Pairing-Based BARG for NP with Constant-Size Proofs and Applications

Zhe Jiang[1], Kai Zhang[2], Junqing Gong[1,3], and Haifeng Qian[1]

[1] East China Normal University, Shanghai, China.
52275902014@stu.ecnu.edu.cn, jqgong@sei.ecnu.edu.cn, hfqian@cs.ecnu.edu.cn
[2] Shanghai University of Electric Power, Shanghai, China.
kzhang@shiep.edu.cn
[3] Shanghai Qi Zhi Institute, Shanghai, China.

Abstract. This paper presents a pairing-based non-interactive batch argument (BARG) for NP in the common reference string (CRS) model with *constant-size* proofs and *constant-cost* verification. Our construction is fully black-box in its use of the underlying group and achieves a weak form of somewhere extractability under a new q-type assumption in composite-order pairing groups. We further show that this extractability guarantee implies somewhere soundness. Prior pairing-based BARGs in this setting suffer from proof size and verification cost proportional to the size of the Boolean circuit computing the NP relation, and our work removes this dependence.

Technically, our starting point is the Waters–Wu framework [Asiacrypt'25] and the main idea is to replace the univariate polynomial commitment with a *bivariate* polynomial commitment together with univariate sumcheck arguments. This allows us to compress both wire and gate checks into a constant number of algebraic identities, leading to constant-size proofs and constant-cost verification.

As a demonstration of the resulting somewhere-sound BARG, we show that it simplifies and improves existing generic constructions of NIZKs and rate-1 BARGs. In particular, we obtain:

- a generic construction of NIZKs for NP from our somewhere sound BARG, together with a one-time dual-mode commitment scheme with extraction and a leakage-resilient weak pseudorandom function. The resulting proof size is independent of the circuit size, whereas prior work incurs circuit-size-dependent proof overhead and either requires a local PRG or commits to all internal wires;
- a generic construction of rate-1 BARGs from pairing-based assumptions by combining the somewhere sound BARG obtained from our construction with existing rate-1 fully local somewhere-extractable hashing. The resulting proof size is $h + \mathsf{poly}(\lambda, \log \ell)$, whereas prior work either requires proof size $h + o(h)\mathsf{poly}(\lambda, \log \ell)$ or relies on a RAM SNARG with partial-input soundness.

Both generic constructions rely on weaker or fewer cryptographic primitives than prior work and the construction of NIZK also avoids the complicated parameter selection in prior work.

Keywords: Polynomial commitment; Univariate sumcheck arguments; Batch arguments; Composite-order bilinear group

1 Introduction

Non-interactive batch arguments (BARGs) for an NP relation $\mathcal{R}$ allow a prover to generate a single, succinct batch proof π certifying the validity of ℓ instances $(\mathbf{x}_1, \ldots, \mathbf{x}_\ell)$ with respect to their corresponding witnesses $(\mathbf{w}_1, \ldots, \mathbf{w}_\ell)$ [24,6,18,16]. The proof π is non-interactive and publicly verifiable with proof size bounded by $\mathsf{poly}(\lambda, |C|, \log \ell)$, where λ is the security parameter and

$|C|$ is the size of the Boolean circuit computing the NP-relation $\mathcal{R}$. Soundness requires that no accepting proof can be generated when a batch instance is false, namely, when

$$\exists i \in [\ell] \quad \text{such that} \quad \forall \mathbf{w} \in \{0,1\}^h,\ C(\mathbf{x}_i, \mathbf{w}) = 0.$$

Beyond amortizing NP verification, BARGs have become a central tool in cryptography, with applications to RAM delegation [23], incrementally verifiable computation [31,15,14], monotone-policy aggregate signatures [9,29,30], multi-key homomorphic signatures [21,1], and non-interactive zero-knowledge proofs [4,10,7]. More recently, BARGs have also emerged as a generic route to succinct non-interactive arguments for broader classes of computation [25,13,12,5,22]. In particular, general BARG-to-SNARG compilers [25,13] show that any BARG yields SNARGs for classes such as P and NTISP, and pairing-based instantiations lead to concrete SNARGs under standard assumptions [24,23,32,11].

In this work, we focus on pairing-based BARGs with extreme succinctness in the CRS model. We want to achieve constant-size proofs and constant pairing-based verification. Existing pairing-based constructions still have proof size and verifier pairing complexity that grow with the circuit size $|C|$ [32,11]. See Fig 1 for more details.

Scheme	Proof Size	Prover Time	Pairing	Linear CRS
[32]	$O(\|C\|)$	$O(\ell^2 \cdot \|C\|)$	$O(\|C\|)$	$O(\ell^2)$
[11]	$O(\|C\|)$	$O(\ell \cdot \|C\|)$	$O(\|C\|)$	$O(\ell)$
[3]	$O(\|\mathbf{w}\|)$	$O(\ell^5 \cdot \|C\|^5)$	$O(\|\mathbf{w}\|)$	$O(\ell^5 \cdot \|C\|^5)$
Ours	$O(1)$	$O(\ell \cdot \|C\|)$	$O(1)$	$O(\ell \cdot \|C\|^2)$

Fig. 1: Comparison of batch argument systems for NP. For readability, we suppress a common linear factor in security parameters λ from the **Proof Size**, **Prover Time**, and **Linear CRS** columns. In the table, $|C|$ denotes the size of the circuit C computing the NP relation, ℓ is the number of instances, $|\mathbf{w}|$ denotes the size of a single witness $\mathbf{w}$, the **Pairing** column reports the number of pairing computations.

Results. In this work, we put forward an efficient pairing-based batch argument for NP within the framework of [32], which achieves constant proof size and requires only a constant number of pairing operations. Moreover, our construction is fully black-box in its use of the underlying group and satisfies weak somewhere extractability under a q-type assumption over composite-order pairing groups. By the derived implication from weak somewhere extractability to somewhere soundness, it also satisfies somewhere soundness. Formally,

Theorem 1 (Informal). *Let $\ell, t, s \in \mathbb{N}$. Assuming the hardness of (ℓ, t, s)-pairwise bivariate subgroup decision exponent assumption 1, there exists a weak somewhere extractability BARG for NP with constant proof size and pairing verification. Moreover, by Lemma 1, the construction also satisfies somewhere soundness.*

Moreover, we present two applications of our BARG with improved communication cost.

Security guarantee. Our base construction provides weak somewhere extractability rather than standard adaptive BARG soundness. The extraction target consists of a hidden instance, a hidden wire, and a hidden gate. The extracted global-wire and gate-local values are required to satisfy public-input consistency, gate validity, gate–wire consistency, and output-wire consistency. We prove that these local extraction guarantees imply somewhere soundness. This weaker hidden-index guarantee is sufficient for both applications below: the NIZK application uses somewhere soundness to rule out false local instances, while the rate-1 BARG application uses somewhere soundness to certify the local-checking circuit and uses the rate-1 somewhere-extractable hash to extract the complete witness.

