness-extraction procedure. Without loss of generality, all the gates $\{G_i\}_{i\in[5]}$ in the statement circuit are set as NAND gates. The procedure starts from the bottom to the top. By setting the (blue) input wires assigned $\perp$ as 0, we extract a valid witness 000001 leading the circuit to output 1. (Color figure online)

Extension to batch argument for all NP. We now explain how to combine our witness-extraction strategy with the WW-BARG proposed by Waters and Wu in [48] to achieve a BARG with shorter proofs.

To prove the existence of witnesses $(\mathsf{w}_i)_{i\in[m]}$ such that $\mathsf{C}(\mathsf{x}_i, \mathsf{w}_i) = 1$ for m statements x_i, WW-BARG first extends each $(\mathsf{x}_i, \mathsf{w}_i)$ to $(\mathsf{w}_{i,j})_{j\in[t]}$ containing bits of all wires in the circuit C. Then it commits to $(\mathsf{w}_{i,j})_{i\in[m]}$ with an additively homomorphic (de-randomized) vector commitment for each wire. Next it generates succinct proofs of wire validity and gate consistency, i.e., for all $i \in [m]$, it proves that $\mathsf{w}_{i,j} \in \{0,1\}$ for each $j \in [t]$ and $1 - \mathsf{w}_{i,d_1}\mathsf{w}_{i,d_2} = \mathsf{w}_{i,d_3}$ for each gate $G_\ell = (d_1, d_2, d_3)$. The final proof size is independent of m.

Alternatively, if we can prove gate consistency with respect to Relation (2) as in the case of NIZK, then we can adopt our aforementioned witness-extraction strategy to avoid generating proofs of wire validity and achieve soundness with shorter proofs. However, we do not have an explicit "batch OR-proof" for doing this. To overcome this, we observe that WW-BARG essentially provides us with a way to prove $\mathsf{w}_{i,1}\mathsf{w}_{i,2} = 0$ for all $i \in [m]$ given two commitments to $(\mathsf{w}_{i,1})_{i\in[m]}$ and $(\mathsf{w}_{i,2})_{i\in[m]}$ respectively. Then for each gate $G_\ell = (d_1, d_2, d_3)$, we let the prover homomorphically evaluate commitments to $(1-\mathsf{w}_{i,d_1}-\mathsf{w}_{i,d_3})_{i\in[m]}$, $(1-\mathsf{w}_{i,d_3})_{i\in[m]}$, and $(1-\mathsf{w}_{i,d_2})_{i\in[m]}$ respectively, and extend WW-BARG to adopt the following relation for consistency checks:

$$(1 - \mathsf{w}_{i,d_1} - \mathsf{w}_{i,d_3})\mathsf{w}_{i,d_2} = 0 \ \wedge \ (1 - \mathsf{w}_{i,d_3})(1 - \mathsf{w}_{i,d_2}) = 0. \tag{3}$$

One can check that Relation (3) implies

$$(1 - \mathsf{w}_{i,d_3} = 0 \wedge \mathsf{w}_{i,d_2} = 0) \vee (1 - \mathsf{w}_{i,d_1} - \mathsf{w}_{i,d_3} = 0 \wedge 1 - \mathsf{w}_{i,d_2} = 0),$$

which is equivalent to Relation (2), or

$$\mathsf{w}_{i,d_1} = 0 \wedge 1 - \mathsf{w}_{i,d_3} = 0.$$

Then for any valid proof, we can extract the extended witness from the bottom to the top of the circuit in a similar way to the witness-extraction strategy for our NIZK. Here, a main difference is that there is a new case $\mathsf{w}_{i,d_1} = 0 \wedge 1 - \mathsf{w}_{i,d_3} = 0$ captured by Relation (3) but not captured by Relation (2). When this happens, we just leave w_{i,d_2} blank and continue to extract values for the gate outputting w_{i,d_1}. We refer the readers to Sect. 4 for the detailed construction and security analysis, which reflects a bulk of our main technical contribution.

2 Preliminaries

Notations. We use $x \overset{\$}{\leftarrow} \mathcal{S}$ to denote the process of sampling an element x from set $\mathcal{S}$ uniformly at random. All our algorithms are probabilistic polynomial time unless we stated otherwise. If $\mathcal{A}$ is a probabilistic algorithm, then we write $a \overset{\$}{\leftarrow} \mathcal{A}(b)$ to denote the random variable that outputted by $\mathcal{A}$ on input b. By $\mathsf{negl}(\cdot)$ we mean an unspecified negligible function.

2.1 Pairing Groups and Matrix Diffie-Hellman Assumptions

Let GGen be a probabilistic polynomial time (PPT) algorithm that on input 1^λ returns a description $\mathcal{G} := (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, P_1, P_2, e)$ of asymmetric pairing groups where $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ are cyclic groups of order p for a λ-bit prime p, P_1 and P_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively, and $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is an efficiently computable (non-degenerate) bilinear map. Define $P_T := e(P_1, P_2)$, which is a generator in $\mathbb{G}_T$. Unless stated otherwise, we consider Type III pairings, where $\mathbb{G}_1 \neq \mathbb{G}_2$ and there is no efficient homomorphism between them.

We use implicit representation of group elements as in [18]. For $s \in \{1, 2, T\}$ and $a \in \mathbb{Z}_p$ define $[a]_s = aP_s \in \mathbb{G}_s$ as the implicit representation of a in $\mathbb{G}_s$. Similarly, for a matrix $\mathbf{A} = (a_{ij}) \in \mathbb{Z}_p^{n\times m}$ we define $[\mathbf{A}]_s$ as the implicit representation of $\mathbf{A}$ in $\mathbb{G}_s$. $\mathsf{Span}(\mathbf{A}) := \{\mathbf{Ar} | \mathbf{r} \in \mathbb{Z}_p^m\} \subset \mathbb{Z}_p^n$ denotes the linear span of $\mathbf{A}$, and similarly $\mathsf{Span}([\mathbf{A}]_s) := \{[\mathbf{Ar}]_s | \mathbf{r} \in \mathbb{Z}_p^m\} \subset \mathbb{G}_s^n$. Note that it is efficient to compute $[\mathbf{AB}]_s$ given $([\mathbf{A}]_s, \mathbf{B})$ or $(\mathbf{A}, [\mathbf{B}]_s)$ with matching dimensions. We define $[\mathbf{A}]_1 \circ [\mathbf{B}]_2 := e([\mathbf{A}]_1, [\mathbf{B}]_2) = [\mathbf{AB}]_T$, which can be efficiently computed given $[\mathbf{A}]_1$ and $[\mathbf{B}]_2$.

Next we recall the definition of the Matrix Decisional Diffie-Hellman (MDDH) [18] and related assumptions [43].

Definition 1 (Matrix distribution). *Let $k, \ell \in \mathbb{N}$ with $\ell > k$. We call $\mathcal{D}_{\ell,k}$ a matrix distribution if it outputs matrices in $\mathbb{Z}_p^{\ell\times k}$ of full rank k in polynomial time. By $\mathcal{D}_k$ we denote $\mathcal{D}_{k+1,k}$.*

For a matrix $\mathbf{A} \overset{\$}{\leftarrow} \mathcal{D}_{\ell,k}$, we define the set of kernel matrices of $\mathbf{A}$ as

$$\mathsf{ker}(\mathbf{A}) := \{\mathbf{A}^\perp \in \mathbb{Z}_p^{\ell\times(\ell-k)} \mid (\mathbf{A}^\perp)^\top \cdot \mathbf{A} = \mathbf{0} \in \mathbb{Z}_p^{(\ell-k)\times k} \text{ and } \mathbf{A} \text{ has rank } (\ell - k)\}.$$

Given a matrix $\mathbf{A}$ over $\mathbb{Z}_p^{\ell\times k}$, it is efficient to sample an $\mathbf{A}^\perp$ from $\mathsf{ker}(\mathbf{A})$.

The $\mathcal{D}_{\ell,k}$-Matrix Diffie-Hellman problem is to distinguish the two distributions $([\mathbf{A}],[\mathbf{Aw}])$ and $([\mathbf{A}],[\mathbf{u}])$ where $\mathbf{A} \xleftarrow{\$} \mathcal{D}_{\ell,k}$, $\mathbf{w} \xleftarrow{\$} \mathbb{Z}_p^k$ and $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^\ell$.

Definition 2 ($\mathcal{D}_{\ell,k}$-matrix decisional Diffie-Hellman assumption *[18]*). *Let $\mathcal{D}_{\ell,k}$ be a matrix distribution and $s \in \{1,2,T\}$. We say that the $\mathcal{D}_{\ell,k}$-Matrix Diffie-Hellman ($\mathcal{D}_{\ell,k}$-MDDH) is hard relative to GGen in group $\mathbb{G}_s$ if for all PPT adversaries $\mathcal{A}$, it holds that*

$$|\Pr[1 \xleftarrow{\$} \mathcal{A}(\mathcal{G},[\mathbf{A}]_s,[\mathbf{Aw}]_s)] - \Pr[1 \xleftarrow{\$} \mathcal{A}(\mathcal{G},[\mathbf{A}]_s,[\mathbf{u}]_s)]| \leq \mathsf{negl}(\lambda),$$

where $\mathcal{G} \xleftarrow{\$} \mathsf{GGen}(\mathsf{par})$, $\mathbf{A} \xleftarrow{\$} \mathcal{D}_{\ell,k}, \mathbf{w} \xleftarrow{\$} \mathbb{Z}_p^k$ and $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^\ell$.

2.2 Non-Interactive Zero-Knowledge Proof

Let $\lambda \in \mathbb{N}$ be the security parameter determining a public parameter par. We define NIZK as follows.

Definition 3 (Non-interactive zero-knowledge proof [30]). *A* non-interactive zero-knowledge proof (NIZK) *for a family of languages $\{\mathcal{L}_{\mathsf{par}}\}$ consists of three PPT algorithms $\mathsf{NIZK} = (\mathsf{NGen}, \mathsf{NProve}, \mathsf{NVer})$ such that:*

- $\mathsf{NGen}(1^\lambda, \mathsf{par})$ *returns a common reference string* crs.
- $\mathsf{NProve}(\mathsf{crs}, \mathsf{C}, \mathsf{x}, \mathsf{w})$ *returns a proof* π.
- $\mathsf{NVer}(\mathsf{crs}, \mathsf{C}, \mathsf{x}, \pi)$ *returns* 1 *(accept) or* 0 *(reject). Here,* NVer *is deterministic.*

Completeness *is satisfied if for all* $(\mathsf{C},\mathsf{x}) \in \mathcal{L}_{\mathsf{par}}$ *and all* w *such that* $\mathsf{C}(\mathsf{x},\mathsf{w}) = 1$, *all* $\mathsf{crs} \in \mathsf{NGen}(1^\lambda, \mathsf{par})$, *and all* $\pi \in \mathsf{NProve}(\mathsf{crs},\mathsf{x},\mathsf{w})$, *we have* $\mathsf{NVer}(\mathsf{crs},\mathsf{x},\pi) = 1$.

Definition 4 (Composable zero-knowledge). *A NIZK* $\mathsf{NIZK} = (\mathsf{NGen}, \mathsf{NProve}, \mathsf{NVer})$ *is said to satsify* composable zero-knowledge *if there exist a simulator consisting of two PPT algorithms* $(\mathsf{NTGen}, \mathsf{NSim})$ *such that*

- $\mathsf{NTGen}(1^\lambda, \mathsf{par})$ *returns* crs *and a trapdoor* td,
- $\mathsf{NSim}(\mathsf{crs}, \mathsf{td}, \mathsf{C}, \mathsf{x})$ *returns a proof* π,

and for any PPT adversary $\mathcal{A}$, we have

$$\begin{aligned}|\Pr[1 \xleftarrow{\$} \mathcal{A}(\mathsf{crs})|\mathsf{crs} \xleftarrow{\$} \mathsf{NGen}(1^\lambda,\mathsf{par})]\\ - \Pr[1 \xleftarrow{\$} \mathcal{A}(\mathsf{crs})|(\mathsf{crs},\mathsf{td}) \xleftarrow{\$} \mathsf{NTGen}(1^\lambda,\mathsf{par})]| \leq \mathsf{negl}(\lambda),\end{aligned}$$

and for all (x,w) *such that* $\mathsf{C}(\mathsf{x},\mathsf{w}) = 1$, *the following distributions are identical.*

$$\pi \xleftarrow{\$} \mathsf{NProve}(\mathsf{crs},\mathsf{C},\mathsf{x},\mathsf{w}) \text{ and } \pi \xleftarrow{\$} \mathsf{NSim}(\mathsf{crs},\mathsf{td},\mathsf{C},\mathsf{x}),$$

where $(\mathsf{crs},\mathsf{td}) \xleftarrow{\$} \mathsf{NTGen}(1^\lambda,\mathsf{par})$.

Definition 5 (Perfect soundness). *A NIZK* $\mathsf{NIZK} = (\mathsf{NGen}, \mathsf{NProve}, \mathsf{NVer})$ *is said to satisfy* perfect soundness *if for all* $\mathsf{x} \notin \mathcal{L}_{\mathsf{par}}$, *all* $\mathsf{crs} \in \mathsf{NGen}(1^\lambda,\mathsf{par})$, *and all* π, *we have* $\mathsf{NVer}(\mathsf{crs},\mathsf{C},\mathsf{x},\pi) = 0$.

Witness-extractor. One can easily see that for any statement, if there exists a (possibly inefficient) witness-extractor that can extract a valid witness from any valid proof passing the verification, then perfect soundness is satisfied.

Dual mode. A NIZK defined as above satisfies computational zero-knowledge and perfect soundness. By generating CRSs with NTGen instead of NGen, we immediately achieve its dual mode with perfect zero-knowledge but computational soundness.

2.3 Batch Argument

Let $\lambda \in \mathbb{N}$ be the security parameter determining a public parameter par. We define batch argument as follows.

Definition 6 (Batch argument). *A* batch argument (BARG) *for a family of languages* $\{\mathcal{L}_{\mathsf{par}}\}$ *consists of three PPT algorithms* $\mathsf{BARG} = (\mathsf{BGen}, \mathsf{BProve}, \mathsf{BVer})$ *such that*

- $\mathsf{BGen}(1^\lambda, \mathsf{par}, 1^m)$ *returns a common reference string* crs.
- $\mathsf{BProve}(\mathsf{crs}, \mathsf{C}, (\mathsf{x}_i)_{i\in[m]}, (\mathsf{w}_i)_{i\in[m]})$ *returns a proof* π.
- $\mathsf{BVer}(\mathsf{crs}, \mathsf{C}, (\mathsf{x}_i)_{i\in[m]}, \pi)$ *returns* 1 *(accept) or* 0 *(reject). Here,* BVer *is deterministic.*

Completeness *is satisfied if for all* $\lambda, m \in \mathbb{N}$*, all* $(\mathsf{C}, (\mathsf{x}_i)_{i\in[m]}) \in \mathcal{L}_{\mathsf{par}}$*, all* $(\mathsf{w}_i)_{i\in[m]})$ *such that* $\mathsf{C}(\mathsf{x}_i, \mathsf{w}_i) = 1$ *for all* $i \in [m]$*, all* $\mathsf{crs} \in \mathsf{BGen}(1^\lambda, \mathsf{par}, 1^m)$*, and all* $\pi \in \mathsf{BProve}(\mathsf{crs}, \mathsf{C}, (\mathsf{x}_i)_{i\in[m]}, (\mathsf{w}_i)_{i\in[m]})$*, we have* $\mathsf{BVer}(\mathsf{crs}, \mathsf{C}, (\mathsf{x}_i)_{i\in[m]}, \pi) = 1$.