Application 1: NIZK from BARG. Combining our BARG with a one-time dual-mode commitment scheme and a leakage-resilient weak pseudorandom function (LRwPRF), we obtain a construction of NIZKs for NP.

Theorem 2 (Informal). *Assuming a somewhere sound BARG for NP with constant proof size, a leakage-resilient weak pseudorandom function and a one-time dual-mode commitment scheme with extraction, there exists a NIZK for NP with proof size $O(|\mathsf{rk}|) + O(1)$. Here, $O(|\mathsf{rk}|)$ comes from the commitments to the LRwPRF key bits and $O(1)$ comes from the BARG proof.*

In contrast to prior hidden-bits generator based constructions, our approach requires neither the small locality assumption and complicated parameter selection of [10] nor the additional commitments to internal wire values used in [4]. As a result, the proof size depends only on the LRwPRF key length and a constant-size BARG proof, and is independent of the size of the underlying circuit.

Application 2: Rate-1 BARG. Combining the somewhere sound BARG from our construction with existing non-black-box transformations [15,8], we obtain a rate-1 BARG from pairing-based assumptions.

Theorem 3 (Informal). *Assuming a somewhere sound BARG with constant-size proofs and a rate-1 fully local somewhere extractable hash function, there exists a construction of a BARG for NP with the proof size $h + \mathsf{poly}(\lambda, \log \ell)$. Here, h is the size of a single witness and ℓ is the number of instances.*

The prior construction of BARG for NP [31] has proof size $h+o(h)\mathrm{poly}(\lambda, \log \ell)$. The constructions of BARG [15,8] have proof size $h+\mathrm{poly}(\lambda, \log \ell)$. However, their compiler relies on a flexible RAM SNARG with partial-input soundness and an index seBARG. In contrast, our construction avoids the RAM SNARG layer: the rate-1 flSEH extracts the full witness at the hidden instance, while the somewhere-sound BARG directly certifies the local consistency of the circuit computation.

Concurrent Work. A concurrent work [3] achieves compact proof size and verification time independent of the circuit size. Their construction builds a BARG generically from a projective chainable functional commitment, with proof size and online verifier runtime scaling as $O(\lambda|\mathbf{w}|)$. Our scheme is based on a bivariate polynomial commitment combined with a univariate sumcheck protocol. This allows us to compress the wire and gate consistency checks into a constant number of algebraic identities and achieve constant proof size and constant pairing-based verification time.

Another concurrent and related work is Soloist [28], which uses bivariate polynomial commitments to construct distributed SNARKs with constant-size proofs. Both Soloist and our work exploit bivariate encodings to aggregate two-dimensional algebraic data and transform large families of constraint checks into polynomial identity tests. However, the proof tasks and security mechanisms are different. Soloist focuses on distributed SNARKs for R1CS, and knowledge soundness is established in the algebraic group model. Our construction instead gives a BARG for NP with weak somewhere extractability. The extraction mechanism uses hidden-index programming in composite-order pairing groups under a q-type assumption. These local extraction guarantees further imply somewhere soundness.

2 Technical Overview

Let $\ell \in \mathbb{N}$ be the number of instances. Let $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ be a Boolean circuit with t wires and s gates that verifies the NP relation. Given statements $\mathbf{x}_1, \ldots, \mathbf{x}_\ell \in \{0,1\}^n$ and $\mathbf{w}_1, \ldots, \mathbf{w}_\ell \in \{0,1\}^h$, BARG for NP works in the crs model and allows us to prove that $C(\mathbf{x}_i, \mathbf{w}_i) = 1$ for all $i \in [\ell]$ with a proof of size sublinear in ℓ. We will construct a BARG with constant-size proof and weaker extractability.By Lemma 1, this property also implies somewhere soundness for our construction. Two applications will be explained to demonstrate that our BARG helps reduce communication cost of various cryptographic primitives and yields simple constructions.

2.1 Our BARG with Constant-Size Proof

Notations. We work with bilinear groups and employ implicit representations. For bilinear group $(\mathbb{G} = \langle g \rangle, \mathbb{T} = \langle e(g,g) \rangle, e : \mathbb{G} \times \mathbb{G} \to \mathbb{T})$ of prime order p and $x \in \mathbb{Z}_p$, we write $[x] = g^x$

and $[x]_T = e(g,g)^x$. For all $x, y \in \mathbb{Z}_p$, we write

$$[x] + [y] = [x+y], \qquad x \cdot [y] = [xy], \qquad [x] \bullet [y] = e([x],[y]),$$

and the first two also apply to $[\cdot]_T$.

Starting Point. Our starting point is pairing-based BARG [11] based on [32]. Number wires with $1, \ldots, t$ and gates with $1, \ldots, s$. For all $k \in [s]$, gate G_k is represented as a tuple $(a_k, b_k, c_k) \in [t]^3$ where input wires are a_k and b_k and output wire is c_k. For each $i \in [\ell]$ and $j \in [t]$, let $v_{i,j} \in \{0,1\}$ be the value of j-th wire when evaluating C on input $(\mathbf{x}_i, \mathbf{w}_i)$. For each $j \in [t]$, a univariate polynomial $\Phi_j(X)$ of degree at most $\ell - 1$ is generated so that $\Phi_j(i) = v_{i,j}$ for all $i \in [\ell]$. Two validity checks are carried out:

– **Wire validity checks.** Prove that $v_{i,j} \in \{0,1\}$ for all $i \in [\ell]$ and $j \in [t]$. This is equivalent to

$$\begin{aligned} & \Phi_j(i)^2 - \Phi_j(i) = 0 && \forall i \in [\ell], j \in [t] \\ \Leftrightarrow \quad & Z_\ell(X) \mid \Phi_j(X)^2 - \Phi_j(X), && \forall j \in [t] \end{aligned}$$

where $Z_\ell(X) = \prod_{i \in [\ell]} (X - i)$ is the vanishing polynomial over $[\ell]$. Therefore, it suffices to give a set of polynomials $\{Q_j(X)\}_{j \in [t]}$ such that

$$Q_j(X) \cdot Z_\ell(X) = \Phi_j(X)^2 - \Phi_j(X) \quad \forall j \in [t].$$

– **Gate validity checks.** Prove that $v_{i,c_k} = \mathsf{NAND}(v_{i,a_k}, v_{i,b_k})$ for all $i \in [\ell]$ and $k \in [s]$. This is equivalent to

$$\begin{aligned} & 1 - v_{i,c_k} - v_{i,a_k} \cdot v_{i,b_k} = 0 && \forall i \in [\ell], k \in [s] \\ \Leftrightarrow \quad & 1 - \Phi_{c_k}(i) - \Phi_{a_k}(i) \cdot \Phi_{b_k}(i) = 0 && \forall i \in [\ell], k \in [s] \\ \Leftrightarrow \quad & Z_\ell \mid 1 - \Phi_{c_k}(X) - \Phi_{a_k}(X) \cdot \Phi_{b_k}(X) && \forall k \in [s] \end{aligned}$$

where $Z_\ell(X) = \prod_{i \in [\ell]} (X - i)$ is the vanishing polynomial over $[\ell]$. Therefore, it suffices to give a set of polynomials $\{R_k(X)\}_{k \in [s]}$ such that

$$R_k(X) \cdot Z_\ell(X) = 1 - \Phi_{c_k}(X) - \Phi_{a_k}(X) \cdot \Phi_{b_k}(X) \quad \forall k \in [s].$$

Following KZG-style polynomial commitment [26], the common reference string (CRS) contains encoding of the powers of a hidden evaluation point τ together with an encoding of the vanishing polynomial:

$$\mathsf{crs} = (\{[\tau^i]\}_{i \in [0, \ell-1]}, [Z_\ell(\tau)])$$

and the proof consists of

$$\{[\Phi_j(\tau)]\}_{j \in [t]}, \ \{[Q_j(\tau)]\}_{j \in [t]}, \ \{[R_k(\tau)]\}_{k \in [s]}.$$

The scheme achieves $|\mathsf{crs}| = \ell \cdot \mathsf{poly}(\lambda)$ and $|\pi| = |C| \cdot \mathsf{poly}(\lambda)$.

Tool: Bivariate Zero-tests [17,28]. Let $m, n \in \mathbb{N}$ and $I, J \subseteq \mathbb{N}$. For a bivariate polynomial $F(X,Y) \in \mathbb{F}_{m,n}[X,Y]$, we have

$$F(i,j) = 0, \forall (i,j) \in I \times J \quad \Leftrightarrow \quad F(X,Y) \in \langle Z_I(X), Z_J(Y) \rangle$$

where $Z_I(X) = \prod_{i \in I}(X - i) \in \mathbb{F}_{|I|}[X]$ and $Z_J(Y) = \prod_{j \in J}(Y - j) \in \mathbb{F}_{|J|}[Y]$ are the vanishing polynomials. Therefore, it suffices to give two polynomials $Q_I(X,Y) \in \mathbb{F}_{m-|I|,n}[X,Y]$ and $Q_J(X,Y) \in \mathbb{F}_{m,n-|J|}[X,Y]$ such that

$$Q_I(X,Y) \cdot Z_I(X) + Q_J(X,Y) \cdot Z_J(Y) = F(X,Y)$$

Wire Encoding and Validity Check. Instead of $\{\Phi_j(X)\}_{j \in [t]}$, we define a *bivariate* polynomial $\Phi(X,Y) \in \mathbb{F}_{\ell-1,t-1}[X,Y]$ such that

$$\Phi(i,j) = v_{i,j} \quad \forall i \in [\ell], j \in [t].$$

The wire validity check becomes:

$$\begin{aligned} & \Phi(i,j)^2 - \Phi(i,j) = 0 && \forall i \in [\ell], j \in [t] \\ \Leftrightarrow \quad & \Phi(X,Y)^2 - \Phi(X,Y) \in \langle Z_\ell(X), Z_t(Y) \rangle \end{aligned}$$

where Z_ℓ is defined as before and $Z_t(Y) = \prod_{j \in [t]}(Y - j)$ is the vanishing polynomial over $[t]$. Therefore, it suffices to give two polynomials $Q_\ell(X,Y)$ and $Q_t(X,Y)$ such that

$$Q_\ell(X,Y) \cdot Z_\ell(X) + Q_t(X,Y) \cdot Z_t(Y) = \Phi(X,Y)^2 - \Phi(X,Y).$$

Public-Input Consistency. In addition to proving that all encoded wire values are Boolean, we must ensure that the first n wires agree with the public statements. Define the public-input interpolation polynomial

$$\Phi_{\mathsf{in}}(i,j) = x_{i,j} \qquad \forall i \in [\ell],\ j \in [n].$$

Public-input consistency is equivalent to

$$\Phi(X,Y) - \Phi_{\mathsf{in}}(X,Y) \in \langle Z_\ell(X), Z_n(Y) \rangle.$$

where $Z_n(Y) = \prod_{j \in [n]}(Y - j)$ is the vanishing polynomial over $[n]$. Thus, it suffices to provide polynomials $Q_\ell^{\mathsf{in}}(X,Y)$ and $Q_n^{\mathsf{in}}(X,Y)$ satisfying

$$\Phi(X,Y) - \Phi_{\mathsf{in}}(X,Y) = Q_\ell^{\mathsf{in}}(X,Y) Z_\ell(X) + Q_n^{\mathsf{in}}(X,Y) Z_n(Y).$$

Since Φ_{in} is determined by the public statements, its commitment can be computed directly by the verifier.

Gate Validity Check. Gate validity check becomes

$$\begin{aligned} & 1 - v_{i,c_k} - v_{i,a_k} \cdot v_{i,b_k} = 0 \qquad \forall i \in [\ell], k \in [s] \\ \Leftrightarrow\ & 1 - \Phi(i, c_k) - \Phi(i, a_k) \cdot \Phi(i, b_k) = 0 \quad \forall i \in [\ell], k \in [s] \end{aligned}$$

Define three auxiliary maps $A(Z), B(Z), C(Z) : [s] \to [t]$ such that

$$A(k) = a_k, \quad B(k) = b_k, \quad C(k) = c_k \quad \forall k \in [s].$$

The last row becomes

$$1 - \Phi(i, C(k)) - \Phi(i, A(k)) \cdot \Phi(i, B(k)) = 0 \quad \forall i \in [\ell], \forall k \in [s]$$

Let

$$\Phi_a(X, Z) = \Phi(X, A(Z)), \quad \Phi_b(X, Z) = \Phi(X, B(Z)), \quad \Phi_c(X, Z) = \Phi(X, C(Z))$$

we work with

$$1 - \Phi_c(i, k) - \Phi_a(i, k) \cdot \Phi_b(i, k) = 0 \quad \forall i \in [\ell], \forall k \in [s]$$

which can be handled as before. That is,

$$\begin{aligned} & 1 - \Phi_c(i, k) - \Phi_a(i, k) \cdot \Phi_b(i, k) = 0 \qquad \forall i \in [\ell], k \in [s] \\ \Leftrightarrow\ & 1 - \Phi_c(X, Z) - \Phi_a(X, Z)\Phi_b(X, Z) \in \langle Z_\ell(X), Z_s(Z) \rangle \end{aligned}$$