Definition 7 (Somewhere argument of knowledge). *A BARG* $\mathsf{BARG} = (\mathsf{BGen}, \mathsf{BProve}, \mathsf{BVer})$ *for* $\{\mathcal{L}_{\mathsf{par}}\}$ *is said to be a* somewhere argument of knowledge *if there exist two PPT algorithms* $(\mathsf{BTGen}, \mathsf{BExt})$ *such that*

- $\mathsf{BTGen}(1^\lambda, \mathsf{par}, 1^m, i^*)$ *returns a common reference string* crs *and a trapdoor* td*,*
- $\mathsf{BExt}(\mathsf{td}, \mathsf{C}, (\mathsf{x}_i)_{i\in[m]}, \pi)$ *returns a witness* w^**. Here,* BExt *is deterministic,*

and $(\mathsf{BTGen}, \mathsf{BExt})$ *satisfy the following two properties.*

CRS indistinguishability*: for all* $\lambda, m \in \mathbb{N}$*, all* $i^* \in [m]$*, and all PPT adversary* $\mathcal{A}$*, we have*

$$|\Pr[1 \xleftarrow{\$} \mathcal{A}(\mathsf{crs}) | \mathsf{crs} \xleftarrow{\$} \mathsf{BGen}(1^\lambda, \mathsf{par}, 1^m)] - \Pr[1 \xleftarrow{\$} \mathcal{A}(\mathsf{crs}) | (\mathsf{crs}, \mathsf{td}) \xleftarrow{\$} \mathsf{BTGen}(1^\lambda, \mathsf{par}, 1^m, i^*)]| \leq \mathsf{negl}(\lambda).$$

Somewhere extractability in trapdoor mode*: for all polynomial* $m = m(\lambda)$*, all* $i^* \in [m]$*, and all adversary* $\mathcal{A}$*, we have*

$$\Pr[\mathsf{BVer}(\mathsf{crs}^*, \mathsf{C}, (\mathsf{x}_i)_{i\in[m]}, \pi) = 1 \wedge C(\mathsf{x}_{i^*}, \mathsf{w}_{i^*}) \neq 1 | (\mathsf{crs}^*, \mathsf{td}) \xleftarrow{\$} \mathsf{BTGen}(1^\lambda, \mathsf{par}, 1^m, i^*), (\mathsf{C}, (\mathsf{x}_i)_{i\in[m]}, \pi) \xleftarrow{\$} \mathcal{A}(\mathsf{crs}^*), \mathsf{w}_{i^*} \xleftarrow{\$} \mathsf{BExt}(\mathsf{td}, \mathsf{C}, (\mathsf{x}_i)_{i\in[m]}, \pi)] \leq \mathsf{negl}(\lambda).$$

As noted in [48], somewhere extractability implies non-adaptive soundness, i.e., soundness for statements independent of the CRS (see [48] for the formal definition), by a standard hybrid argument.[3]

[3] A security loss of $O(m)$ occurs in the hybrid argument.

Definition 8 (Succinctness). *A batch argument* $\mathsf{BARG} = (\mathsf{BGen}, \mathsf{BProve}, \mathsf{BVer})$ *for* $\{\mathcal{L}_{\mathsf{par}}\}$ *is said to satisfy* succinctness *if there exists a fixed polynomial* $\mathsf{poly}(\cdot,\cdot,\cdot)$ *such that for all* $\lambda, m \in \mathbb{N}$*, all* $\mathsf{crs} \in \mathsf{BGen}(1^\lambda, \mathsf{par}, 1^m)$*, and all* $(\mathsf{C} : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}, (\mathsf{x}_i)_{i\in[m]}) \in \mathcal{L}_{\mathsf{par}}$*, the following properties hold:*

Succinct proofs: all $\pi \in \mathsf{BProve}(\mathsf{crs}, \mathsf{C}, (\mathsf{x}_i)_{i\in[m]}, (\mathsf{w}_i)_{i\in[m]})$ *where* $\mathsf{C}(\mathsf{x}_i, \mathsf{w}_i) = 1$ *for all* $i \in [m]$ *satisfies* $|\pi| \le \mathsf{poly}(\lambda, \log m, s)$*.*

Succinct CRS: all $\mathsf{crs} \in \mathsf{Gen}(1^\lambda, \mathsf{par}, 1^m)$ *satisfies* $|\mathsf{crs}| \le \mathsf{poly}(\lambda, m, n) + \mathsf{poly}(\lambda, \log m, s)$*.*

Succinct verification: BVer *runs in time* $\mathsf{poly}(\lambda, m, n) + \mathsf{poly}(\lambda, \log m, s)$*.*

Above by s *we denote the number of gates in* C*.*

3 Simple NIZK from OR-Proof

In this section, we recall an efficient instantiation of OR-proof and give a new framework for converting an OR-proof into an efficient NIZK for circuit satisfiability in NP.

3.1 NIZK for OR-Language

We now recall the OR-proof system based on the MDDH assumptions presented in [39,45] and implicitly given in [29]. As far as we know, this is the most efficient OR-proof by now in the standard model.

For the language

$$\mathsf{L}^{\mathsf{or}}_{[\mathbf{A}]_1} = \{(\mathsf{C}_{[\mathbf{A}]_1}, ([\mathbf{x}_0]_1, [\mathbf{x}_1]_1)) | \exists \mathbf{w} \in \mathbb{Z}_p^t : \mathsf{C}_{[\mathbf{A}]_1}([\mathbf{x}_0, \mathbf{x}_1]_1, \mathbf{w}) = 1\},$$

where $[\mathbf{A}]_1 \in \mathbb{G}_1^{n\times t}$ is public and $\mathsf{C}_{[\mathbf{A}]_1}$ is a Boolean circuit on input $([\mathbf{x}_0, \mathbf{x}_1]_1, \mathbf{w})$ outputting 1 iff $[\mathbf{x}_0]_1 = [\mathbf{A}]_1\mathbf{w} \vee [\mathbf{x}_1]_1 = [\mathbf{A}]_1\mathbf{w}$, the OR-proof system ORNIZK with each public parameter containing $\mathsf{par} = (\mathcal{G} \overset{\$}{\leftarrow} \mathsf{GGen}(1^\lambda))$ is defined as in Fig. 2.

Lemma 1. *If the* $\mathcal{D}_k$-MDDH *assumption holds in the group* $\mathbb{G}_2$*, then the proof system* $\mathsf{ORNIZK} = (\mathsf{NGen}_{\mathsf{or}}, \mathsf{NTGen}_{\mathsf{or}}, \mathsf{NProve}_{\mathsf{or}}, \mathsf{NVer}_{\mathsf{or}}, \mathsf{NSim}_{\mathsf{or}})$ *is a NIZK with perfect completeness, perfect soundness, and composable zero-knowledge. For any adversary* $\mathcal{A}$ *against the composable zero-knowledge of* ORNIZK*, there exists a tight reduction algorithm breaking the* MDDH *assumption by using* $\mathcal{A}$ *in a black-box way with security loss* $O(1)$*.*

We refer the reader to [39,45] for the detailed proof.

3.2 Our NIZK for NP

Before giving our NIZK for NP, we first introduce the notion of circuit satisfiability.

```
$\underline{\mathsf{NGen_{or}}(1^\lambda, \mathsf{par})}$:
$\mathbf{D} \xleftarrow{\$} \mathcal{D}_k$, $\mathbf{z} \xleftarrow{\$} \mathbb{Z}_p^{k+1} \backslash \mathsf{Span}(\mathbf{D})$
Return $\mathsf{crs} = (\mathsf{par}, [\mathbf{D}]_2, [\mathbf{z}]_2)$

$\underline{\mathsf{NProve_{or}}(\mathsf{crs}, \mathsf{C}_{[\mathbf{A}]_1}, ([\mathbf{x}_0]_1, [\mathbf{x}_1]_1), \mathbf{r})}$:
let $j \in \{0,1\}$ s.t. $[\mathbf{x}_j]_1 = [\mathbf{A}]_1 \cdot \mathbf{r}$
$\mathbf{v} \xleftarrow{\$} \mathbb{Z}_p^k$, $[\mathbf{z}_{1-j}]_2 = [\mathbf{D}]_2 \cdot \mathbf{v}$, $[\mathbf{z}_j]_2 = [\mathbf{z}]_2 - [\mathbf{z}_{1-j}]_2$, $\mathbf{S}_0, \mathbf{S}_1 \xleftarrow{\$} \mathbb{Z}_p^{t \times k}$
$[\mathbf{C}_j]_2 = \mathbf{S}_j \cdot [\mathbf{D}]_2^\top + \mathbf{r} \cdot [\mathbf{z}_j]_2^\top \in \mathbb{G}_2^{t \times (k+1)}$, $[\pi_j]_1 = [\mathbf{A}]_1 \cdot \mathbf{S}_j \in \mathbb{G}_1^{n \times k}$
$[\mathbf{C}_{1-j}]_2 = \mathbf{S}_{1-j} \cdot [\mathbf{D}]_2^\top$, $[\pi_{1-j}]_1 = [\mathbf{A}]_1 \cdot \mathbf{S}_{1-j} - [\mathbf{x}_{1-j}]_1 \cdot \mathbf{v}^\top$
Return $\pi = ([\mathbf{z}_0]_2, ([\mathbf{C}_i]_2, [\pi_i]_1)_{i \in \{0,1\}})$

$\underline{\mathsf{NVer_{or}}(\mathsf{crs}, \mathsf{C}_{[\mathbf{A}]_1}, [\mathbf{x}]_1, \pi)}$:
$[\mathbf{z}_1]_2 = [\mathbf{z}]_2 - [\mathbf{z}_0]_2$
If for all $i \in \{0,1\}$ it holds $[\mathbf{A}]_1 \circ [\mathbf{C}_i]_2 = [\pi_i]_1 \circ [\mathbf{D}^\top]_2 + [\mathbf{x}_i]_1 \circ [\mathbf{z}_i^\top]_2$, return 1
Else return 0

$\underline{\mathsf{NTGen_{or}}(1^\lambda, \mathsf{par})}$:
$\mathbf{D} \xleftarrow{\$} \mathcal{D}_k$, $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^k$, $\mathbf{z} = \mathbf{D} \cdot \mathbf{u}$
Return $(\mathsf{crs} = (\mathsf{par}, [\mathbf{D}]_2, [\mathbf{z}]_2), \mathsf{td} = \mathbf{u}$

$\underline{\mathsf{NSim_{or}}(\mathsf{crs}, \mathsf{td}, \mathsf{C}_{[\mathbf{A}]_1}, ([\mathbf{x}_0]_1, [\mathbf{x}_1]_1))}$:
$\mathbf{v} \xleftarrow{\$} \mathbb{Z}_p^k$, $[\mathbf{z}_0]_2 = [\mathbf{D}]_2 \cdot \mathbf{v}$, $[\mathbf{z}_1]_2 = [\mathbf{z}]_2 - [\mathbf{z}_0]_2$
$\mathbf{S}_0, \mathbf{S}_1 \xleftarrow{\$} \mathbb{Z}_p^{t \times k}$, $[\mathbf{C}_0]_2 = \mathbf{S}_0 \cdot [\mathbf{D}]_2^\top$, $[\pi_0]_1 = [\mathbf{A}_0]_1 \cdot \mathbf{S}_0 - [\mathbf{x}_0]_1 \cdot \mathbf{v}^\top$
$[\mathbf{C}_1]_2 = \mathbf{S}_1 \cdot [\mathbf{D}]_2^\top$, $[\pi_1]_1 = [\mathbf{A}]_1 \cdot \mathbf{S}_1 - [\mathbf{x}_1]_1 \cdot (\mathbf{u} - \mathbf{v})^\top$
Return $\pi = ([\mathbf{z}_0]_2, ([\mathbf{C}_i]_2, [\pi_i]_1)_{i \in \{0,1\}})$
```

Fig. 2. Construction of $\mathsf{ORNIZK} = (\mathsf{NGen_{or}}, \mathsf{NProve_{or}}, \mathsf{NVer_{or}})$ with the simulator $(\mathsf{NTGen_{or}}, \mathsf{NSim_{or}})$.

Definition 9 (Circuit satisfiability). *Let λ be the security parameter. The* circuit satisfiability *language is defined as*

$$\mathcal{L}_\lambda^{\mathsf{CSAT}} = \{(\mathsf{C}, \mathsf{x}) | \exists \mathsf{w} \in \{0,1\}^h : \mathsf{C}(\mathsf{x}, \mathsf{w}) = 1\},$$

where $\mathsf{C} : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ is any Boolean circuit with polynomial size in λ and $\mathsf{x} \in \{0,1\}^n$ is the instance. Without loss of generality, we assume that C consists only of fan-in-2 NAND gates.

Let λ be the security parameter and $\mathsf{par} = \mathcal{G}$ be the public parameter, where $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, [1]_1, [1]_2, e) \xleftarrow{\$} \mathsf{GGen}(1^\lambda)$. Let $\mathsf{ORNIZK} = (\mathsf{NGen_{or}}, \mathsf{NProve_{or}}, \mathsf{NVer_{or}})$ be an OR-proof with the simulator $(\mathsf{NTGen_{or}}, \mathsf{NSim_{or}})$, where each public parameter is comprised of $\mathcal{G}$. Let $\mathsf{L}_{[\mathbf{M}']_1}^{\mathsf{or}}$ be the following language it supports.

$$\mathsf{L}_{[\mathbf{M}']_1}^{\mathsf{or}} = \{(\mathsf{C}_{[\mathbf{M}']_1}, ([\mathbf{x}_0]_1, [\mathbf{x}_1]_1)) | \exists \mathbf{w} \in \mathbb{Z}_p^{2k} : \mathsf{C}_{[\mathbf{M}']_1}(([\mathbf{x}_0]_1, [\mathbf{x}_1]_1), \mathbf{w}) = 1\},$$

where $\mathsf{C}_{[\mathbf{M}']_1} : \mathbb{G}_1^{2k+2} \times \mathbb{G}_1^{2k+2} \times \mathbb{Z}_p^{2k} \to \{0,1\}$ is a Boolean circuit on input $((\mathbf{x}_0, \mathbf{x}_1), \mathbf{w})$ outputting 1 iff $[\mathbf{x}_0]_1 = [\mathbf{M}']_1 \mathbf{w} \vee [\mathbf{x}_1]_1 = [\mathbf{M}']_1 \mathbf{w}$ for $\mathbf{M}' = \begin{pmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{pmatrix}$

and $\mathbf{M} \in \mathcal{D}_k$. We give our NIZK for $\mathcal{L}_\lambda^{\mathsf{CSAT}}$ in Fig. 3. Roughly, we first extend the witness to all wires, commit to all the values, and use the OR-proof to prove that committed values satisfy Relation (2) (see Sect. 1.3) for each gate.[4]

```
$\underline{\mathsf{NGen}(1^\lambda, \mathsf{par})\text{:}}$
$\mathbf{M} \xleftarrow{\$} \mathcal{D}_k$, $\mathbf{z} \xleftarrow{\$} \mathbb{Z}_p^{k+1} \backslash \mathsf{Span}(\mathbf{M})$, $\mathsf{crs}_{\mathsf{or}} \xleftarrow{\$} \mathsf{NGen}_{\mathsf{or}}(1^\lambda, \mathsf{par})$
Return $\mathsf{CRS} = (\mathsf{crs}_{\mathsf{or}}, [\mathbf{M}]_1, [\mathbf{z}]_1)$

$\underline{\mathsf{NProve}(\mathsf{CRS}, \mathsf{C}, \mathsf{x}, \mathsf{w})\text{:}}$
Hardwire $\mathsf{x}$ in $\mathsf{C}$ to obtain the circuit $\mathsf{C}(\mathsf{x}, \cdot) : \{0,1\}^h \to \{0,1\}$
Define $s$ and $t$ to be the numbers of gates and wires of $\mathsf{C}(\mathsf{x}, \cdot)$ respectively
Extend $\mathsf{w}$ to $(\mathsf{w}_i)_{i \in [t]}$ containing the bits of all wires in $\mathsf{C}(\mathsf{x}, \cdot)$
Compute $\mathbf{r}_i \xleftarrow{\$} \mathbb{Z}_p^k$ and $\mathsf{cm}_i = [\mathbf{M}]_1 \mathbf{r}_i + [\mathbf{z}]_1 \mathsf{w}_i$ for all $i \in [t-1]$
Set $\mathbf{r}_t = \mathbf{0}$ and $\mathsf{cm}_t = [\mathbf{z}]_1$ for the output wire
For each $\mathsf{NAND}$ gate $G_\ell = (d_1, d_2, d_3) \in [t]^3$ where $\ell \in [s]$, run
  – $\mathsf{x}_{\ell,1} = \begin{pmatrix} -[\mathbf{z}]_1 + \mathsf{cm}_{d_1} + \mathsf{cm}_{d_3} \\ -[\mathbf{z}]_1 + \mathsf{cm}_{d_2} \end{pmatrix}$, $\mathbf{r}'_{\ell,1} = \begin{pmatrix} \mathbf{r}_{d_1} + \mathbf{r}_{d_3} \\ \mathbf{r}_{d_2} \end{pmatrix}$
  – $\mathsf{x}_{\ell,2} = \begin{pmatrix} -[\mathbf{z}]_1 + \mathsf{cm}_{d_3} \\ \mathsf{cm}_{d_2} \end{pmatrix}$, $\mathbf{r}'_{\ell,2} = \begin{pmatrix} \mathbf{r}_{d_3} \\ \mathbf{r}_{d_2} \end{pmatrix}$
  – $\pi_\ell \xleftarrow{\$} \mathsf{NProve}_{\mathsf{or}}(\mathsf{crs}_{\mathsf{or}}, \mathsf{C}_{[\mathbf{M}']_1}, (\mathsf{x}_{\ell,1}, \mathsf{x}_{\ell,2}), \mathbf{r}'_b)$ if $\mathsf{x}_{\ell,b} = [\mathbf{M}']_1 \mathbf{r}'_{\ell,b}$ for $b \in \{1,2\}$
    and abort otherwise
Return $\Pi = ((\mathsf{cm}_i)_{i \in [t]}, (\pi_\ell)_{\ell \in [s]})$

$\underline{\mathsf{NVer}(\mathsf{CRS}, \mathsf{C}, \mathsf{x}, \Pi)\text{:}}$
Hardwire $\mathsf{x}$ in $\mathsf{C}$ to obtain $\mathsf{C}(\mathsf{x}, \cdot)$ in the same way as $\mathsf{NProve}$ does
Check that all wires in $\mathsf{C}(\mathsf{x}, \cdot)$ have a corresponding commitment and $\mathsf{cm}_t = [\mathbf{z}]_1$
Check that all $\mathsf{NAND}$ gates have a valid OR-proof of compliance
Return 1 iff all checks pass
```

Fig. 3. Definition of $\mathsf{NIZK} = (\mathsf{NGen}, \mathsf{NProve}, \mathsf{NVer})$. By $G_\ell = (d_1, d_2, d_3)$ we mean that the left and right input wires of the gate G_ℓ are indexed as d_1 and d_2 respectively, while the output wire of G_ℓ is indexed as d_3. Notice that for each multiple fan-out gate, we only increment the count of wires t by 1 for its output wires and generate only one commitment for these wires, since all output wires of the gate are assigned the same value and serves as input wires for multiple other gates. The same argument is also made for all other proof systems given later.