Thus, it suffices to give two polynomials $Q'_\ell(X, Z)$ and $Q'_s(X, Z)$ such that

$$Q'_\ell(X, Z) \cdot Z_\ell(X) + Q'_s(X, Z) \cdot Z_s(Z) = 1 - \Phi_c(X, Z) - \Phi_a(X, Z)\Phi_b(X, Z).$$

Gate-Wire Consistency. We prove that Φ and Φ_a, Φ_b, Φ_c are consistent in the sense that

$$\Phi_a(X, Z) = \Phi(X, A(Z)), \quad \Phi_b(X, Z) = \Phi(X, B(Z)), \quad \Phi_c(X, Z) = \Phi(X, C(Z))$$

Let us take Φ_a as an example. Rewrite the definition in a sum-check fashion

$$\Phi_a(X, Z) = \sum_{y \in \mathbb{V}} \Phi(X, y) \cdot \mathsf{eq}_a(Z, y)$$

where $\mathbb{V}$ is a subgroup of size t and eq_a is defined as follows:

$$\mathsf{eq}_a(u, v) = \begin{cases} 1, & A(u) = v \\ 0, & A(u) \neq v \end{cases}.$$

Here eq_a depends on C. By univariate sum-check, this can be proved by showing $H_a(Y) \in \mathbb{Z}_p^{\leq D-t}[Y]$ and $F_a(Y) \in \mathbb{Z}_p^{<t-1}[Y]$ such that

$$\Phi(X,Y) \cdot \mathsf{eq}_a(Z,Y) - \Phi_a(X,Z) \cdot t^{-1} = H_a(Y) \cdot Z_t(Y) + Y \cdot F_a(Y).$$

The consistency of Φ_b and Φ_c can be verified analogously. Therefore, the CRS contains

$$\mathsf{crs} = (\{[\tau_X^i \tau_Y^j], [\tau_X^i \tau_Z^k], [\tau_Y^j \tau_Z^k]\}_{i\in[0,\ell],j\in[0,t],k\in[0,s]})$$

and the proof consists of

- encoding: $[\Phi(\tau_X,\tau_Y), \Phi_a(\tau_X,\tau_Z), \Phi_b(\tau_X,\tau_Z), \Phi_c(\tau_X,\tau_Z)]$
- wire validity: $[Q_\ell(\tau_X,\tau_Y), Q_t(\tau_X,\tau_Y)]$
- public-input consistency: $[Q_\ell^{\mathsf{in}}(\tau_X,\tau_Y), Q_n^{\mathsf{in}}(\tau_X,\tau_Y)]$
- gate validity: $[Q'_\ell(\tau_X,\tau_Z), Q'_s(\tau_X,\tau_Z)]$
- wire-gate consistency: $[H_a(\tau_Y), F_a(\tau_Y), H_b(\tau_Y), F_b(\tau_Y), H_c(\tau_Y), F_c(\tau_Y)]$

The verification checks:

$$\begin{aligned}
&[Q_\ell(\tau_X,\tau_Y)] \bullet [Z_\ell(\tau_X)] + [Q_t(\tau_X,\tau_Y)] \bullet [Z_t(\tau_Y)] \\
&\qquad = [\Phi(\tau_X,\tau_Y)] \bullet [\Phi(\tau_X,\tau_Y)] - [\Phi(\tau_X,\tau_Y)] \bullet [1]. \\
&[\Phi(\tau_X,\tau_Y)] \bullet [1] = [\Phi_{\mathsf{in}}(\tau_X,\tau_Y)] \bullet [1] \\
&\qquad + [Q_\ell^{\mathsf{in}}(\tau_X,\tau_Y)] \bullet [Z_\ell(\tau_X)] + [Q_n^{\mathsf{in}}(\tau_X,\tau_Y)] \bullet [Z_n(\tau_Y)]. \\
&[Q'_\ell(\tau_X,\tau_Z)] \bullet [Z_\ell(\tau_X)] + [Q'_s(\tau_X,\tau_Z)] \bullet [Z_s(\tau_Z)] \\
&\qquad = [1]_T - [\Phi_c(\tau_X,\tau_Z)] \bullet [1] - [\Phi_a(\tau_X,\tau_Z)] \bullet [\Phi_b(\tau_X,\tau_Z)]. \\
&[\Phi(\tau_X,\tau_Y)] \bullet [\mathsf{eq}_a(\tau_Z,\tau_Y)] - [\Phi_a(\tau_X,\tau_Z)] \bullet [t^{-1}] \\
&\qquad = [H_a(\tau_Y)] \bullet [Z_t(\tau_Y)] + [\tau_Y] \bullet [F_a(\tau_Y)].
\end{aligned}$$

where $[Z_n(\tau_Y)]$, $[Z_\ell(\tau_X)]$, $[Z_t(\tau_Y)]$, $[Z_s(\tau_Z)]$, $[\mathsf{eq}_a(\tau_Z,\tau_Y)]$ are computed from crs.

Weak Extractability in the Standard Model. To discuss extractability, we switch to the *trapdoor mode*, namely, the CRS is generated by a trapdoor setup algorithm for a tuple of hidden target indices. Concretely, we work in a composite-order bilinear group $\mathbb{G}$ of order $N = pq$, with subgroups $\mathbb{G}_p$ and $\mathbb{G}_q$ generated by g_p and g_q, respectively. The $\mathbb{G}_p$-part plays the role of the normal commitment key, while the $\mathbb{G}_q$-part is used to embed hidden indices and enable extraction. At a high level, a trapdoor CRS replaces the monomial encodings $[\tau_X^i \tau_Y^j]$, $[\tau_X^i \tau_Z^k]$ with

$$[\tau_X^i \tau_Y^j]_{\mathsf{trap}} := g_p^{\tau_X^i \tau_Y^j} \cdot g_q^{r_{i,j}}, \quad [\tau_X^i \tau_Z^k]_{\mathsf{trap}} := g_p^{\tau_X^i \tau_Z^k} \cdot g_q^{r_{i,k}}$$

where $r_{i,j}$ and $r_{i,k}$ are chosen so as to encode the desired hidden location. For bivariate polynomials $\Phi(X,Y) = \sum_{i,j} \phi_{i,j} X^i Y^j$, $\Phi_a(X,Z) = \sum_{i,k} \phi_{a,i,k} X^i Z^k$, $\Phi_b(X,Z) = \sum_{i,k} \phi_{b,i,k} X^i Z^k$, $\Phi_c(X,Z) = \sum_{i,k} \phi_{c,i,k} X^i Z^k$, their commitments in the trapdoor mode carry the hidden-subgroup components

$$\pi_{\mathbb{G}_q}(\mathsf{com}(\Phi)) = g_q^{\sum_{i,j} r_{i,j}\phi_{i,j}},\ \pi_{\mathbb{G}_q}(\mathsf{com}(\Phi_a)) = g_q^{\sum_{i,k} r_{i,k}\phi_{a,i,k}},$$
$$\pi_{\mathbb{G}_q}(\mathsf{com}(\Phi_b)) = g_q^{\sum_{i,k} r_{i,k}\phi_{b,i,k}},\ \pi_{\mathbb{G}_q}(\mathsf{com}(\Phi_c)) = g_q^{\sum_{i,k} r_{i,k}\phi_{c,i,k}}.$$

Weak somewhere extractability in the trapdoor mode and somewhere soundness. The trapdoor is associated with a hidden target tuple

$$(i^*, j^*, k^*) \in [\ell] \times [t] \times [s].$$

The G_q-components of the CRS are programmed so that

$$r_{i,j} = (i^*)^i (j^*)^j,\ r_{i,k} = (i^*)^i (k^*)^k.$$

As a result, the hidden-subgroup projection of the commitments to Φ, Φ_a, Φ_b and Φ_c become

$$\pi_{\mathbb{G}_q}(\mathsf{com}(\Phi)) = g_q^{\Phi(i^*,j^*)},\ \pi_{\mathbb{G}_q}(\mathsf{com}(\Phi_a)) = g_q^{\Phi_a(i^*,k^*)},$$
$$\pi_{\mathbb{G}_q}(\mathsf{com}(\Phi_b)) = g_q^{\Phi_b(i^*,k^*)},\ \pi_{\mathbb{G}_q}(\mathsf{com}(\Phi_c)) = g_q^{\Phi_c(i^*,k^*)}.$$

The extractor therefore obtains a global target-wire value $\widehat{v}_{i^*,j^*}$:

$$\widehat{v}_{i^*,j^*} = 0 \quad \text{if } g_q^{\Phi(i^*,j^*)} = 1, \qquad \widehat{v}_{i^*,j^*} = 1 \quad \text{otherwise}$$

and the gate-local values $\widehat{\boldsymbol{v}}_{i^*,G_{k^*}} = \left(\widehat{v}_{i^*,a_{k^*}}, \widehat{v}_{i^*,b_{k^*}}, \widehat{v}_{i^*,c_{k^*}}\right)$:

$$\widehat{v}_{i^*,a_{k^*}} = 0 \quad \text{if } g_q^{\Phi_a(i^*,k^*)} = 1, \qquad \widehat{v}_{i^*,a_{k^*}} = 1 \quad \text{otherwise;}$$
$$\widehat{v}_{i^*,b_{k^*}} = 0 \quad \text{if } g_q^{\Phi_b(i^*,k^*)} = 1, \qquad \widehat{v}_{i^*,b_{k^*}} = 1 \quad \text{otherwise;}$$
$$\widehat{v}_{i^*,c_{k^*}} = 0 \quad \text{if } g_q^{\Phi_c(i^*,k^*)} = 1, \qquad \widehat{v}_{i^*,c_{k^*}} = 1 \quad \text{otherwise.}$$

The verification equations ensure that these extracted values are Boolean and satisfy the following conditions:

- if $j^* \in [n]$, then $\widehat{v}_{i^*,j^*} = x_{i^*,j^*}$;
- the gate-local values satisfy

$$\widehat{v}_{i^*,c_{k^*}} = \mathsf{NAND}\left(\widehat{v}_{i^*,a_{k^*}}, \widehat{v}_{i^*,b_{k^*}}\right);$$

- if $j^* \in \{a_{k^*}, b_{k^*}, c_{k^*}\}$, then the extracted global wire value agrees with the corresponding gate-local value;
- if the extracted wire is the circuit output, then its value is 1, and if the output of G_{k^*} is the circuit output, then $\widehat{v}_{i^*,c_{k^*}} = 1$.

These local extraction guarantees imply somewhere soundness: an accepting proof for a false hidden instance must violate either a gate constraint or a gate–wire consistency constraint, which is detected with inverse-polynomial probability over the hidden target indices.

2.2 Application 1: NIZK from BARG

We show how to apply our BARG construction to obtain a non-interactive zero-knowledge (NIZK) argument for NP. We follow the hidden-bits-generator (HBG) framework of [10,4]:

$$\text{NIZK for NP in HBM + HBG} \Longrightarrow \text{NIZK for NP in CRS.}$$

Hidden-bits model. In this model, a trusted party gives the prover a random string $\mathbf{r} \in \{0,1\}^m$ and gives $\mathbf{r}$ to the prover. To construct a NIZK proof π_{NIZK} for a statement x, the prover chooses a subset of the bits $\mathbf{r}_I = \{r_i\}_{i \in I}$ for indices $I \subseteq [m]$ and a proof π_I. Then the verifier receives the bits $\mathbf{r}_I$ and π_I from the trusted party.

Hidden-bits generators. An HBG cryptographically realizes this trusted sampling of the hidden-bits string in HBM. Concretely, an HBG with subset-dependent proofs consists of four algorithms

$$\Pi_{\mathsf{HBG}} = (\mathsf{HBG.Setup}, \mathsf{HBG.GBits}, \mathsf{HBG.Prove}, \mathsf{HBG.Verify}).$$

Π_{HBG} is required to satisfy hiding and binding properties.

Champion-Wu HBG [10]. Their HBG uses a leakage-resilient local pseudorandom generator (PRG) with small locality. The prover samples a short PRG seed $\mathsf{sd} \in \{0,1\}^n$, commits to each bit of sd using a dual-mode commitment, and then runs a BARG to prove that for every opened index $i \in I$, the revealed bit $r_i = \mathsf{PRG}_i(\mathsf{sd})$. The final HBG proof is

$$\pi_{\mathsf{HBG}} = (\pi_{\mathsf{BARG}}, \{\mathsf{com}_j^{\mathsf{sd}}\}_{j \in [n]}).$$

Since the BARG proof π_{BARG} may leak information about the seed, its $|\pi_{\mathsf{BARG}}|$ size must remain below the leakage-resilience budget of the PRG. This forces the relation circuit to be small, and hence necessitates a local PRG where each output bit depends on only a few input bits. Consequently, since π_{BARG} depends on the PRG circuit size, their construction requires a complicated selection of parameters to balance the leakage bound against the leakage resilience of the PRG.