Theorem 1 (Completeness). *If* ORNIZK *is complete, then* NIZK *is complete.*

Proof. Let w_{d_1} and w_{d_2} be the input bits of a NAND gate, and w_{d_3} be the true output. We must have

$$(-1 + \mathsf{w}_{d_1} + \mathsf{w}_{d_3} = 0 \wedge -1 + \mathsf{w}_{d_2} = 0) \text{ or } (-1 + \mathsf{w}_{d_3} = 0 \wedge \mathsf{w}_{d_2} = 0).$$

[4] Notice that we do not define the commitment and its properties in advance since we use a concrete construction based on the MDDH assumption in a non-black-box way.

Let $\mathsf{cm}_{d_1} = [\mathbf{M}\mathbf{r}_{d_1} + \mathbf{z}\mathsf{w}_{d_1}]_1$ and $\mathsf{cm}_{d_2} = [\mathbf{M}\mathbf{r}_{d_2} + \mathbf{z}\mathsf{w}_{d_2}]_1$ be the input commitments and $\mathsf{cm}_{d_3} = [\mathbf{M}\mathbf{r}_{d_3} + \mathbf{z}\mathsf{w}_{d_3}]_1$ be the output commitment. We have

$$\begin{aligned}\mathsf{x}_{\ell,1} &= \begin{pmatrix} -[\mathbf{z}]_1 + \mathsf{cm}_{d_1} + \mathsf{cm}_{d_3} \\ -[\mathbf{z}]_1 + \mathsf{cm}_{d_2} \end{pmatrix} = [\mathbf{M}']_1 \begin{pmatrix} \mathbf{r}_{d_1} + \mathbf{r}_{d_3} \\ \mathbf{r}_{d_2} \end{pmatrix} + \begin{pmatrix} [\mathbf{z}]_1(-1 + \mathsf{w}_{d_1} + \mathsf{w}_{d_3}) \\ [\mathbf{z}]_1(-1 + \mathsf{w}_{d_2}) \end{pmatrix} \\ &= [\mathbf{M}']_1 \begin{pmatrix} \mathbf{r}_{d_1} + \mathbf{r}_{d_3} \\ \mathbf{r}_{d_2} \end{pmatrix}\end{aligned}$$

$$\text{or} \quad \mathbf{x}_{\ell,2} = \begin{pmatrix} -[\mathbf{z}]_1 + \mathsf{cm}_{d_3} \\ \mathsf{cm}_{d_2} \end{pmatrix} = [\mathbf{M}']_1 \begin{pmatrix} \mathbf{r}_{d_3} \\ \mathbf{r}_{d_2} \end{pmatrix} + \begin{pmatrix} [\mathbf{z}]_1(-1 + \mathsf{w}_{d_3}) \\ [\mathbf{z}]_1 \mathsf{w}_{d_2} \end{pmatrix} = [\mathbf{M}']_1 \begin{pmatrix} \mathbf{r}_{d_3} \\ \mathbf{r}_{d_2} \end{pmatrix}.$$

Therefore, we have $\mathsf{x}_{\ell,1} \in \mathsf{Span}([\mathbf{M}']_1)$ if $\mathsf{w}_{d_2} = 1$ and $\mathsf{x}_{\ell,2} \in \mathsf{Span}([\mathbf{M}']_1)$ otherwise. Then the completeness of NIZK follows from the completeness of ORNIZK, completing the proof of Theorem 1. ☐

Theorem 2 (Composable zero-knowledge). *Under the $\mathcal{D}_k$-MDDH assumption, if ORNIZK satisfies composable zero-knowledge, then NIZK satisfies composable zero-knowledge.*

```
NTGen(1^λ, par):
$\mathbf{M} \xleftarrow{\$} \mathcal{D}_k$, $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^k$, $\mathbf{z} = \mathbf{M} \cdot \mathbf{u}$, $(\mathsf{crs}_{\mathsf{or}}, \mathsf{td}_{\mathsf{or}}) \xleftarrow{\$} \mathsf{NTGen}_{\mathsf{or}}(1^\lambda, \mathsf{par})$
Return $\mathsf{CRS} = (\mathsf{crs}_{\mathsf{or}}, [\mathbf{M}]_1, [\mathbf{z}]_1)$ and $\mathsf{TD} = \mathsf{td}_{\mathsf{or}}$

NSim(CRS, TD, C, x):
Hardwire x in C to obtain $\mathsf{C}(\mathsf{x}, \cdot)$ in the same way as NProve does
Define $s$ and $t$ to be the numbers of gates and wires of $\mathsf{C}(\mathsf{x}, \cdot)$ respectively
Compute $\mathbf{r}_i \xleftarrow{\$} \mathbb{Z}_p^k$ and $\mathsf{cm}_i = [\mathbf{M}\mathbf{r}_i]_1$ for all $i \in [t-1]$
Set $\mathsf{cm}_t = [\mathbf{z}]_1$ for the output wire
For each NAND gate $G_\ell = (d_1, d_2, d_3) \in [t]^3$ where $\ell \in [s]$, run
  – $\mathsf{x}_{\ell,1} = \begin{pmatrix} -[\mathbf{z}]_1 + \mathsf{cm}_{d_1} + \mathsf{cm}_{d_3} \\ -[\mathbf{z}]_1 + \mathsf{cm}_{d_2} \end{pmatrix}$, $\mathsf{x}_{\ell,2} = \begin{pmatrix} -[\mathbf{z}]_1 + \mathsf{cm}_{d_3} \\ \mathsf{cm}_{d_2} \end{pmatrix}$
  – $\pi_\ell \xleftarrow{\$} \mathsf{NSim}_{\mathsf{or}}(\mathsf{crs}_{\mathsf{or}}, \mathsf{td}_{\mathsf{or}}, \mathsf{C}_{[\mathbf{M}']_1}, (\mathsf{x}_{\ell,1}, \mathsf{x}_{\ell,2}))$
Return $\Pi = ((\mathsf{cm}_i)_{i \in [t]}, (\pi_\ell)_{\ell \in [s]})$
```

Fig. 4. Definition of the simulator (NTGen, NSim).

Proof. We define the simulator (NTGen, NSim) as in Fig. 4.

First we note that the distribution of $\mathbf{z} \xleftarrow{\$} \mathbb{Z}_p^{k+1} \setminus \mathsf{Span}(\mathbf{M})$ is $1/p$-statistically close to the uniform distribution over $\mathbb{Z}_p^{k+1}$. Then the indistinguishability of CRSs generated by $\mathsf{NGen}(1^\lambda, \mathsf{par})$ and $\mathsf{NTGen}(1^\lambda, \mathsf{par})$ follows immediately from the $\mathcal{D}_k$-MDDH assumption and the composable zero-knowledge of ORNIZK (which says that $\mathsf{crs}_{\mathsf{or}}$ generated by $\mathsf{NGen}_{\mathsf{or}}$ and $\mathsf{NTGen}_{\mathsf{or}}$ are computationally close).

Next we define a modified prover NProve', which is exactly the same as NProve except that for each NAND gate, π_ℓ is generated as

$$\pi_\ell \xleftarrow{\$} \mathsf{NSim}_{\mathsf{or}}(\mathsf{crs}_{\mathsf{or}}, \mathsf{td}_{\mathsf{or}}, \mathsf{C}_{[\mathbf{M}']_1}, (\mathsf{x}_{\ell,1}, \mathsf{x}_{\ell,2})).$$

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

$$\varPi \xleftarrow{\$} \mathsf{NProve}(\mathsf{CRS}, \mathsf{C}, \mathsf{x}, \mathsf{w}) \text{ and } \varPi \xleftarrow{\$} \mathsf{NProve}'(\mathsf{CRS}, \mathsf{C}, \mathsf{x}, \mathsf{w})$$

for $(\mathsf{CRS}, \mathsf{TD}) \xleftarrow{\$} \mathsf{NTGen}(1^\lambda, \mathsf{par})$ and any (x, w) such that $\mathsf{C}(\mathsf{x}, \mathsf{w}) = 1$.

Moreover, since the distribution of $\mathsf{cm}_i = [\mathbf{M}\mathbf{r}_i]_1$ is identical to that of $\mathsf{cm}_i = [\mathbf{M}\mathbf{r}_i + \mathbf{z}\mathsf{w}_i]_1$ for $\mathbf{r}_i \xleftarrow{\$} \mathbb{Z}_p^\lambda$ when $\mathbf{z} \in \mathsf{Span}(\mathbf{M})$, the distributions of

$$\varPi \xleftarrow{\$} \mathsf{NProve}'(\mathsf{CRS}, \mathsf{C}, \mathsf{x}, \mathsf{w}) \text{ and } \varPi \xleftarrow{\$} \mathsf{NSim}(\mathsf{CRS}, \mathsf{TD}, \mathsf{C}, \mathsf{x}),$$

where $(\mathsf{CRS}, \mathsf{TD}) \xleftarrow{\$} \mathsf{NTGen}(1^\lambda, \mathsf{par})$ and $\mathsf{C}(\mathsf{x}, \mathsf{w}) = 1$, are identical as well, completing the proof of Theorem 2. □

Theorem 3 (Soundness). *If* ORNIZK *is perfectly sound, then* NIZK *is perfectly sound.*

Proof. To prove perfect soundness, we just have to show that we can extract a valid witness from any proof passing the verification. Let $\mathbf{k}$ be the vector in the kernel of $\mathbf{M}$ such that $\mathbf{k}^\top \mathbf{z} = 1$, which must exist when $\mathbf{z} \notin \mathsf{Span}(\mathbf{M})$. We define an extractor as in Fig. 5. For any valid statement/proof pair $(\mathsf{x}, \varPi)$, we argue that the extractor must be able to extract a valid witness w for x as below.

Due to the perfect soundness of ORNIZK, for each NAND gate with input commitments $(\mathsf{cm}_{d_1}, \mathsf{cm}_{d_2})$ and an output commitment cm_{d_3} in a valid proof, we have

$$\mathsf{x}_{\ell,1} = \begin{pmatrix} -[\mathbf{z}]_1 + \mathsf{cm}_{d_1} + \mathsf{cm}_{d_3} \\ -[\mathbf{z}]_1 + \mathsf{cm}_{d_2} \end{pmatrix} \in \mathsf{Span}([\mathbf{M}']_1)$$

$$\text{or} \quad \mathbf{x}_{\ell,2} = \begin{pmatrix} -[\mathbf{z}]_1 + \mathsf{cm}_{d_3} \\ \mathsf{cm}_{d_2} \end{pmatrix} \in \mathsf{Span}([\mathbf{M}']_1).$$

Then we have

$$\begin{aligned} \mathbf{k}^\top(-[\mathbf{z}]_1 + \mathsf{cm}_{d_1} + \mathsf{cm}_{d_3}) = -[1]_1 + \mathbf{k}^\top \mathsf{cm}_{d_1} + \mathbf{k}^\top \mathsf{cm}_{d_3} = [0]_1 \\ \wedge \mathbf{k}^\top(-[\mathbf{z}]_1 + \mathsf{cm}_{d_2}) = -[1]_1 + \mathbf{k}^\top \mathsf{cm}_{d_2} = [0]_1 \end{aligned}$$

```
Ext_NIZK(C(x, ·), Π):
Initialize the values for all wires in C(x, ·) as ⊥.
Run F_NIZK(k, C, G_t, Π), where G_t is the gate for the final output, to assign
values for each wire in the circuit C(x, ·)
For each input wire with index i assigned ⊥, set w_i = 0
Return the witness w = (w_i)_{i∈[h]} (containing all the bits for input values)
```

Fig. 5. Definition of $\mathsf{Ext}_{\mathsf{NIZK}}$. $\mathsf{F}_{\mathsf{NIZK}}$ is the recursion algorithm defined as in Fig. 6.

```
$\mathsf{F}_{\mathsf{NIZK}}(\mathbf{k}, \mathsf{C}, G_\ell = (d_1, d_2, d_3), \Pi)$:
Compute $\mathsf{temp}_i = \mathbf{k}^\top \mathsf{cm}_i$ for $i = 1, 2, 3$
If $\mathsf{temp}_2 = [0]_1$ and $\mathsf{temp}_3 = [1]_1$:
  – set $(\mathsf{w}_{d_1}, \mathsf{w}_{d_2}) = (\perp, 0)$
  – stop if $\mathsf{Parent}_r(G_\ell) = \perp$
  – run $\mathsf{F}_{\mathsf{NIZK}}(\mathbf{k}, \mathsf{C}, \mathsf{Parent}_r(G_\ell), \Pi)$
If $\mathsf{temp}_2 = [1]_1$ and $[\mathsf{temp}_1 + \mathsf{temp}_3]_1 = [1]_1$:
  – if $\mathsf{temp}_3 \notin \{[0]_1, [1]_1\}$, abort $\mathsf{Ext}_{\mathsf{NIZK}}$
  – set $(\mathsf{w}_{d_1}, \mathsf{w}_{d_2}) = (b, 1)$ where $[b]_1 = \mathsf{temp}_1$ and $b \in \{0, 1\}$
  – stop if $\mathsf{Parent}_l(G_\ell) = \perp$ and $\mathsf{Parent}_r(G_\ell) = \perp$
  – run $\mathsf{F}_{\mathsf{NIZK}}(\mathbf{k}, \mathsf{C}, \mathsf{Parent}_l(G_\ell), \Pi)$ and $\mathsf{F}_{\mathsf{NIZK}}(\mathbf{k}, \mathsf{C}, \mathsf{Parent}_r(G_\ell), \Pi)$
Otherwise, abort $\mathsf{Ext}_{\mathsf{NIZK}}$
```

Fig. 6. Definition of $\mathsf{F}_{\mathsf{NIZK}}$. Parent_l (repsectively, Parent_r) denotes the gate whose output is the left (respectively, right) input to G_ℓ.

$$\text{or } \mathbf{k}^\top(-[\mathbf{z}]_1 + \mathsf{cm}_{d_3}) = -[1]_1 + \mathbf{k}^\top \mathsf{cm}_{d_3} = [0]_1 \wedge \mathbf{k}^\top \mathsf{cm}_{d_2} = [0]_1.$$

Moreover, we must have $\mathbf{k}^\top \mathsf{cm}_t = \mathbf{k}^\top [\mathbf{z}]_1 = [1]_1$ for the output wire. As a result, for a valid proof, $\mathsf{F}_{\mathsf{NIZK}}$ (see Fig. 6) will never abort during the execution of $\mathsf{Ext}_{\mathsf{NIZK}}$, and running $\mathsf{F}_{\mathsf{NIZK}}$ recursively will result in bits for input wires leading the statement circuit to output 1. Notice that after running $\mathsf{F}_{\mathsf{NIZK}}(\mathbf{k}, \mathsf{C}, G_t, \Pi)$, there might be some input wires assigned $\perp$. However, these wires do not affect the final output and we can just assign 0 to them.