Bradley-Waters-Wu HBG [4]. To remove the locality assumption, they additionally commit to all internal wire values of the circuit that computes the PRG (or more generally a leakage-resilient weak PRF). For each index $i \in I$, the BARG verifies every gate of that circuit, including the consistency of all committed wires. The final HBG proof is

$$\pi_{\mathsf{HBG}} = (\pi_{\mathsf{BARG}}, \{\mathsf{com}_j^{\mathsf{rk}}\}_{j \in [\kappa]}, \{\mathsf{com}_{i,j}^{\mathsf{wire}}\}_{i \in I, j \in [t_{\mathsf{LRwPRF}}]}),$$

where rk with $|\mathsf{rk}| = \kappa$ is the LRwPRF key and t_{LRwPRF} is the number of wires in the LRwPRF circuit. This avoids locality but introduces a large number of additional commitments (one per wire per opened instance). Consequently, the proof size scales with the total wire count, limiting succinctness. Like the construction [10], it also needs delicate parameter choice to keep the BARG leakage within the PRF's leakage budget. Moreover, they shows that one-time equivocation for a bit commitment suffices for the security analysis.

Our approach. We instantiate the HBG using our BARG (which has constant-size proofs) combined with:

– a one-time dual-mode commitment scheme with extraction

$$\Pi_{\mathsf{BC}} = (\mathsf{BC.SetupBind}, \mathsf{BC.SetupEquiv}, \mathsf{BC.Commit}, \mathsf{BC.Verify}, \mathsf{BC.Extract}),$$

– and a leakage-resilient weak pseudorandom function

$$\Pi_{\mathsf{LRwPRF}} = (\mathsf{LRwPRF.Setup}, \mathsf{LRwPRF.Eval}).$$

The HBG works as follows:

– To generate a hidden-bits string, run $\mathsf{rk} \leftarrow \mathsf{LRwPRF.Setup}$ and set hidden bits $r_i \leftarrow \mathsf{LRwPRF.Eval}(\mathsf{rk}, \mathbf{z}_i)$ for each $i \in [m]$ and public inputs $\mathbf{z}_i$.
– To open a subset $\mathbf{r}_I$ for $I \subseteq [m]$, the prover does the following:
  • Compute $\mathsf{com}_{\mathsf{rk}_j} \leftarrow \mathsf{BC.Commit}(\mathsf{rk}_j)$ for each bit $j \in [\kappa]$.
  • Define the NP relation $\mathcal{R}$ as

$$\begin{aligned} \text{statement:} \quad & \mathbf{x}_i = \left(C_{\mathsf{LRwPRF}}(\cdot), \mathbf{z}_i, \{\mathsf{com}_{\mathsf{rk}_j}\}_{j\in[\kappa]},\ r_i\right) \\ \text{witness:} \quad & \mathbf{w}_i = \left(\{\mathsf{rk}_j,\ \sigma_j\}_{j\in[\kappa]}\right) \end{aligned}$$

    where C_{LRwPRF} is the circuit computing LRwPRF.Eval and σ_j is the opening proof corresponding to $\mathsf{com}_{\mathsf{rk}_j}$. Output 1 if the following conditions hold

$$\begin{cases} \mathsf{BC.Verify}(\mathsf{crs}_{\mathsf{BC}}, \mathsf{com}_{\mathsf{rk}_j}, \mathsf{rk}_j, \sigma_{\mathsf{rk}_j}) = 1 \quad \forall\, j \in [\kappa] \\ r_i = \mathsf{LRwPRF.Eval}(\mathsf{rk}, \mathbf{z}_i) \end{cases}$$

    Both conditions are compiled into a single NAND circuit $C_{\mathcal{R}}$ with t wires and s gates. Compute

$$\pi_{\mathsf{BARG}} \leftarrow \mathsf{BARG.Prove}(\mathsf{crs}_{\mathsf{BARG}}, C_{\mathcal{R}}, \{\mathbf{x}_i, \mathbf{w}_i\}_{i\in I}).$$

    The final HBG proof is

$$\pi_{\mathsf{HBG}} = (\pi_{\mathsf{BARG}}, \{\mathsf{com}_j^{\mathsf{rk}}\}_{j\in[\kappa]}).$$

As summarized in Fig. 2, our approach differs from prior HBG constructions [10,4] in three aspects: (i) we rely on a constant-size BARG proof, which limits leakage on the PRF key to a constant; (ii) we avoid committing to internal wire values, keeping the proof size independent of the circuit size; (iii) we do not require the PRG to have small locality, instead using a leakage-resilient weak PRF. Consequently, our HBG achieves a simpler and more efficient instantiation.

Scheme	No locality	No wire commitment	Underlying BARG	
			constant proof	security
[10]	✗	✓	✗	SAS
[4]	✓	✗	✗	AS
Ours	✓	✓	✓	sSound

Fig. 2: Comparison of HBG instantiations. Here, SAS denotes semi-adaptive soundness, AS denotes adaptive soundness, and sSound denotes somewhere soundness, which in our construction is derived from weak somewhere extractability.

Security analysis. Unlike the construction of [4], which only relies on one-time equivocation for hiding and statistical binding for its external commitments Π_{BC}, our adaptation additionally assumes that the bit commitment is extractable in the binding mode. This allows us to identify a unique PRF key shared across all instances.

- Binding. In the binding mode, the extractable commitments determine a unique key $\widehat{\mathsf{rk}}$. If $r_i \neq \mathsf{LRwPRF.Eval}(\widehat{\mathsf{rk}}, \mathbf{z}_i)$ for some opened index $i \in I$, then the corresponding BARG instance is false, while the BARG proof is accepting. This contradicts the somewhere soundness of the underlying BARG, which follows from its weak somewhere extractability. Therefore, binding reduces to the extractability of the outer key commitments and the somewhere soundness of the BARG.
- Hiding. Switching the commitments to equivocable mode hides rk statistically. The only leakage on rk is the BARG proof π_{BARG}, whose size is constant. Hence the leakage is bounded by $|\pi_{\mathsf{BARG}}|$, which we set to be within the leakage tolerance of the LRwPRF. It follows that the unrevealed bits remain pseudorandom given $(\mathsf{crs}, \mathbf{r}_I, \pi_{\mathsf{HBG}})$. Therefore, hiding reduces to the equivocable mode of the key commitment and the leakage resilience of LRwPRF.

2.3 Application 2: Rate-1 BARG from Pairing

Rate-1 BARGs from pairings are already known via non-black-box techniques [8], where the proof size essentially equals to the size of a single witness. In this subsection, we discuss how the somewhere-sound BARG obtained from our weak somewhere-extractable construction can be combined with these the non-black-box techniques to obtain a rate-1 weak seBARG. Let ℓ be the number of instances, and $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ be the circuit defining the NP relation.

The DGKV compiler. Devadas et al. [15] gave a generic compiler for rate-1 BARGs from three ingredients:

- a rate-1 fully-local somewhere-extractable hash (flSEH);

– a flexible RAM SNARG with partial input soundness [23];
– an index somewhere-extractable BARG.

The compiler works as follows.

– The prover hashes the witnesses $W = (\mathbf{w}_1, \ldots, \mathbf{w}_\ell)$ using a rate-1 flSEH, producing a short hash value hv and a digest rt.
– For each instance $i \in [\ell]$, a RAM machine $M^{C,i}_{\mathsf{RAM}}$ takes the public input $(C, \mathbf{x}_1, \ldots, \mathbf{x}_\ell, \mathsf{hv}, \mathsf{rt})$ and index i, reads only the i-th witness block $\mathbf{w}_i$ from the hashed memory, verifies its consistency with $(\mathsf{hv}, \mathsf{rt})$, and checks $C(\mathbf{x}_i, \mathbf{w}_i) = 1$. The prover generates a short RAM SNARG proof π_i, attesting to the correct execution of RAM $M^{C,i}_{\mathsf{RAM}}$.
– Instead of sending all π_i, the prover runs an index seBARG to prove that for every $i \in [\ell]$, there exists an accepting π_i. Define the index relation

$$R_{\mathsf{RAM}}\big((C, \mathbf{x}_1, \ldots, \mathbf{x}_\ell, \mathsf{hv}, \mathsf{rt}),\ i,\ \pi_i\big) = 1 \iff \pi_i \text{ is valid for } M^{C,i}_{\mathsf{RAM}}.$$

  The prover computes

$$\pi_{\mathsf{BARG}} \leftarrow \mathsf{BARG.Prove}(\mathsf{crs}_{\mathsf{BARG}}, C_{\mathsf{RAM}}, \{(\mathbf{x}'_i, \mathbf{w}'_i)\}_{i\in[\ell]}),$$

  where $\mathbf{x}'_i$ contains all public inputs and $\mathbf{w}'_i = \pi_i$, and $\mathsf{C}_{\mathsf{RAM}}$ is the circuit verifying R_{RAM}.
– Output the final proof $\pi_{\mathsf{r1BARG}} = (\mathsf{hv}, \mathsf{rt}, \pi_{\mathsf{BARG}})$.

The proof length is $|\pi_{\mathsf{r1BARG}}| = h + \mathsf{poly}(\lambda, \log \ell)$, but it relies on LWE assumption.

The BDSZ construction. Branco et al. [8] follow the above framework. Their main contribution is a new construction of a rate-1 fully-local SEH from DDH, which in turn yields rate-1 BARGs from subexponential DDH or k-LIN. The resulting proof length is $|\pi_{\mathsf{r1BARG}}| = h + \mathsf{poly}(\lambda, \log \ell)$.

Our approach. Our construction follows the high-level framework [8], but it does not use a RAM SNARG. Instead, we use the somewhere-soundness property of our pairing-based BARG directly to certify the local-checking relation for all instances. We define a direct local-checking circuit C_{loc}. This circuit C_{loc} verifies the consistency with $(\mathsf{hv}, \mathsf{rt})$, and checks $C(\mathbf{x}_i, \mathbf{w}_i) = 1$ for each instance $i \in [\ell]$. We define the local-checking relation

$$\mathbf{x}'_i = (C, \mathbf{x}_i, \mathsf{vk}_{\mathsf{flSEH}}, \mathsf{hv}, \mathsf{rt}, i),$$

$$\mathbf{w}'_i = (\mathbf{w}_i, \rho_i).$$

Here, ρ_i is the corresponding flSEH opening. The prover computes

$$\pi_{\mathsf{BARG}} \leftarrow \mathsf{BARG.Prove}\big(\mathsf{crs}_{\mathsf{BARG}}, C_{\mathsf{loc}}, \{(\mathbf{x}'_i, \mathbf{w}'_i)\}_{i\in[\ell]}\big).$$

The final proof is $\pi_{\mathsf{r1BARG}} = (\mathsf{hv}, \mathsf{rt}, \pi_{\mathsf{BARG}})$. The resulting proof length is $|\pi_{\mathsf{r1BARG}}| = h + \mathsf{poly}(\lambda, \log \ell)$.

Security analysis. The rate-1 flSEH extractor recovers a witness $\widehat{\mathbf{w}}_{i^*}$ for the hidden instance. Except with negligible probability, every collection of valid local openings at the hidden positions is consistent with $\widehat{\mathbf{w}}_{i^*}$. If $C(\mathbf{x}_{i^*}, \widehat{\mathbf{w}}_{i^*}) \neq 1$, then the corresponding instance of the local-checking relation is false. An accepting BARG proof would therefore contradict the somewhere soundness of the underlying BARG, established for Construction 4.2 by Corollary 1. Hence, $C(\mathbf{x}_{i^*}, \widehat{\mathbf{w}}_{i^*}) = 1$ except with negligible probability.

3 Preliminaries

Notations. For an integer $n \in \mathbb{N}$, we write $[n] := \{1, \ldots, n\}$. For a finite set S, we use $s \leftarrow S$ to denote the procedure of sampling s from S uniformly. Let $\approx_s$ (resp. $\approx_c$) stand for statistical (resp. computational) indistinguishability. We use lower-case boldface to denote (column) vectors (e.g., $\mathbf{a}$) and upper-case boldface to denote matrices (e.g. $\mathbf{M}$). We use $\mathbf{a}[i]$ (resp. $\mathbf{M}[i, j]$) to indicate the entry in position i (resp. (i, j)) and use $\mathbf{a}_i$ (resp. $\mathbf{M}_i$) to denote i-th vector (resp. i-th Matrix). We write $\mathbf{a}_{1:\ell} := \{\mathbf{a}_1, \ldots, \mathbf{a}_\ell\}$. We write $\mathrm{poly}(\lambda)$ to denote a fixed function that is bounded by some polynomial in λ and $\varepsilon(\lambda)$ to denote a function that is negligible in λ. We say that a probabilistic algorithm is PPT if it runs in polynomial time.

3.1 Composite-Order Bilinear Groups

Assume an efficient algorithm CompGroupGen that takes as input security parameter λ and outputs $\mathcal{G} := (p, q, \mathbb{G}, \mathbb{G}_T, g, e)$. Here $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of prime order $N = pq$ where $p, q > 2^\lambda$, $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a non-degenerate bilinear map, and all group operations and bilinear map are efficient.

3.2 Batch Arguments

A *non-interactive batch argument* (BARG) [32,12] for circuit satisfiability is defined by the following algorithms:

- $\mathsf{Setup}(\lambda, \ell, t, s) \to \mathsf{crs}$: It takes as input a security parameter $\lambda \in \mathbb{N}$, the number of instances $\ell \in \mathbb{N}$, upper bound of circuit wire size $t \in \mathbb{N}$ and upper bound of circuit gate size $s \in \mathbb{N}$, outputs a common reference string crs.
- $\mathsf{Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_\ell), (\mathbf{w}_1, \ldots, \mathbf{w}_\ell)) \to \pi$: It takes as input crs, a Boolean circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$, statements $\mathbf{x}_1, \ldots, \mathbf{x}_\ell \in \{0,1\}^n$ and witnesses $\mathbf{w}_1, \ldots, \mathbf{w}_\ell \in \{0,1\}^h$, outputs a proof π.
- $\mathsf{Verify}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_\ell), \pi) \to b$: It takes as input crs, $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$, $\mathbf{x}_1, \ldots, \mathbf{x}_\ell \in \{0,1\}^n$ and a proof π, outputs $b \in \{0,1\}$.