As a result, we can extract the bits for all wires consisting of valid input/output pairs for all NAND gates and leading the statement circuit to output 1. Therefore, for all proofs passing the verification, there must exist a valid witness for the statement x, i.e., $\mathsf{x} \in \mathcal{L}_\lambda^{\mathsf{CSAT}}$, completing the proof of Theorem 3.[5] $\square$

Remark on the representation of circuits. We represent the circuits by NAND gates only for conceptual simplicity. In practice, this conversion is unnecessary. For any original circuit represented as AND, OR, NOT gates, the NOT gates are free, and by slightly changing Relation 2 on Page 6, we can directly adopt the OR-proof for AND and OR gates. Concretely, for AND gates, we prove $(\mathsf{w}_{d_1} - \mathsf{w}_{d_3} = 0 \wedge -1 + \mathsf{w}_{d_2} = 0) \ \vee \ (\mathsf{w}_{d_3} = 0 \wedge \mathsf{w}_{d_2} = 0)$, and for OR gates, we prove $(\mathsf{w}_{d_1} - \mathsf{w}_{d_3} = 0 \wedge \mathsf{w}_{d_2} = 0) \ \vee \ (\mathsf{w}_{d_3} = 1 \wedge \mathsf{w}_{d_2} = 1)$. Then our new technique saves overhead for AND and OR gates with our witness-extraction strategy in the same way as for NAND-gates. The same argument can also be made for our BARG given later.

Instantiation of our NIZK. By instantiating the OR-proof system as in Sect. 3.1 under the SXDH assumption, each proof of our NIZK consists of $(2t+8s)$

[5] One can see that our construction is also a NIZK proof of knowledge, i.e., we can generate the extraction key $\mathbf{k}$ along with a binding CRS and use it to extract a valid witness from any valid proof.

elements in $\mathbb{G}_1$ and $10s$ elements in $\mathbb{G}_2$, where t and s are the number of wires and gates in the statement circuit respectively. Compared to the GOS-NIZK given in Appendix A, which requires $(6t + 4s)$ elements in $\mathbb{G}_1$ and $(6t + 6s)$ elements in $\mathbb{G}_2$ for each proof, our proof size is strictly smaller since t must be larger than s in any circuit. Moreover, the numbers of exponentiations and pairing products required in our proving and verification procedures are only $2t + 30s$ and $24s$ respectively, while those in the GOS-NIZK are $18t+16s$ and $12(s+t)$. Notice that when adopting the OR-proof in our construction, the statement $\mathbf{M}'$ determining the language has half of the entries being $[0]_1$. We do not count exponentiations of these entries and pairing products between these entries and other elements in verification, since the computing results can always be fixed as $[0]_1$ or $[0]_T$.

More instantiations. By instantiating the underlying OR-proof system based on the Extended-Kernel Matrix Diffie-Hellman assumption, which holds unconditionally in the generic group model and implied by the discrete logarithm assumption in the algebraic group model, as in Fig. 6 of [15], we can further reduce the OR-proof size used by our construction by 5 elements in $\mathbb{G}_2$, compared to our SXDH based instantiation (see Table 1). While this also works for the GOS-NIZK in Appendix A, its OR-proof size can only be reduced by 3 elements in $\mathbb{G}_2$.

Extension to non-interactive zaps. In [29], Groth, Sahai, and Ostrovsky gave a generic conversion from any NIZK with verifiable correlated key generation into a non-interactive zap, i.e., non-interactive witness-indistinguishability proof systems in the plain model. To date, this is the only known non-interactive zap for NP based on standard assumptions. Here, verifiable correlated key generation refers to the ability to efficiently generate two correlated common reference strings (CRSs) along with one trapdoor. One CRS is binding, meaning it provides perfect soundness, while the other CRS is hiding, meaning it offers perfect zero-knowledge and corresponds to the trapdoor. It is crucial that a PPT adversary cannot distinguish which CRS is hiding when given both of them. Additionally, it is required that a verification algorithm exist such that honestly sampled CRS pairs always pass verification, and for any CRS pair that passes verification, one of them must be binding. We refer the reader to [29] for a detailed description of the conversion method, while we argue that our NIZK proof system satisfies the requirements for verifiable correlated key generation as outlined above. Consequently, it can be converted into a non-interactive zap, thereby improve the construction in [29] without any trade-offs. As far as we know, this results in the most efficient non-interactive zap based on standard assumptions.

To show that our NIZK has verifiable correlated key generation, we first recall that in both our NIZK and the GOS-NIZK, each CRS essentially consists of a CRS from the underlying NIZK and a key for a homomorphic commitment. Since both parts have the same distribution, we can combine them into a single tuple $(\mathsf{par}, [\mathbf{M}]_1, [\mathbf{z}_{\mathsf{bind}}]_1)$, where $\mathsf{par} = (\mathcal{G} \xleftarrow{\$} \mathsf{GGen}(1^\lambda))$, $\mathbf{M} \xleftarrow{\$} \mathcal{D}_k$, and $\mathbf{z}_{\mathsf{bind}} \xleftarrow{\$} \mathbb{Z}_p^{k+1} \backslash \mathsf{Span}(\mathbf{M})$, without compromising security. In the hiding mode, we replace $\mathbf{z}_{\mathsf{bind}}$ with $\mathbf{z}_{\mathsf{hide}} = \mathbf{M}\mathbf{u}$ where $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^k$. We can further change the distribution of $\mathbf{z}_{\mathsf{bind}}$ to that of $\mathbf{z}_{\mathsf{hide}} + \mathbf{f}$, where $\mathbf{f}$ is some fixed vector outside $\mathsf{Span}(\mathbf{M})$. One can easily see that changed $\mathbf{z}_{\mathsf{bind}}$ remains binding due to non-linearility and composable zero-knowledge still holds due to the MDDH assumption. Then we can set the correlated CRSs as $(\mathsf{par}, [\mathbf{M}, \mathbf{z}_{\mathsf{bind}}]_1)$ and $(\mathsf{par}, [\mathbf{M}, \mathbf{z}_{\mathsf{hide}} = \mathbf{z}_{\mathsf{bind}} - \mathbf{f}]_1)$ and set $\mathbf{u}$ as the trapdoor. Due to the MDDH assumption, any PPT adversary cannot tell which one is hiding. The verification algorithm given the two CRSs checks the validity of par and $[\mathbf{M}]_1$ and whether $\mathbf{z}_{\mathsf{bind}} + \mathbf{z}_{\mathsf{hide}} = \mathbf{f}$ holds. For any two CRS $(\mathsf{par}, [\mathbf{M}, \mathbf{z}_0]_1)$ and $(\mathsf{par}, [\mathbf{M}, \mathbf{z}_1]_1)$ passing the verification, we must have either $[\mathbf{z}_0]_1 \notin \mathsf{Span}([\mathbf{M}]_1)$ or $[\mathbf{z}_1]_1 \notin \mathsf{Span}([\mathbf{M}]_1)$, i.e., one of them must be binding. Therefore, our NIZK proof system, as well as the GOS-NIZK in the asymmetric pairing setting, has verifiable correlated key generation.

4 Batch Argument for NP

In this section, we extend our framework for NIZK to give an efficient construction of BARG for batch circuit satisfiability in NP.

Definition 10 (Batch circuit satisfiability). *Let λ be the security parameter. The* batch circuit satisfiability *language for an integer $m \in \mathbb{N}$ is defined as follows.*

$$\mathcal{L}_{\lambda}^{\mathsf{BatchCSAT}} = \{(\mathsf{C}, (\mathsf{x}_i)_{i\in[m]}) | \forall i \in [m] : \exists \mathsf{w}_i \in \{0,1\}^h : C(\mathsf{x}_i, \mathsf{w}_i) = 1\},$$

where $\mathsf{C} : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ is any Boolean circuit with polynomial size in λ and $\mathsf{x}_1, \cdots, \mathsf{x}_m \in \{0,1\}^n$ are the statements. Without loss of generality, we assume that C consists only of fan-in-2 NAND gates.

Let $\mathsf{par} = \mathcal{G}$ be the public parameter, where $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, [1]_1, [1]_2, e) \overset{\$}{\leftarrow} \mathsf{GGen}(1^\lambda)$. We give our BARG for $\mathcal{L}_{\lambda}^{\mathsf{BatchCSAT}}$ in Fig. 7.

Theorem 4 (Completeness). BARG *is complete.*

Proof. **Validity of statement.** Since the first n wires corresponds to the statement, for honestly generated $([\mathbf{u}_d]_1 = \sum_{i\in[m]} \mathsf{w}_{i,d}[\mathbf{a}_i]_1)_{d\in[t]}$ and $([\mathbf{u}_d^*]_1 = \sum_{i\in[m]} \mathsf{x}_{i,d}[\mathbf{a}_i]_1)_{d\in[n]}$, we must have $\mathsf{x}_{i,d} = \mathsf{w}_{i,d}$ for all $i \in [m]$ and $d \in [n]$. Hence, we have $[\mathbf{u}_d]_1 = [\mathbf{u}_d^*]_1$ for all $d \in [n]$. Similarly, we have $[\hat{\mathbf{u}}_d]_2 = [\hat{\mathbf{u}}_d^*]_2$ for all $d \in [n]$. Moreover, when the witnesses are valid, we must have $\mathsf{w}_{i,t} = 1$ for all $i \in [m]$ for the output wire. Hence, we have $[\mathbf{u}_t]_1 = [\sum_{i\in[m]} \mathbf{a}_i]_1 = [\mathbf{a}]_1$ and $[\hat{\mathbf{u}}_t]_2 = [\sum_{i\in[m]} \hat{\mathbf{a}}_i]_2 = [\hat{\mathbf{a}}]_2$.

Validity of gate computation. For witnesses $(\mathsf{w}_i)_{i\in[m]}$, for each gate $G_\ell = (d_1, d_2, d_3)$, we have $(-1 + \mathsf{w}_{i,d_1} + \mathsf{w}_{i,d_3} = 0 \wedge -1 + \mathsf{w}_{i,d_2} = 0)$ or $(-1 + \mathsf{w}_{i,d_3} = 0 \wedge \mathsf{w}_{i,d_2} = 0)$ for all $i \in [m]$, which in turn implies

$$(-1 + \mathsf{w}_{i,d_1} + \mathsf{w}_{i,d_3})\mathsf{w}_{i,d_2} = 0 \text{ and } (-1 + \mathsf{w}_{i,d_3})(-1 + \mathsf{w}_{i,d_2}) = 0.$$

```
$\underline{\mathsf{BGen}(1^\lambda, \mathsf{par}, 1^m):}$
Sample $\mathbf{M}, \hat{\mathbf{M}} \xleftarrow{\$} \mathcal{D}_k$ and $\alpha_i, \hat{\alpha}_i \xleftarrow{\$} \mathbb{Z}_p^k$ for all $i \in [m]$
Set $\mathbf{a}_i = \mathbf{M}\alpha_i$ and $\hat{\mathbf{a}}_i = \hat{\mathbf{M}}\hat{\alpha}_i$ for all $i \in [m]$, $\mathbf{a} = \sum_{i\in[m]} \mathbf{a}_i$, and $\hat{\mathbf{a}} = \sum_{i\in[m]} \hat{\mathbf{a}}_i$
For each $i, j \in [m]$ such that $i \neq j$, sample $\mathbf{R}_{i,j} \xleftarrow{\$} \mathbb{Z}_p^{k\times k}$, and set $\mathbf{B}_{i,j} = \mathbf{M}(\alpha_i\hat{\alpha}_j^\top + \mathbf{R}_{i,j})$ and $\hat{\mathbf{B}}_{i,j} = -\hat{\mathbf{M}}\mathbf{R}_{i,j}^\top$
Return $\mathsf{crs} = (\mathsf{par}, [\mathbf{M}]_1, [\hat{\mathbf{M}}]_2, [\mathbf{a}]_1, [\hat{\mathbf{a}}]_2, ([\mathbf{a}_i]_1, [\hat{\mathbf{a}}_i]_2)_{i\in[m]}, \{[\mathbf{B}_{i,j}]_1, [\hat{\mathbf{B}}_{i,j}]_2\}_{i\neq j})$

$\underline{\mathsf{BProve}(\mathsf{crs}, \mathsf{C} : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}, (\mathsf{x}_i)_{i\in[m]}, (\mathsf{w}_i)_{i\in[m]}):}$
Define $s$ and $t$ to be the numbers of gates and wires of $\mathsf{C}$ respectively
For all $i \in [m]$, extend $(\mathsf{x}_i, \mathsf{w}_i)$ to $(\mathsf{w}_{i,j})_{j\in[t]}$ containing the bits of all wires in $\mathsf{C}$
For each $d \in [t]$, set $[\mathbf{u}_d]_1 = \sum_{i\in[m]} \mathsf{w}_{i,d}[\mathbf{a}_i]_1$ and $[\hat{\mathbf{u}}_d]_2 = \sum_{i\in[m]} \mathsf{w}_{i,d}[\hat{\mathbf{a}}_i]_2$
For each gate $G_\ell = (d_1, d_2, d_3) \in [t]^3$ where $\ell \in [s]$, set
  – $[\mathbf{V}_{\ell,1}]_1 = \sum_{i\neq j}(1 - \mathsf{w}_{i,d_1} - \mathsf{w}_{i,d_3})\mathsf{w}_{j,d_2}[\mathbf{B}_{i,j}]_1$
  – $[\hat{\mathbf{V}}_{\ell,1}]_2 = \sum_{i\neq j}(1 - \mathsf{w}_{i,d_1} - \mathsf{w}_{i,d_3})\mathsf{w}_{j,d_2}[\hat{\mathbf{B}}_{i,j}]_2$
  – $[\mathbf{V}_{\ell,2}]_1 = \sum_{i\neq j}\Big(\mathsf{w}_{i,d_2} - (\mathsf{w}_{i,d_1} + \mathsf{w}_{i,d_3})\mathsf{w}_{j,d_2}\Big)[\mathbf{B}_{i,j}]_1$
  – $[\hat{\mathbf{V}}_{\ell,2}]_2 = \sum_{i\neq j}\Big(\mathsf{w}_{i,d_2} - (\mathsf{w}_{i,d_1} + \mathsf{w}_{i,d_3})\mathsf{w}_{j,d_2}\Big)[\hat{\mathbf{B}}_{i,j}]_2$
  – $[\mathbf{W}_\ell]_1 = \sum_{i\neq j}(1 - \mathsf{w}_{i,d_3})(1 - \mathsf{w}_{j,d_2})[\mathbf{B}_{i,j}]_1$
  – $[\hat{\mathbf{W}}_\ell]_2 = \sum_{i\neq j}(1 - \mathsf{w}_{i,d_3})(1 - \mathsf{w}_{j,d_2})[\hat{\mathbf{B}}_{i,j}]_2$
Return $\Pi = (([\mathbf{u}_d]_1, [\hat{\mathbf{u}}_d]_2)_{d\in[t]}, ([\mathbf{V}_{\ell,i}]_1, [\hat{\mathbf{V}}_{\ell,i}]_2)_{\ell\in[s],i\in[2]}, ([\mathbf{W}_\ell]_1, [\hat{\mathbf{W}}_\ell]_2)_{\ell\in[s]})$

$\underline{\mathsf{BVer}(\mathsf{crs}, \mathsf{C}, (\mathsf{x}_i)_{i\in[m]}, \Pi):}$

  – $\underline{\mathsf{GenVK}(\mathsf{crs}, (\mathsf{x}_i)_{i\in[m]}):}$
    Parse $\mathsf{x}_i = (\mathsf{x}_{i,1}, \cdots, \mathsf{x}_{i,n})$ for all $i \in [m]$
    For each $d \in [n]$, set $[\mathbf{u}_d^*]_1 = \sum_{i\in[m]} \mathsf{x}_{i,d}[\mathbf{a}_i]_1$ and $[\hat{\mathbf{u}}_d^*]_2 = \sum_{i\in[m]} \mathsf{x}_{i,d}[\hat{\mathbf{a}}_i]_2$
    Output $\mathsf{vk} = ([\mathbf{u}_d^*]_1, [\hat{\mathbf{u}}_d^*]_2)_{d\in[n]}$

  – $\underline{\mathsf{OnlineVer}(\mathsf{vk}, \mathsf{C}, \Pi):}$
    Check that $[\mathbf{u}_d]_1 = [\mathbf{u}_d^*]_1$ and $[\hat{\mathbf{u}}_d]_2 = [\hat{\mathbf{u}}_d^*]_2$ for all $d \in [n]$
    Check that $[\mathbf{u}_t]_1 = [\mathbf{a}]_1$ and $[\hat{\mathbf{u}}_t]_2 = [\hat{\mathbf{a}}]_2$
    For all $\ell \in [s]$, check that
      • $[\mathbf{a} - \mathbf{u}_{d_1} - \mathbf{u}_{d_3}]_1 \circ [\hat{\mathbf{u}}_{d_2}^\top]_2 = [\mathbf{M}]_1 \circ [\hat{\mathbf{V}}_{\ell,1}^\top]_2 + [\mathbf{V}_{\ell,1}]_1 \circ [\hat{\mathbf{M}}^\top]_2$
      • $[\mathbf{u}_{d_2}]_1 \circ [\hat{\mathbf{a}}^\top]_2 - [\mathbf{u}_{d_1} + \mathbf{u}_{d_3}]_1 \circ [\hat{\mathbf{u}}_{d_2}^\top]_2 = [\mathbf{M}]_1 \circ [\hat{\mathbf{V}}_{\ell,2}^\top]_2 + [\mathbf{V}_{\ell,2}]_1 \circ [\hat{\mathbf{M}}^\top]_2$
      • $[\mathbf{a} - \mathbf{u}_{d_3}]_1 \circ [\hat{\mathbf{a}}^\top - \hat{\mathbf{u}}_{d_2}^\top]_2 = [\mathbf{M}]_1 \circ [\hat{\mathbf{W}}_\ell^\top]_2 + [\mathbf{W}_\ell]_1 \circ [\hat{\mathbf{M}}^\top]_2$
    Return 1 iff all checks pass
```

Fig. 7. Definition of $\mathsf{BARG} = (\mathsf{BGen}, \mathsf{BProve}, \mathsf{BVer})$.

Moreover, for the CRS, we have

$$\mathbf{B}_{i,j}\hat{\mathbf{M}}^\top + \mathbf{M}\hat{\mathbf{B}}_{i,j}^\top = \mathbf{M}(\alpha_i\hat{\alpha}_j^\top + \mathbf{R}_{i,j})\hat{\mathbf{M}}^\top - \mathbf{M}\mathbf{R}_{i,j}\hat{\mathbf{M}}^\top = \mathbf{M}\alpha_i\hat{\alpha}_j^\top\hat{\mathbf{M}}^\top = \mathbf{a}_i\hat{\mathbf{a}}_j^\top.$$

Then for $((\mathbf{u}_d, \hat{\mathbf{u}}_d)_{d\in[t]}, ([\mathbf{V}_{\ell,i}, \mathbf{W}_\ell]_1, [\hat{\mathbf{V}}_{\ell,i}, \hat{\mathbf{W}}_\ell]_2)_{\ell\in[s],i\in[2]})$ in a valid proof, we have

$$\begin{aligned}
(\mathbf{a} - \mathbf{u}_{d_1} - \mathbf{u}_{d_3})\hat{\mathbf{u}}_{d_2}^\top &= \sum_{i\in[m]}(1 - \mathsf{w}_{i,d_1} - \mathsf{w}_{i,d_3})\mathbf{a}_i \sum_{i\in[m]}\mathsf{w}_{i,d_2}\hat{\mathbf{a}}_i^\top \\
&= \Big(\underbrace{\sum_{i\in[m]}(1 - \mathsf{w}_{i,d_1} - \mathsf{w}_{i,d_3})\mathsf{w}_{i,d_2}\mathbf{a}_i\hat{\mathbf{a}}_i^\top}_{=0} + \sum_{i\neq j}(1 - \mathsf{w}_{i,d_1} - \mathsf{w}_{i,d_3})\mathsf{w}_{j,d_2}\mathbf{a}_i\hat{\mathbf{a}}_j^\top\Big) \\
&= \sum_{i\neq j}(1 - \mathsf{w}_{i,d_1} - \mathsf{w}_{i,d_3})\mathsf{w}_{j,d_2}\underbrace{(\mathbf{B}_{i,j}\hat{\mathbf{M}}^\top + \mathbf{M}\hat{\mathbf{B}}_{i,j}^\top)}_{=\mathbf{a}_i\hat{\mathbf{a}}_j^\top} = \mathbf{M}\hat{\mathbf{V}}_{\ell,1}^\top + \mathbf{V}_{\ell,1}\hat{\mathbf{M}}^\top,
\end{aligned}$$

$$\begin{aligned}
&\mathbf{u}_{d_2}\hat{\mathbf{a}}^\top - (\mathbf{u}_{d_1} + \mathbf{u}_{d_3})\hat{\mathbf{u}}_{d_2}^\top \\
&= \sum_{i\in[m]}\mathsf{w}_{i,d_2}\mathbf{a}_i\sum_{i\in[m]}\hat{\mathbf{a}}_i^\top - \sum_{i\in[m]}(\mathsf{w}_{i,d_1} + \mathsf{w}_{i,d_3})\mathbf{a}_i\sum_{i\in[m]}\mathsf{w}_{i,d_2}\hat{\mathbf{a}}_i^\top \\
&= \Big(\underbrace{\sum_{i\in[m]}(1 - \mathsf{w}_{i,d_1} - \mathsf{w}_{i,d_3})\mathsf{w}_{i,d_2}\mathbf{a}_i\hat{\mathbf{a}}_i^\top}_{=0} + \sum_{i\neq j}\big(\mathsf{w}_{i,d_2} - (\mathsf{w}_{i,d_1} + \mathsf{w}_{i,d_3})\mathsf{w}_{j,d_2}\big)\mathbf{a}_i\hat{\mathbf{a}}_j^\top\Big) \\
&= \sum_{i\neq j}\big(\mathsf{w}_{i,d_2} - (\mathsf{w}_{i,d_1} + \mathsf{w}_{i,d_3})\mathsf{w}_{j,d_2}\big)\underbrace{(\mathbf{B}_{i,j}\hat{\mathbf{M}}^\top + \mathbf{M}\hat{\mathbf{B}}_{i,j}^\top)}_{=\mathbf{a}_i\hat{\mathbf{a}}_j^\top} = \mathbf{M}\hat{\mathbf{V}}_{\ell,2}^\top + \mathbf{V}_{\ell,2}\hat{\mathbf{M}}^\top,
\end{aligned}$$

$$\begin{aligned}
&(\mathbf{a} - \mathbf{u}_{d_3})(\hat{\mathbf{a}}^\top - \hat{\mathbf{u}}_{d_2}^\top) \\
&= \sum_{i\in[m]}(1 - \mathsf{w}_{i,d_3})\mathbf{a}_i\sum_{i\in[m]}(1 - \mathsf{w}_{i,d_2})\hat{\mathbf{a}}_i^\top \\
&= \Big(\underbrace{\sum_{i\in[m]}(1 - \mathsf{w}_{i,d_3})(1 - \mathsf{w}_{i,d_2})\mathbf{a}_i\hat{\mathbf{a}}_i^\top}_{=0} + \sum_{i\neq j}(1 - \mathsf{w}_{i,d_3})(1 - \mathsf{w}_{j,d_2})\mathbf{a}_i\hat{\mathbf{a}}_j^\top\Big) \\
&= \sum_{i\neq j}(1 - \mathsf{w}_{i,d_3})(1 - \mathsf{w}_{j,d_2})\underbrace{(\mathbf{B}_{i,j}\hat{\mathbf{M}}^\top + \mathbf{M}\hat{\mathbf{B}}_{i,j}^\top)}_{=\mathbf{a}_i\hat{\mathbf{a}}_j^\top} = \mathbf{M}\hat{\mathbf{W}}_\ell^\top + \mathbf{W}_\ell\hat{\mathbf{M}}^\top.
\end{aligned}$$

This completes the proof of completeness. □

Theorem 5 (Succinctness). BARG *is succinct.*

Proof. For our BARG in Fig. 7, we check the succinctness as follows.

Proof size. Each proof π consists of $t(k+1) + 3s(k+1)k$ group elements in each of $\mathbb{G}_1$ and $\mathbb{G}_2$, where each group element can be represented in $\mathsf{poly}(\lambda)$ bits and k is constant. Since $t = \mathsf{poly}(s)$, we have $|\pi| = \mathsf{poly}(\lambda, s)$.

CRS size. Each CRS crs consists of the group description and $(k+1)k + (m+1)(k+1) + m(m-1)/2 \cdot (k+1)k = O(k^2m^2)$ elements in each of $\mathbb{G}_1$ and $\mathbb{G}_2$. Thus we have $|\mathsf{crs}| = m^2 \cdot \mathsf{poly}(\lambda)$.

Verification key. Each verification key vk output by GenVK consists of $n(k+1)$ elements in each of $\mathbb{G}_1$ and $\mathbb{G}_2$. Thus we have $|\mathsf{vk}| = n \cdot \mathsf{poly}(\lambda)$.

Verification key generation time. GenVK performs $2mn(k+1)$ group operations, which requires $\mathsf{poly}(\lambda, m, n)$ time.

Online verification time. The $\mathsf{OnlineVer}$ consists of 3 steps in total, where the running time of each step is bounded by $nk \cdot \mathsf{poly}(\lambda)$, $k \cdot \mathsf{poly}(\lambda)$, and $sk^3 \cdot \mathsf{poly}(\lambda)$ respectively. Since $n = \mathsf{poly}(s)$, the total running time is bounded by $\mathsf{poly}(s, \lambda)$

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

Theorem 6 (Somewhere argument of knowledge). *Under the* $\mathcal{D}_k$-MDDH *assumption,* BARG *is a somewhere argument of knowledge.*

Proof. We define the trapdoor setup and extraction algorithms as in Fig. 8.

CRS indistinguishability. We prove the CRS indistinguishability by defining a sequence of intermediate games.

Let $\mathcal{A}$ be any PPT adversary against the CRS indistinguishability of BARG for some index $i^* \in [m]$. It receives a CRS crs generated by the challenger $\mathcal{CH}$ in each game as defined in Fig. 10.

Game G_0 and G_1. Game G_0 is the game where $\mathcal{CH}$ on receiving the index i^* from the adversary returns crs generated as $\mathsf{crs} \overset{\$}{\leftarrow} \mathsf{BGen}(1^\lambda, \mathsf{par}, 1^m)$ to $\mathcal{A}$. Game G_1 is exactly the same as G_0 except that $\mathbf{B}_{i,j}$ and $\hat{\mathbf{B}}_{i,j}$ are generated in a different way. □

Lemma 2. $\Pr[\mathsf{G}_0^{\mathcal{A}} \Rightarrow 1] = \Pr[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1]$.

Proof. For $j \neq i^*$, the distributions of $\mathbf{B}_{i,j}$ in Games G_0 and G_1 are identical, since

$$\mathbf{M}(\alpha_i\hat{\alpha}_j^\top + \mathbf{R}_{i,j}) = (\mathbf{M}\alpha_i)\hat{\alpha}_j^\top + \mathbf{M}\mathbf{R}_{i,j} = \mathbf{a}_i\hat{\alpha}_j^\top + \mathbf{M}\mathbf{R}_{i,j}.$$

For $j = i^*$, in G_0, we have $\mathbf{B}_{i,j} = \mathbf{M}(\alpha_i\hat{\alpha}_j^\top + \mathbf{R}_{i,j})$ and

$$\hat{\mathbf{B}}_{i,j} = -\hat{\mathbf{M}}\mathbf{R}_{i,j}^\top = \hat{\mathbf{M}}(\alpha_i\hat{\alpha}_j^\top)^\top - \hat{\mathbf{M}}(\mathbf{R}_{i,j} + \alpha_i\hat{\alpha}_j^\top)^\top = \hat{\mathbf{a}}_j\alpha_i^\top - \hat{\mathbf{M}}(\mathbf{R}_{i,j} + \alpha_i\hat{\alpha}_j^\top).$$

Since the distribution of $\mathbf{R}_{i,j} + \alpha_i\hat{\alpha}_j^\top$ is uniformly distributed, the distribution of $\mathbf{B}_{i,j}$ and $\hat{\mathbf{B}}_{i,j}$ is identical to that in G_1, completing this part of proof. □

```
$\underline{\mathsf{BTGen}(1^\lambda, \mathsf{par}, 1^m, i^*)}$:
Sample $\mathbf{M}, \hat{\mathbf{M}} \xleftarrow{\$} \mathcal{D}_k$ and $\alpha_i, \hat{\alpha}_i \xleftarrow{\$} \mathbb{Z}_p^k$ for all $i \neq i^*$
Sample $\mathbf{a}_{i^*} \xleftarrow{\$} \mathbb{Z}_p^{k+1} \backslash \mathsf{Span}(\mathbf{M})$ and $\hat{\mathbf{a}}_{i^*} \xleftarrow{\$} \mathbb{Z}_p^{k+1} \backslash \mathsf{Span}(\hat{\mathbf{M}})$
Set $\mathbf{a}_i = \mathbf{M}\alpha_i$ and $\hat{\mathbf{a}}_i = \hat{\mathbf{M}}\hat{\alpha}_i$ for all $i \neq i^*$, $\mathbf{a} = \sum_{i \in [m]} \mathbf{a}_i$, and $\hat{\mathbf{a}} = \sum_{i \in [m]} \hat{\mathbf{a}}_i$
For each $i, j \in [m]$ such that $i \neq j$,
  – sample $\mathbf{R}_{i,j} \xleftarrow{\$} \mathbb{Z}_p^{k \times k}$
  – set $\mathbf{B}_{i,j} = \mathbf{a}_i \hat{\alpha}_j^\top + \mathbf{M}\mathbf{R}_{i,j}$ and $\hat{\mathbf{B}}_{i,j} = -\hat{\mathbf{M}}\mathbf{R}_{i,j}^\top$ for all $i, j \in [m]$ and $j \neq i^*$
  – set $\mathbf{B}_{i,j} = \mathbf{M}\mathbf{R}_{i,j}$ and $\hat{\mathbf{B}}_{i,j} = \hat{\mathbf{a}}_j \alpha_i^\top - \hat{\mathbf{M}}\mathbf{R}_{i,j}^\top$ for all $i \in [m]$ and $j = i^*$
Compute a non-zero vector $\tau \in \mathbb{Z}_p^{k+1}$ such that $\tau^\top \mathbf{M} = 0$ and $\tau^\top \mathbf{a}_{i^*} = 1$, which
must exist and can be efficiently computed since $\mathbf{M}$ is of rank $k$
Return $\mathsf{crs} = (\mathsf{par}, [\mathbf{M}]_1, [\hat{\mathbf{M}}]_2, [\mathbf{a}]_1, [\hat{\mathbf{a}}]_2, ([\mathbf{a}_i]_1, [\hat{\mathbf{a}}_i]_2)_{i \in [m]}, \{[\mathbf{B}_{i,j}]_1, [\hat{\mathbf{B}}_{i,j}]_2\}_{i \neq j})$ and
$\mathsf{td} = \tau$

$\underline{\mathsf{Ext}_{\mathsf{BARG}}(\mathsf{td}, \mathsf{C}, (\mathsf{x}_i)_{i \in [m]}, \Pi)}$:
Initialize the values for all wires in $\mathsf{C}(\mathsf{x}_{i^*}, \cdot)$ as $\perp$.
Run $\mathsf{F}_{\mathsf{BARG}}(\mathsf{td}, \mathsf{C}, G_t, \Pi)$, where $G_t$ is the gate for the final output, to assign values
for each wire in the circuit $\mathsf{C}(\mathsf{x}_{i^*}, \cdot)$
For each input wire $d \in [t]$ assigned with $\perp$, set $\mathsf{w}_{i^*,d} = 0$
Return the witness $\mathsf{w}_{i^*} = (\mathsf{w}_{i^*,i})_{i \in [h]}$ containing all bits for input values of $\mathsf{C}(\mathsf{x}_{i^*}, \cdot)$
```

Fig. 8. Definition of $(\mathsf{BTGen}, \mathsf{Ext}_{\mathsf{BARG}})$. $\mathsf{F}_{\mathsf{BARG}}$ is the recursion algorithm defined as in Fig. 9.

```
$\underline{\mathsf{F}_{\mathsf{BARG}}(\mathsf{td}, \mathsf{C}, G_\ell = (d_1, d_2, d_3), \Pi)}$:
Compute $\mathsf{temp}_i = \tau^\top [\mathbf{u}_{d_i}]_1$ for all $i = 1, 2, 3$
If $\mathsf{temp}_2 = [0]_1$ and $\mathsf{temp}_3 = [1]_1$:
  – set $(\mathsf{w}_{i^*,d_1}, \mathsf{w}_{i^*,d_2}) = (\perp, 0)$
  – stop if $\mathsf{Parent}_r(G_\ell) = \perp$, and run $\mathsf{F}_{\mathsf{BARG}}(\mathsf{td}, \mathsf{C}, \mathsf{Parent}_r(G_\ell), \Pi)$ otherwise
If $\mathsf{temp}_2 = [1]_1$ and $\mathsf{temp}_1 + \mathsf{temp}_3 = [1]_1$:
  – if $\mathsf{temp}_3 \notin \{[0]_1, [1]_1\}$, abort $\mathsf{Ext}_{\mathsf{BARG}}$
  – set $(\mathsf{w}_{i^*,d_1}, \mathsf{w}_{i^*,d_2}) = (b, 1)$ where $[b]_1 = \mathsf{temp}_1$ and $b \in \{0, 1\}$
  – stop if $\mathsf{Parent}_l(G_\ell) = \perp$ and $\mathsf{Parent}_r(G_\ell) = \perp$
  – run $\mathsf{F}_{\mathsf{BARG}}(\mathsf{td}, \mathsf{C}, \mathsf{Parent}_l(G_\ell), \Pi)$ and $\mathsf{F}_{\mathsf{BARG}}(\mathsf{td}, \mathsf{C}, \mathsf{Parent}_r(G_\ell), \Pi)$
If $\mathsf{temp}_1 = [0]_1$ and $\mathsf{temp}_3 = [1]_1$:
  – set $(\mathsf{w}_{i^*,d_1}, \mathsf{w}_{i^*,d_2}) = (0, \perp)$
  – stop if $\mathsf{Parent}_l(G_\ell) = \perp$
  – run $\mathsf{F}_{\mathsf{BARG}}(\mathsf{td}, \mathsf{C}, \mathsf{Parent}_l(G_\ell), \Pi)$ otherwise
Otherwise, abort $\mathsf{Ext}_{\mathsf{BARG}}$
```

Fig. 9. Definition of $\mathsf{F}_{\mathsf{BARG}}$. Parent_l (repsectively, Parent_r) denotes the gate whose output is the left (respectively, right) input to G_ℓ.

Game G_2. G_2 is the same as G_1 except that $\mathbf{a}_{i^*}$ is randomly sampled outside the span of $\mathbf{M}$.

Lemma 3. *There exists an adversary $\mathcal{B}_1$ breaking the $\mathcal{D}_k$-MDDH assumption in $\mathbb{G}_1$ with probability at least* $|\Pr[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1] - \Pr[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1]| - 1/p$.

```
$\underline{\mathcal{CH}(\mathsf{par}, 1^m, i^*)}$: $\mathsf{G}_0$, $\boxed{\mathsf{G}_1}$, $\mathsf{G}_2$, $\mathsf{G}_3$

Sample $\mathbf{M}, \hat{\mathbf{M}} \xleftarrow{\$} \mathcal{D}_k$ and $\alpha_i, \hat{\alpha}_i \xleftarrow{\$}$ for all $i \in [m]$
Set $\mathbf{a}_i = \mathbf{M}\alpha_i$ for all $i \in [m]$
Set $\mathbf{a}_i = \mathbf{M}\alpha_i$ for all $i \neq i^*$ and sample $\mathbf{a}_{i^*} \xleftarrow{\$} \mathbb{Z}_p^{k+1} \backslash \mathsf{Span}(\mathbf{M})$
Set $\hat{\mathbf{a}}_i = \hat{\mathbf{M}}\hat{\alpha}_i$ for all $i \in [m]$
Set $\hat{\mathbf{a}}_i = \hat{\mathbf{M}}\hat{\alpha}_i$ for all $i \neq i^*$ and sample $\hat{\mathbf{a}}_{i^*} \xleftarrow{\$} \mathbb{Z}_p^{k+1} \backslash \mathsf{Span}(\hat{\mathbf{M}})$
Set $\mathbf{a} = \sum_{i \in [m]} \mathbf{a}_i$ and $\hat{\mathbf{a}} = \sum_{i \in [m]} \hat{\mathbf{a}}_i$
For each $i, j \in [m]$ such that $i \neq j$,
  – sample $\mathbf{R}_{i,j} \xleftarrow{\$} \mathbb{Z}_p^{k \times k}$
  – set $\mathbf{B}_{i,j} = \mathbf{M}(\alpha_i \hat{\alpha}_j^\top + \mathbf{R}_{i,j})$ and $\hat{\mathbf{B}}_{i,j} = -\hat{\mathbf{M}}\mathbf{R}_{i,j}^\top$
  – $\boxed{\text{set } \mathbf{B}_{i,j} = \mathbf{a}_i \hat{\alpha}_j^\top + \mathbf{M}\mathbf{R}_{i,j} \text{ and } \hat{\mathbf{B}}_{i,j} = -\hat{\mathbf{M}}\mathbf{R}_{i,j}^\top \text{ for all } j \neq i^*}$
  – $\boxed{\text{set } \mathbf{B}_{i,j} = \mathbf{M}\mathbf{R}_{i,j} \text{ and } \hat{\mathbf{B}}_{i,j} = \hat{\mathbf{a}}_j \alpha_i^\top - \hat{\mathbf{M}}\mathbf{R}_{i,j}^\top \text{ for } j = i^*}$
Return $\mathsf{crs} = (\mathsf{par}, [\mathbf{M}]_1, [\hat{\mathbf{M}}]_2, [\mathbf{a}]_1, [\hat{\mathbf{a}}]_2, ([\mathbf{a}_i]_1, [\hat{\mathbf{a}}_i]_2)_{i \in [m]}, \{[\mathbf{B}_{i,j}]_1, [\hat{\mathbf{B}}_{i,j}]_2\}_{i \neq j})$
```

Fig. 10. Challenger $\mathcal{CH}$ in the intermediate games.

Proof. We build $\mathcal{B}_1$ as follows.

$\mathcal{B}_1$ runs in exactly the same way as the challenger of G_1 except that instead of generating $[\mathbf{a}_{i^*}]_1$ by itself, it takes as input $[\mathbf{a}_{i^*}]_1$ generated as $\mathbf{a}_{i^*} \xleftarrow{\$} \mathbb{Z}_p^{k+1}$ or $\mathbf{a}_{i^*} = \mathbf{M}\alpha_{i^*}$ where $\alpha_{i^*} \xleftarrow{\$} \mathbb{Z}_p^k$ from its own challenger. When $\mathcal{A}$ outputs $\beta \in \{0, 1\}$, $\mathcal{B}_1$ outputs β as well.

If $\mathbf{a}_{i^*}$ is generated as $\mathbf{a}_{i^*} = \mathbf{M}\alpha_{i^*}$ where $\alpha_{i^*} \xleftarrow{\$} \mathbb{Z}_p^k$, the view of $\mathcal{A}$ is the same as its view in G_1. Otherwise, the view of $\mathcal{A}$ is $1/p$-statistically close to its view in G_2. Hence, the probability that $\mathcal{B}_1$ breaks the $\mathcal{D}_k$-MDDH assumption is at least $|\Pr[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1] - \Pr[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1]| - 1/p$, completing this part of proof. $\square$

Game G_3. G_3 is the game $\mathcal{CH}$ returns crs generated by $\mathsf{BTGen}(1^\lambda, \mathsf{par}, 1^m, i^*)$. It is exactly the same as G_2 except that $\hat{\mathbf{a}}_{i^*}$ is randomly sampled outside the span of $\hat{\mathbf{M}}$.

Lemma 4. *There exists an adversary* $\mathcal{B}_2$ *breaking the* $\mathcal{D}_k$-MDDH *assumption in* $\mathbb{G}_2$ *with probability at least* $|\Pr[\mathsf{G}_3^{\mathcal{A}} \Rightarrow 1] - \Pr[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1]| - 1/p$.

Proof. We build $\mathcal{B}_2$ as follows.

$\mathcal{B}_2$ runs in exactly the same way as the challenger of G_2 except that instead of generating $[\hat{\mathbf{a}}_{i^*}]_2$ by itself, it takes as input $[\hat{\mathbf{a}}_{i^*}]_2$ generated as $\hat{\mathbf{a}}_{i^*} \xleftarrow{\$} \mathbb{Z}_p^{k+1}$ or $\hat{\mathbf{a}}_{i^*} = \hat{\mathbf{M}}\hat{\alpha}_{i^*}$ where $\hat{\alpha}_{i^*} \xleftarrow{\$} \mathbb{Z}_p^k$ from its own challenger. When $\mathcal{A}$ outputs $\beta \in \{0, 1\}$, $\mathcal{B}_2$ outputs β as well.

If $\hat{\mathbf{a}}_{i^*}$ is generated as $\hat{\mathbf{a}}_{i^*} = \hat{\mathbf{M}}\hat{\alpha}_{i^*}$ where $\hat{\alpha}_{i^*} \xleftarrow{\$} \mathbb{Z}_p^k$, the view of $\mathcal{A}$ is the same as its view in G_2. Otherwise, the view of $\mathcal{A}$ is $1/p$-statistically close to G_3. Hence, the probability that $\mathcal{B}_2$ breaks the k-MDDH assumption is $|\Pr[\mathsf{G}_3^{\mathcal{A}} \Rightarrow 1] - \Pr[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1]| - 1/p$, completing this part of proof. $\square$

Putting all the above together, the CRS indistinguishability of BARG immediately follows.

Somewhere extractability in the trapdoor mode. We now argue that for any valid statement/proof pair $((\mathsf{x}_i)_{i\in[m]}, \Pi)$, the extractor must be able to extract a valid witness w_{i^*} for x_{i^*}.

For each NAND gate G_ℓ with commitments $(\mathbf{u}_{d_i}, \hat{\mathbf{u}}_{d_i})_{i\in[3]}$ and proof $(([\mathbf{V}_{\ell,i}]_1, [\hat{\mathbf{V}}_{\ell,i}]_2)_{i\in[2]}, [\mathbf{W}_\ell, \hat{\mathbf{W}}_\ell]_1)$, we have

$$[\mathbf{a} - \mathbf{u}_{d_1} - \mathbf{u}_{d_3}]_1 \circ [\hat{\mathbf{u}}_{d_2}^\top]_2 = [\mathbf{M}]_1 \circ [\hat{\mathbf{V}}_{\ell,1}^\top]_2 + [\mathbf{V}_{\ell,1}]_1 \circ [\hat{\mathbf{M}}^\top]_2,$$

$$[\mathbf{u}_{d_2}]_1 \circ [\hat{\mathbf{a}}^\top]_2 - [\mathbf{u}_{d_1} + \mathbf{u}_{d_3}]_1 \circ [\hat{\mathbf{u}}_{d_2}^\top]_2 = [\mathbf{M}]_1 \circ [\hat{\mathbf{V}}_{\ell,2}^\top]_2 + [\mathbf{V}_{\ell,2}]_1 \circ [\hat{\mathbf{M}}^\top]_2,$$

$$[\mathbf{a} - \mathbf{u}_{d_3}]_1 \circ [\hat{\mathbf{a}}^\top - \hat{\mathbf{u}}_{d_2}^\top]_2 = [\mathbf{M}]_1 \circ [\hat{\mathbf{W}}_{\ell,1}^\top]_2 + [\mathbf{W}_{\ell,1}]_1 \circ [\hat{\mathbf{M}}^\top]_2.$$

Recall that τ is the trapdoor in Fig. 8, and let $\hat{\tau}$ be the vector in the kernel of $\hat{\mathbf{M}}$ such that $\hat{\tau}^\top \hat{\mathbf{a}}_{i^*} = 1$, which must exist when $\hat{\mathbf{a}}_{i^*} \notin \mathsf{Span}(\hat{\mathbf{M}})$. Since $\tau^\top \mathbf{a} = \hat{\tau}^\top \hat{\mathbf{a}} = 1$ and $\tau^\top \mathbf{M} = \hat{\tau}^\top \hat{\mathbf{M}}$, where τ is the trapdoor in Fig. 8, the above equations imply

$$[1 - \tau^\top \mathbf{u}_{d_1} - \tau^\top \mathbf{u}_{d_3}]_1 \circ [\hat{\mathbf{u}}_{d_2}^\top \hat{\tau}]_2 = [0]_T \tag{4}$$

$$[\tau^\top \mathbf{u}_{d_2}]_1 \circ [1]_2 - [\tau^\top \mathbf{u}_{d_1} + \tau^\top \mathbf{u}_{d_3}]_1 \circ [\hat{\mathbf{u}}_{d_2}^\top \hat{\tau}]_2 = [0]_T, \tag{5}$$

$$[1 - \tau^\top \mathbf{u}_{d_3}]_1 \circ [1 - \hat{\mathbf{u}}_{d_2}^\top \hat{\tau}]_2 = [0]_T. \tag{6}$$

The quotient of the Eqs. (4) and (5) yields $[\tau^\top \mathbf{u}_{d_2}]_T = [\hat{\mathbf{u}}_{d_2}^\top \hat{\tau}]_T$. Then, combining Eqs. (4) and (6) yields $1 - \tau^\top \mathbf{u}_{d_1} - \tau^\top \mathbf{u}_{d_3} = 0 \wedge 1 - \tau^\top \mathbf{u}_{d_2} = 0$ or $1 - \tau^\top \mathbf{u}_{d_3} = 0 \wedge \tau^\top \mathbf{u}_{d_2} = 0$ or $1 - \tau^\top \mathbf{u}_{d_1} - \tau^\top \mathbf{u}_{d_3} = 0 \wedge 1 - \tau^\top \mathbf{u}_{d_3} = 0$, i.e., $1 - \tau^\top \mathbf{u}_{d_3} = 0 \wedge \tau^\top \mathbf{u}_{d_1} = 0$. Moreover, we must have $\tau^\top \mathbf{u}_{d_t} = \tau^\top [\mathbf{a}]_1 = [1]_1$ for the output wire. As a result, for a valid proof, $\mathsf{F}_{\mathsf{BARG}}$ (see Fig. 9) will never abort during the execution of $\mathsf{Ext}_{\mathsf{BARG}}$, and running $\mathsf{F}_{\mathsf{BARG}}$ recursively will result in bits for input wires leading the statement circuit to output 1. Notice that after running $\mathsf{F}_{\mathsf{BARG}}(\mathsf{td}, \mathsf{C}, G_t, \Pi)$, there might be some input wires assigned with $\perp$. However, these wires do not affect the final output and can be assigned with 0.

As a result, we can extract the bits for all wires consisting of valid input/output pairs for all NAND gates and leading the statement circuit to output 1, completing the proof of perfect soundness.

Putting all the above together, Theorem 6 immediately follows. $\square$

Proof size and proving and online verification cost. By instantiating our construction under the SXDH assumption, each proof of our BARG consists of $(2t+6s)$ elements in both $\mathbb{G}_1$ and $\mathbb{G}_2$, where t and s are the numbers of wires and gates in the statement circuit respectively. The proof size is strictly smaller than that of WW-BARG, which require $(4t + 4s)$ elements in both $\mathbb{G}_1$ and $\mathbb{G}_2$. Moreover, the proving and online verification procedures in our construction require only $4mt + 6m(m-1)s$ multiplications and $40s$ pairing products respectively. In contrast, those in WW-BARG require $4m^2t + 4m(m-1)s$ multiplications and $24t + 32s$ pairing products (after merging items with multiplication in $\mathbb{G}_1$ and $\mathbb{G}_2$).

Construction in the symmetric pairing. Transplanting our construction to the setting of symmetric composite-order pairing groups yields a BARG under the subgroup decision assumption. Compared to the WW-BARG, we reduce the proof size by $(2t + s) - (t + 2s) = t - s$ group elements in $\mathbb{G}$. Also, the number of multiplications and pairing products required in the proving and online verification procedures are reduced by $m(m-1)t - (m(m-1)/2)s$ and $(2t + 3s) - 4s = 2t - s$ respectively. We refer the reader to the full paper for the construction and security proof.

Bootstrapping to reduce CRS size. Similar to WW-BARG, by using the bootstrapping technique in [48], we can reduce the CRS size of our BARG to $m^c \cdot \mathsf{poly}(\lambda, s)$ for any $c > 0$. As a trade-off, the proof size will be dependent on $\log(m)$. A recent work by Kalai et al. [35] shows a general construction to convert BARGs into ones having both CRSs and proofs of size $\mathsf{poly}(\lambda, \log m, s)$. Instantiating the underlying BARG with ours immediately an efficient construction with both succinct CRSs and succinct proofs.

5 Experimental Performance

In this section, we experimentally evaluate the proving cost, verification cost, and the proof size of our NIZK and BARG for NP and compare them with GOS-NIZK and WW-BARG respectively. We focus on SXDH based implementations in asymmetric Type-3 pairings, since it is the most efficient one amongst all different types of pairings as mentioned in the introduction. The GOS-NIZK and WW-BARG are implemented by ourselves since the open sourced implementations are not available.

We implement NIZK and BARG schemes in C++ atop pairing-friendly curve `bls12-381` in the `mcl` library [47]. Parameters of all schemes are set to achieve 128-bit security level. All experiments are carried on a Macbook Pro with Intel i5-7360U CPU (2.30 GHz) and 16 GB, where a single exponentiation and pairing respectively take about 0.08 ms and 0.8 ms.

In Tables 3 and 4, we present experimental results regarding the proving and verification costs and the proof sizes of our NIZK and GOS-NIZK. The comparisons are carried out for both schemes under different ratios between the number of gates and wires, namely 2.00, 1.50, and 1.06. We also evaluated their performance across statement circuit sizes ranging from 2^8 to 2^{12}. Our prover is $1.52\times$, $1.32\times$, and $1.11\times$ faster than GOS-NIZK when the ratios are 2.00, 1.50, and 1.06 respectively. For the same ratios, our verifier is $1.44\times$, $1.21\times$, and $1.02\times$ faster. Additionally, our proof sizes are $1.62\times$, $1.38\times$, and $1.16\times$ smaller. One can see that our scheme outperforms GOS-NIZK in every aspect, and the significance of our improvement increases as the ratio becomes larger. Additionally, we note

Table 3. Comparison of the proving and verification cost (in seconds) between GOS-NIZK and our NIZK.

Scheme	**Ratio**	**Proving Cost (seconds)**					**Verification Cost (seconds)**				
		2^8	2^9	2^{10}	2^{11}	2^{12}	2^8	2^9	2^{10}	2^{11}	2^{12}
GOS12 [29]	2.00	1.38	2.69	5.39	10.81	21.72	12.55	25.80	50.57	101.11	201.95
Ours		0.87	1.82	3.51	6.99	14.37	8.68	17.38	37.23	70.04	138.70
GOS12 [29]	1.50	1.17	2.23	4.55	9.27	17.87	10.61	21.15	42.28	84.91	168.13
Ours		0.85	1.69	3.49	6.74	13.75	8.61	17.27	34.74	68.60	141.79
GOS12 [29]	1.06	0.91	1.83	3.65	7.32	14.65	8.61	17.25	34.49	69.01	138.28
Ours		0.83	1.65	3.30	6.64	13.25	8.58	17.12	34.81	68.53	137.28

Table 4. Comparison of the proof size (in MB) between GOS-NIZK and our NIZK.

Scheme	**Proof Size (MB) (Ratio: 2.00)**					**Proof Size (MB) (Ratio: 1.50)**					**Proof Size (MB) (Ratio: 1.06)**				
	2^8	2^9	2^{10}	2^{11}	2^{12}	2^8	2^9	2^{10}	2^{11}	2^{12}	2^8	2^9	2^{10}	2^{11}	2^{12}
GOS12 [29]	0.61	1.22	2.44	4.87	9.75	0.50	1.01	2.01	4.03	8.06	0.41	0.82	1.64	3.29	6.58
Ours	0.37	0.75	1.50	3.00	6.00	0.36	0.73	1.45	2.90	5.81	0.35	0.70	1.41	2.82	5.65

that the ratio tends to be 2 (i.e., its upper bound) when most gates do not share common input wires, and the ratio tends to be close to 1 (i.e., its lower bound) when most gates share common input wires, which may happen when most gates have multiple fan-out and the witness size is very small. Similar same argument can also be made for our BARG.

In Tables 5 and 6, we present experimental results regarding the proving and verification costs and the proof sizes of our BARG and WW-BARG when proving 50 and 100 statements. The comparisons are carried out for both schemes under different ratios between the number of gates and wires, namely 2.00, 1.50, and 1.06. We also evaluated their performance across statement circuit sizes ranging from 2^8 to 2^{12}. For proving 100 statement instances, our prover is 2.27×, 1.63×, and 1.35× faster than WW-BARG when the ratios are 2.00, 1.50, and 1.06 respectively. For the same ratios, the verifier is 2.70×, 2.35×, and 1.92× faster. For proving 50 statement instances with respect to these ratios, our prover is 2.13×, 1.51×, and 1.28× faster, and our verifier is 2.63×, 2.27×, and 1.94× faster. Additionally, our proof sizes are 1.20×, 1.11×, and 1.02× smaller, regardless of the number of statement instances. As a result, our scheme outperforms WW-BARG in every aspect.

Table 5. Comparison of the proving and verification costs (in seconds) between WW-BARG and our BARG. "stats." means statement instances.

Scheme	**Ratio**	**Proving Cost (seconds)**					**Verification Cost (seconds)**				
		2^8	2^9	2^{10}	2^{11}	2^{12}	2^8	2^9	2^{10}	2^{11}	2^{12}
WW22 [48] (100 stats.)	2.00	2.50	4.64	9.93	18.36	37.44	15.69	30.23	65.45	123.66	255.95
Ours (100 stats.)		1.07	2.02	4.10	8.00	16.91	5.90	11.61	23.38	46.41	94.46
WW22 [48] (50 stats.)	2.00	0.61	1.22	2.46	4.71	9.74	16.43	31.16	62.21	118.37	253.20
Ours (50 stats.)		0.29	0.55	1.20	2.05	4.67	5.68	11.44	23.40	46.56	95.28
WW22 [48] (100 stats.)	1.50	1.51	3.11	6.06	12.61	25.43	13.38	26.69	52.00	108.68	212.98
Ours (100 stats.)		1.02	1.87	4.09	7.56	16.57	5.95	11.20	22.94	44.92	92.28
WW22 [48] (50 stats.)	1.50	0.39	0.82	1.56	3.39	6.49	12.56	26.17	52.23	108.45	211.41
Ours (50 stats.)		0.26	0.57	1.03	2.23	4.30	6.25	11.90	23.27	46.47	93.22
WW22 [48] (100 stats.)	1.06	1.84	3.80	7.67	14.80	30.73	15.27	30.21	62.68	119.36	248.12
Ours (100 stats.)		1.00	1.99	3.82	8.53	15.95	5.81	12.19	23.11	46.63	96.11
WW22 [48] (50 stats.)	1.06	0.42	0.67	1.30	2.61	5.40	11.51	22.91	43.82	94.81	182.75
Ours (50 stats.)		0.25	0.50	1.13	2.08	4.13	6.16	11.86	23.97	47.32	93.92

Table 6. Comparison of the proof size (in MB) between WW-BARG and our BARG. "stats." means statement instances.

Scheme	**Proof Size (MB) (Ratio: 2.00)**					**Proof Size (MB) (Ratio: 1.50)**					**Proof Size (MB) (Ratio: 1.06)**				
	2^8	2^9	2^{10}	2^{11}	2^{12}	2^8	2^9	2^{10}	2^{11}	2^{12}	2^8	2^9	2^{10}	2^{11}	2^{12}
WW22 [48] (100 stats.)	0.42	0.84	1.69	3.37	6.75	0.35	0.70	1.41	2.81	5.62	0.29	0.58	1.16	2.32	4.64
Ours (100 stats.)	0.35	0.70	1.41	2.81	5.62	0.32	0.63	1.26	2.53	5.06	0.28	0.57	1.14	2.28	4.57
WW22 [48] (50 stats.)	0.42	0.84	1.69	3.37	6.75	0.35	0.70	1.41	2.81	5.62	0.29	0.58	1.16	2.32	4.64
Ours (50 stats.)	0.35	0.70	1.41	2.81	5.62	0.32	0.63	1.26	2.53	5.06	0.28	0.57	1.14	2.28	4.57

Acknowledgement. Parts of Yuyu Wang's work was 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 National Key Research and Development Program of China under Grant Number 2022YFB3104600. Parts of Jiaxin Pan's work was supported by the Research Council of Norway under Project No. 324235. Parts of Yu Chen's work was supported by the National Key Research and Development Program of China under Grant Number 2021YFA1000600, the National Natural Science Foundation of China under Grant Numbers 62272269 and 61932019, Taishan Scholar Program of Shandong Province.

Appendix

A GOS-NIZK in the Asymmetric Pairing Setting

The original GOS-NIZK [29] was proposed in the symmetric pairing setting. In this section, we give the GOS-NIZK in the asymmetric pairing setting. It was previously indicated by [22,45] but has never been treated explicitly.

Let λ be the security parameter and $\mathsf{par} = \mathcal{G}$ be the public parameter, where $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, [1]_1, [1]_2, e) \overset{\$}{\leftarrow} \mathsf{GGen}(1^\lambda)$, and $\mathsf{ORNIZK} = (\mathsf{NGen}_{\mathsf{or}}, \mathsf{NProve}_{\mathsf{or}}, \mathsf{NVer}_{\mathsf{or}})$ be a NIZK with the simulator $(\mathsf{NTGen}_{\mathsf{or}}, \mathsf{NSim}_{\mathsf{or}})$. Let $\mathsf{L}^{\mathsf{or}}_{[\mathbf{M}]_1}$ be the following language it supports.

$$\mathsf{L}^{\mathsf{or}}_{[\mathbf{M}]_1} = \{(\mathsf{C}_{[\mathbf{M}]_1}, ([\mathbf{x}_0]_1, [\mathbf{x}_1]_1)) | \exists \mathbf{w} \in \mathbb{Z}_p^{2k} : \mathsf{C}_{[\mathbf{M}]_1}(([\mathbf{x}_0]_1, [\mathbf{x}_1]_1), \mathbf{w}) = 1\},$$

where $\mathsf{C}_{[\mathbf{M}]_1} : \mathbb{G}_1^{k+1} \times \mathbb{G}_1^{k+1} \times \mathbb{Z}_p^k \to \{0,1\}$ is a Boolean circuit on input $((\mathbf{x}_0, \mathbf{x}_1), \mathbf{w})$ outputting 1 iff $[\mathbf{x}_0]_1 = [\mathbf{M}]_1\mathbf{w} \vee [\mathbf{x}_1]_1 = [\mathbf{M}]_1\mathbf{w}$ for $\mathbf{M} \in \mathcal{D}_k$. We give the NIZK NIZK^* in Fig. 11.

Theorem 7 (Completeness). *If* ORNIZK *is complete, then* NIZK^* *is complete.*

Proof. Let w_{d_1} and w_{d_2} be the input bits of a NAND gate, and w_{d_3} be the true output. We must have $\mathsf{w}_{d_1} + \mathsf{w}_{d_2} + 2\mathsf{w}_{d_3} - 2 \in \{0,1\}$. Let $\mathsf{cm}_{d_1} = [\mathbf{M}\mathbf{r}_{d_1} + \mathbf{z}\mathsf{w}_{d_1}]_1$ and $\mathsf{cm}_{d_2} = [\mathbf{M}\mathbf{r}_{d_2} + \mathbf{z}\mathsf{w}_{d_2}]_1$ be the input commitments and $\mathsf{cm}_{d_3} = [\mathbf{M}\mathbf{r}_{d_3} + \mathbf{z}\mathsf{w}_{d_3}]_1$ be the output commitment. We have

$$\begin{aligned} \mathsf{x}_\ell &= \mathsf{cm}_{d_1} + \mathsf{cm}_{d_2} + 2\mathsf{cm}_{d_3} - [\mathbf{z} \cdot 2]_1 \\ &= [\mathbf{M}]_1(\mathbf{r}_{d_1} + \mathbf{r}_{d_2} + 2\mathbf{r}_{d_3}) + [\mathbf{z}]_1(\mathsf{w}_{d_1} + \mathsf{w}_{d_2} + 2\mathsf{w}_{d_3} - 2). \end{aligned}$$

Therefore, for all $\ell \in [s]$, we must have $\mathsf{x}_\ell \in \mathsf{Span}([\mathbf{M}]_1)$ or $\mathsf{x}_\ell - [\mathbf{z}]_1 \in \mathsf{Span}([\mathbf{M}]_1)$. Moreover, for all $i \in [t]$, we have $\mathsf{cm}_i \in \mathsf{Span}([\mathbf{M}]_1)$ or $\mathsf{cm}_i - [\mathbf{z}]_1 \in \mathsf{Span}([\mathbf{M}]_1)$. Then the completeness of NIZK follows from the completeness of ORNIZK, completing the proof of Theorem 7. □

Theorem 8 (Composable zero-knowledge). *Under the* $\mathcal{D}_k$-MDDH *assumption, if* ORNIZK *is a NIZK with composable zero-knowledge, then* NIZK *is a NIZK with composable zero-knowledge.*

```
NGen*(1^λ, par):
$\mathbf{M} \xleftarrow{\$} \mathcal{D}_k$, $\mathsf{crs}_{\mathsf{or}} \xleftarrow{\$} \mathsf{NTGen}_{\mathsf{or}}(\mathsf{par}, \mathsf{C}_{[\mathbf{M}]_1})$, $\mathbf{z} \xleftarrow{\$} \mathbb{Z}_p^{k+1} \backslash \mathsf{Span}(\mathbf{M})$
Return $\mathsf{CRS} = (\mathsf{crs}_{\mathsf{or}}, [\mathbf{M}]_1, [\mathbf{z}]_1)$

NProve*(CRS, C, x, w):
Hardwire $\mathsf{x}$ in $\mathsf{C}$ to obtain the circuit $\mathsf{C}(\mathsf{x}, \cdot) : \{0,1\}^h \to \{0,1\}$
Define $s$ and $t$ to be the numbers of gates and wires of $\mathsf{C}(\mathsf{x}, \cdot)$ respectively
Extend $\mathsf{w}$ to $(\mathsf{w}_i)_{i \in [t]}$ containing the bits of all wires in $\mathsf{C}(\mathsf{x}, \cdot)$
Compute $\mathbf{r}_i \xleftarrow{\$} \mathbb{Z}_p^{k+1}$ and $\mathsf{cm}_i = [\mathbf{M}]_1 \mathbf{r}_i + [\mathbf{z}]_1 \mathsf{w}_i$ for all $i \in [t-1]$
Set $\mathbf{r}_t = \mathbf{0}$ and $\mathsf{cm}_t = [\mathbf{z}]_1$ for the output wire
Compute $\hat{\pi}_i \xleftarrow{\$} \mathsf{NProve}_{\mathsf{or}}(\mathsf{crs}_{\mathsf{or}}, \mathsf{C}_{[\mathbf{M}]_1}, (\mathsf{cm}_i, \mathsf{cm}_i - [\mathbf{z}]_1), \mathbf{r}_i)$ for all $i \in [t]$
For each NAND gate $G_\ell = (d_1, d_2, d_3) \in [t]^3$ where $\ell \in [s]$, run
  – $\mathsf{x}_\ell = \mathsf{cm}_{d_1} + \mathsf{cm}_{d_2} + 2\mathsf{cm}_{d_3} - [\mathbf{z} \cdot 2]_1$, $\mathsf{r}_\ell = \mathbf{r}_{d_1} + \mathbf{r}_{d_2} + 2\mathbf{r}_{d_3}$
  – $\pi_\ell \xleftarrow{\$} \mathsf{NProve}_{\mathsf{or}}(\mathsf{crs}_{\mathsf{or}}, \mathsf{C}_{[\mathbf{M}]_1}, (\mathsf{x}_\ell, \mathsf{x}_\ell - [\mathbf{z}]_1), \mathsf{r}_\ell)$
Return $\Pi = ((\mathsf{cm}_i, \hat{\pi}_i)_{i \in [t]}, (\pi_\ell)_{\ell \in [s]})$

NVer*(CRS, C, x, Π):
Hardwire $\mathsf{x}$ in $\mathsf{C}$ to obtain $\mathsf{C}(\mathsf{x}, \cdot)$ in the same way as NProve* does
Check that all wires in $\mathsf{C}(\mathsf{x}, \cdot)$ have a corresponding commitment and $\mathsf{cm}_t = [\mathbf{z}]_1$
Check that all NAND gates have a valid NIZK proof of compliance
Return 1 iff all checks pass
```

Fig. 11. Definition of $\mathsf{NIZK}^* = (\mathsf{NGen}^*, \mathsf{NProve}^*, \mathsf{NVer}^*)$.

Proof. We define the simulator $(\mathsf{NTGen}^*, \mathsf{NSim}^*)$ as in Fig. 12.

First we note that the distribution of $\mathbf{z} \xleftarrow{\$} \mathbb{Z}_p^{k+1} \backslash \mathsf{Span}(\mathbf{M})$ is $1/p$-statistically close to the uniform distribution over $\mathbb{Z}_p^{k+1}$. Then the indistinguishability of CRSs generated by NGen^* and NTGen^* follows immediately from the $\mathcal{D}_k$-MDDH assumption and the composable zero-knowledge of ORNIZK (which says that $\mathsf{crs}_{\mathsf{or}}$ generated by $\mathsf{NGen}_{\mathsf{or}}(1^\lambda, \mathsf{par})$ and $\mathsf{NTGen}_{\mathsf{or}}(1^\lambda, \mathsf{par})$ are computationally close).

Next we define a modified prover $\mathsf{NProve}^{*\prime}$, which is exactly the same as NProve^* except that $\hat{\pi}_i$ is generated as

$$\hat{\pi}_i \xleftarrow{\$} \mathsf{NSim}_{\mathsf{or}}(\mathsf{crs}_{\mathsf{or}}, \mathsf{td}_{\mathsf{or}}, \mathsf{C}_{[\mathbf{M}]_1}, (\mathsf{cm}_i, \mathsf{cm}_i - [\mathbf{z}]_1))$$

for $i \in [t]$, and for each NAND gate π_ℓ is generated as

$$\pi_\ell \xleftarrow{\$} \mathsf{NSim}_{\mathsf{or}}(\mathsf{crs}_{\mathsf{or}}, \mathsf{td}_{\mathsf{or}}, \mathsf{C}_{[\mathbf{M}]_1}, (\mathsf{x}_\ell, \mathsf{x}_\ell - [\mathbf{z}]_1)).$$

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

$$\Pi \xleftarrow{\$} \mathsf{NProve}^*(\mathsf{CRS}, \mathsf{C}, \mathsf{x}, \mathsf{w}) \text{ and } \Pi \xleftarrow{\$} \mathsf{NProve}^{*\prime}(\mathsf{CRS}, \mathsf{C}, \mathsf{x}, \mathsf{w})$$

for $(\mathsf{CRS}, \mathsf{TD}) \xleftarrow{\$} \mathsf{NTGen}^*(1^\lambda, \mathsf{par})$ and any (x, w) such that $\mathsf{C}(\mathsf{x}, \mathsf{w}) = 1$.

Moreover, since the distribution of $\mathsf{cm}_i = [\mathbf{M}\mathbf{r}_i]_1$ is identical to that of $\mathsf{cm}_i = [\mathbf{M}\mathbf{r}_i + \mathbf{z}\mathsf{w}_i]_1$ for $\mathbf{r}_i \xleftarrow{\$} \mathbb{Z}_p^\lambda$ when $\mathbf{z} \in \mathsf{Span}(\mathbf{M})$, the distributions of

$$\Pi \xleftarrow{\$} \mathsf{NProve}^{*\prime}(\mathsf{CRS}, \mathsf{C}, \mathsf{x}, \mathsf{w}) \text{ and } \Pi \xleftarrow{\$} \mathsf{NSim}^*(\mathsf{CRS}, \mathsf{TD}, \mathsf{C}, \mathsf{x}),$$

```
$\underline{\mathsf{NTGen}^*(1^\lambda, \mathsf{par})}$:
$(\mathsf{crs}_{\mathsf{or}}, \mathsf{td}_{\mathsf{or}}) \xleftarrow{\$} \mathsf{NTGen}_{\mathsf{or}}(\mathsf{par})$, $\mathbf{M} \xleftarrow{\$} \mathcal{D}_k$, $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^k$, $\mathbf{z} = \mathbf{M} \cdot \mathbf{u}$
Return $\mathsf{CRS} = (\mathsf{crs}_{\mathsf{or}}, \mathbf{M}, [\mathbf{z}]_1)$ and $\mathsf{TD} = \mathsf{td}_{\mathsf{or}}$

$\underline{\mathsf{NSim}^*(\mathsf{CRS}, \mathsf{TD}, \mathsf{C}, \mathsf{x})}$:
Hardwire $\mathsf{x}$ in $\mathsf{C}$ to obtain $\mathsf{C}(\mathsf{x}, \cdot)$ in the same way as $\mathsf{NProve}^*$ does
Define $s$ and $t$ to be the numbers of gates and wires of $\mathsf{C}(\mathsf{x}, \cdot)$ respectively
Compute $\mathbf{r}_i \xleftarrow{\$} \mathbb{Z}_p^\lambda$ and $\mathsf{cm}_i = [\mathbf{M}\mathbf{r}_i]_1$ for all $i \in [t-1]$
Set $\mathsf{cm}_t = [\mathbf{z}]_1$ for the output wire
Compute $\hat{\pi}_i \xleftarrow{\$} \mathsf{NSim}_{\mathsf{or}}(\mathsf{crs}_{\mathsf{or}}, \mathsf{td}_{\mathsf{or}}, [\mathbf{M}]_1, (\mathsf{cm}_i, \mathsf{cm}_i - [\mathbf{z}]_1))$ for $i \in [t]$
For each NAND gate $G_\ell = (d_1, d_2, d_3) \in [t]^3$ where $\ell \in [s]$, run
  – $\mathsf{x}_\ell = \mathsf{cm}_{d_1} + \mathsf{cm}_{d_2} + 2\mathsf{cm}_{d_3} - [\mathbf{z} \cdot 2]_1$
  – $\pi_\ell \xleftarrow{\$} \mathsf{NSim}_{\mathsf{or}}(\mathsf{crs}_{\mathsf{or}}, \mathsf{td}_{\mathsf{or}}, [\mathbf{M}]_1, (\mathsf{x}_\ell, \mathsf{x}_\ell - [\mathbf{z}]_1))$
Return $\varPi = ((\mathsf{cm}_i, \hat{\pi}_i)_{i \in [t]}, (\pi_\ell)_{\ell \in [s]})$
```

Fig. 12. Definition of the simulator $(\mathsf{NTGen}^*, \mathsf{NSim}^*)$.

where $(\mathsf{CRS}, \mathsf{TD}) \xleftarrow{\$} \mathsf{NTGen}^*(1^\lambda, \mathsf{par})$ and $\mathsf{C}(\mathsf{x}, \mathsf{w}) = 1$, are identical as well, completing the proof of Theorem 8. □

Theorem 9 (Soundness). *If* ORNIZK *is perfectly sound, then* NIZK* *is perfectly sound.*

Proof. Due to the perfect soundness of ORNIZK, for each NAND gate with input commitments $(\mathsf{cm}_{d_1}, \mathsf{cm}_{d_2})$ and an output commitment cm_{d_3} in a valid proof, we have $\mathsf{cm}_d \in \mathsf{Span}([\mathbf{M}]_1)$ or $\mathsf{cm}_d - [\mathbf{z}]_1 \in \mathsf{Span}([\mathbf{M}]_1)$ for $d \in \{d_1, d_2, d_3\}$, and

$$\mathsf{x}_\ell = (\mathsf{cm}_{d_1} + \mathsf{cm}_{d_2} + \mathsf{cm}_{d_3} - [\mathbf{z} \cdot 2]_1) \in \mathsf{Span}([\mathbf{M}]_1)$$

$$\text{or} \quad \mathsf{x}_\ell = (\mathsf{cm}_{d_1} + \mathsf{cm}_{d_2} + \mathsf{cm}_{d_3} - [\mathbf{z} \cdot 2]_1) - [\mathbf{z}]_1 \in \mathsf{Span}([\mathbf{M}]_1).$$

Let $\mathbf{k}$ be the vector in the kernel of $\mathbf{M}$ such that $\mathbf{k}^\top \mathbf{z} = 1$, which must exist when $\mathbf{z} \notin \mathsf{Span}(\mathbf{M})$. We have $\mathbf{k}^\top \mathsf{cm}_{d_1}, \mathbf{k}^\top \mathsf{cm}_{d_2}, \mathbf{k}^\top \mathsf{cm}_{d_3} \in \{[0]_1, [1]_1\}$ and

$$\mathbf{k}^\top \mathsf{cm}_{d_1} + \mathbf{k}^\top \mathsf{cm}_{d_2} + \mathbf{k}^\top \mathsf{cm}_{d_3} - [2]_1 \in \{[0]_1, [1]_1\}.$$

As a result, we can extract the bits for all wires consisting of valid input/output pairs for all NAND gates and leading the statement circuit to output 1, completing the proof of Theorem 9. □

Instantiation of the OR-proof system. The underlying OR-proof system can be instantiated as in [22,45] (see Sect. 3.1 for the instantiation). Under the SXDH assumption, each CRS consists of 4 elements in $\mathbb{G}_2$ and each proof consists of 4 and 6 elements in $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. In this case, the proof size of the resulting NIZK consists of $6t+4s$ elements in $\mathbb{G}_1$ and $6t+6s$ elements in $\mathbb{G}_2$, where t and s are the number of wires and gates in the statement circuit respectively.

References

1. Abe, M., Hoshino, F., Ohkubo, M.: Design in Type-I, Run in Type-III: fast and scalable bilinear-type conversion using integer programming. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9816, pp. 387–415. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53015-3_14
2. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Scalable, transparent, and post-quantum secure computational integrity. IACR Cryptol. ePrint Arch, p. 46 (2018)
3. Bitansky, N., et al.: The hunting of the SNARK. J. Cryptol. **30**(4), 989–1066 (2017)
4. Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.: From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In: Goldwasser, S. (ed.) ITCS 2012, pp. 326–349. ACM (2012)
5. Bitansky, N., Canetti, R., Paneth, O., Rosen, A.: On the existence of extractable one-way functions. In: Shmoys, D.B. (ed.) 46th ACM STOC, pp. 505–514. ACM Press (2014)
6. Bitansky, N., Chiesa, A., Ishai, Y., Paneth, O., Ostrovsky, R.: Succinct non-interactive arguments via linear interactive proofs. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 315–333. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36594-2_18
7. Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications (extended abstract). In: STOC, pp. 103–112. ACM (1988)
8. Boneh, D., Ishai, Y., Sahai, A., Wu, D.J.: Lattice-based SNARGs and their application to more efficient obfuscation. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017. LNCS, vol. 10212, pp. 247–277. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56617-7_9
9. Chase, M., et al.: Post-quantum zero-knowledge and signatures from symmetric-key primitives. In: Thuraisingham, B.M., Evans, D., Malkin, T., Xu, D. (eds.) ACM CCS 2017, pp. 1825–1842. ACM Press (2017)
10. Chatterjee, S., Menezes, A.: Type 2 structure-preserving signature schemes revisited. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9452, pp. 286–310. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48797-6_13
11. Chiesa, A., Hu, Y., Maller, M., Mishra, P., Vesely, N., Ward, N.: Marlin: preprocessing zkSNARKs with universal and updatable SRS. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12105, pp. 738–768. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45721-1_26
12. Chiesa, A., Ojha, D., Spooner, N.: Fractal: post-quantum and transparent recursive proofs from holography. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part I. LNCS, vol. 12105, pp. 769–793. Springer, Heidelberg (May (2020). https://doi.org/10.1007/978-3-030-45721-1_27
13. Choudhuri, A.R., Jain, A., Jin, Z.: Non-interactive batch arguments for NP from standard assumptions. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12828, pp. 394–423. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84259-8_14
14. Choudhuri, A.R., Jain, A., Jin, Z.: Snargs for $\mathcal{P}$ from LWE. In: FOCS, pp. 68–79. IEEE (2021)
15. Couteau, G., Hartmann, D.: Shorter non-interactive zero-knowledge arguments and ZAPs for algebraic languages. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12172, pp. 768–798. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56877-1_27

16. Damgård, I., Faust, S., Hazay, C.: Secure two-party computation with low communication. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 54–74. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28914-9_4
17. 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. LNCS, vol. 8873, pp. 532–550. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45611-8_28
18. Escala, A., Herold, G., Kiltz, E., Ràfols, C., Villar, J.: An algebraic framework for Diffie-Hellman assumptions. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8043, pp. 129–147. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40084-1_8
19. Feige, U., Lapidot, D., Shamir, A.: Multiple noninteractive zero knowledge proofs under general assumptions. SIAM J. Comput. **29**(1), 1–28 (1999)
20. Galbraith, S., Paterson, K., Smart, N.: Pairings for cryptographers. Cryptology ePrint Archive, Report 2006/165 (2006). https://eprint.iacr.org/2006/165
21. Garg, S., Jain, A., Sahai, A.: Leakage-resilient zero knowledge. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 297–315. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_17
22. Gay, R., Hofheinz, D., Kohl, L., Pan, J.: More efficient (almost) tightly secure structure-preserving signatures. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10821, pp. 230–258. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78375-8_8
23. 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 (2013). https://doi.org/10.1007/978-3-642-38348-9_37
24. Giacomelli, I., Madsen, J., Orlandi, C.: ZKBoo: faster zero-knowledge for Boolean circuits. In: Holz, T., Savage, S. (eds.) USENIX Security 2016, pp. 1069–1083. USENIX Association (2016)
25. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM J. Comput. **18**(1), 186–208 (1989)
26. Groth, J.: Short non-interactive zero-knowledge proofs. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 341–358. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17373-8_20
27. Groth, J.: Short pairing-based non-interactive zero-knowledge arguments. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 321–340. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17373-8_19
28. Groth, J.: On the size of pairing-based non-interactive arguments. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9666, pp. 305–326. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_11
29. Groth, J., Ostrovsky, R., Sahai, A.: New techniques for noninteractive zero-knowledge. J. ACM **59**(3), 11:1–11:35 (2012)
30. Groth, J., Sahai, A.: Efficient noninteractive proof systems for bilinear groups. SIAM J. Comput. **41**(5), 1193–1232 (2012)
31. Joux, A.: Faster index calculus for the medium prime case application to 1175-bit and 1425-bit finite fields. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 177–193. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38348-9_11
32. Joux, A.: A new index calculus algorithm with complexity $L(1/4 + o(1))$ in very small characteristic. Cryptology ePrint Archive, Report 2013/095 (2013). https://eprint.iacr.org/2013/095

33. Jutla, C.S., Roy, A.: Switching lemma for bilinear tests and constant-size NIZK proofs for linear subspaces. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8617, pp. 295–312. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44381-1_17
34. Jutla, C.S., Roy, A.: Shorter quasi-adaptive NIZK proofs for linear subspaces. J. Cryptol. **30**(4), 1116–1156 (2017)
35. Kalai, Y., Lombardi, A., Vaikuntanathan, V., Wichs, D.: Boosting batch arguments and RAM delegation. In: STOC, pp. 1545–1552. ACM (2023)
36. Kalai, Y.T., Paneth, O., Yang, L.: How to delegate computations publicly. In: Charikar, M., Cohen, E. (eds.) 51st ACM STOC, pp. 1115–1124. ACM Press (2019)
37. Katsumata, S., Nishimaki, R., Yamada, S., Yamakawa, T.: Compact NIZKs from standard assumptions on bilinear maps. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12107, pp. 379–409. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45727-3_13
38. Katsumata, S., Nishimaki, R., Yamada, S., Yamakawa, T.: Compact NIZKs from standard assumptions on bilinear maps. Cryptology ePrint Archive, Report 2020/223 (2020). https://eprint.iacr.org/2020/223
39. Kiltz, E., Wee, H.: Quasi-adaptive NIZK for linear subspaces revisited. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 101–128. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46803-6_4
40. 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 (2012). https://doi.org/10.1007/978-3-642-28914-9_10
41. Lipmaa, H.: Succinct non-interactive zero knowledge arguments from span programs and linear error-correcting codes. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013. LNCS, vol. 8269, pp. 41–60. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-42033-7_3
42. Micali, S.: Computationally sound proofs. SIAM J. Comput. **30**(4), 1253–1298 (2000)
43. Morillo, P., Ràfols, C., Villar, J.L.: The kernel matrix Diffie-Hellman assumption. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016. LNCS, vol. 10031, pp. 729–758. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53887-6_27
44. Parno, B., Howell, J., Gentry, C., Raykova, M.: Pinocchio: nearly practical verifiable computation. In: 2013 IEEE Symposium on Security and Privacy, pp. 238–252. IEEE Computer Society Press (2013)
45. Ràfols, C.: Stretching Groth-Sahai: NIZK proofs of partial satisfiability. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015. LNCS, vol. 9015, pp. 247–276. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46497-7_10
46. Setty, S.: Spartan: efficient and general-purpose zkSNARKs without trusted setup. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12172, pp. 704–737. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56877-1_25
47. Shigeo, M.: A portable and fast pairing-based cryptography library. https://github.com/herumi/mcl
48. Waters, B., Wu, D.J.: Batch arguments for sfNP and more from standard bilinear group assumptions. In: CRYPTO 2022, Part II, pp. 433–463. LNCS, Springer, Heidelberg (2022)