Here, we require that Verify is deterministic and the batch argument satisfies the following properties:

Completeness. For all $\lambda, \ell, t, s \in \mathbb{N}$, all Boolean circuits $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ of size at most $(t(\lambda), s(\lambda))$, all statements $\mathbf{x}_1, \ldots, \mathbf{x}_\ell \in \{0,1\}^n$, and all witnesses $\mathbf{w}_1, \ldots, \mathbf{w}_\ell \in \{0,1\}^h$ satisfying $C(\mathbf{x}_i, \mathbf{w}_i) = 1$ for all $i \in [\ell]$, it holds that

$$\Pr\left[\mathsf{Verify}(\mathsf{crs}, C, \mathbf{x}_{1:\ell}, \pi) = 1 \,\middle|\, \begin{array}{l} \mathsf{crs} \leftarrow \mathsf{Setup}(\lambda, \ell, t, s); \\ \pi \leftarrow \mathsf{Prove}(\mathsf{crs}, C, \mathbf{x}_{1:\ell}, \mathbf{w}_{1:\ell}) \end{array}\right] = 1.$$

Weak somewhere extractable. There exists a pair of efficient algorithms TrapSetup and Extract with following syntax:

- $\mathsf{TrapSetup}(1^\lambda, \ell, t, s, i^*, j^*, k^*) \to (\mathsf{crs}, \mathsf{td})$ takes as input the security parameter, the batch size, the wire and gate bounds, and target indices $(i^*, j^*, k^*) \in [\ell] \times [t] \times [s]$, and outputs a common reference string crs and an extraction trapdoor td.
- $\mathsf{Extract}(\mathsf{td}, C, x_{1:\ell}, \pi) \to (\widehat{v}_{i^*,j^*}, \widehat{v}_{i^*,G_{k^*}})$ takes as input the extraction trapdoor, a circuit C, statements $x_{1:\ell}$, and a proof π, and outputs

$$\widehat{v}_{i^*,j^*} \in \{0,1\} \quad \text{and} \quad \widehat{v}_{i^*,G_{k^*}} = \left(\widehat{v}^a_{i^*,k^*}, \widehat{v}^b_{i^*,k^*}, \widehat{v}^c_{i^*,k^*}\right) \in \{0,1\}^3,$$

  where $G_{k^*} = (a_{k^*}, b_{k^*}, c_{k^*})$ is the k^*-th gate of C.

The target indices (i^*, j^*, k^*) are sampled by the security experiment and are not given to the adversary. The following properties are required.

CRS indistinguishability. For every PPT adversary $\mathcal{A}$, define

$$\mathsf{Adv}^{\mathsf{crs\text{-}ind}}_{\Pi_{\mathsf{BARG}},\mathcal{A}}(\lambda, \ell, t, s) := \left| \Pr\left[\begin{array}{l} b = 1 : (i^*, j^*, k^*) \leftarrow [\ell] \times [t] \times [s]; \\ \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, \ell, t, s); \\ b \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right] - \Pr\left[\begin{array}{l} b = 1 : (i^*, j^*, k^*) \leftarrow [\ell] \times [t] \times [s]; \\ (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{TrapSetup}(1^\lambda, \ell, t, s, i^*, j^*, k^*); \\ b \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right] \right|.$$

We require

$$\mathsf{Adv}^{\mathsf{crs\text{-}ind}}_{\Pi_{\mathsf{BARG}},\mathcal{A}}(\lambda, \ell, t, s) \leq \mathsf{negl}(\lambda).$$

Weak somewhere extractability in trapdoor mode. Let

$$C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$$

be a Boolean circuit whose wires are indexed by $[t]$, where the first n wires contain the public input, the next h wires contain the witness, and $o_C \in [t]$ denotes the output wire. For $G_{k^*} =$

$(a_{k^*}, b_{k^*}, c_{k^*})$, define

$$\begin{aligned}\mathsf{Good} := \ & \left(j^* \in [n] \implies \widehat{v}_{i^*,j^*} = x_{i^*,j^*}\right) \\ & \wedge \left(j^* = a_{k^*} \implies \widehat{v}_{i^*,j^*} = \widehat{v}_{i^*,a_{k^*}}\right) \\ & \wedge \left(j^* = b_{k^*} \implies \widehat{v}_{i^*,j^*} = \widehat{v}_{i^*,b_{k^*}}\right) \\ & \wedge \left(j^* = c_{k^*} \implies \widehat{v}_{i^*,j^*} = \widehat{v}_{i^*,c_{k^*}}\right) \\ & \wedge \left(\widehat{v}_{i^*,c_{k^*}} = \mathrm{NAND}\left(\widehat{v}_{i^*,a_{k^*}}, \widehat{v}_{i^*,b_{k^*}}\right)\right) \\ & \wedge \left(j^* = o_C \implies \widehat{v}_{i^*,j^*} = 1\right) \\ & \wedge \left(c_{k^*} = o_C \implies \widehat{v}^{c}_{i^*,k^*} = 1\right).\end{aligned}$$

Define Bad := ¬Good.

For every PPT adversary $\mathcal{A}$, let

$$\begin{aligned}&\mathsf{Adv}^{\mathsf{wExt\text{-}td}}_{\Pi_{\mathsf{BARG}},\mathcal{A}}(\lambda, \ell, t, s) \\ &:= \Pr\left[\begin{array}{l}\mathsf{Verify}(\mathsf{crs}, C, x_{1:\ell}, \pi) = 1 \ \wedge \ \mathsf{Bad} : \\ (i^*, j^*, k^*) \xleftarrow{\$} [\ell] \times [t] \times [s]; \\ (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{TrapSetup}(1^\lambda, \ell, t, s, i^*, j^*, k^*); \\ (C, x_{1:\ell}, \pi) \leftarrow \mathcal{A}(\mathsf{crs}); \\ \left(\widehat{v}_{i^*,j^*}, \widehat{v}_{i^*,G_{k^*}}\right) \leftarrow \mathsf{Extract}(\mathsf{td}, C, x_{1:\ell}, \pi)\end{array}\right].\end{aligned}$$

We require

$$\mathsf{Adv}^{\mathsf{wExt\text{-}td}}_{\Pi_{\mathsf{BARG}},\mathcal{A}}(\lambda, \ell, t, s) \leq \mathsf{negl}(\lambda).$$

If this condition holds against computationally unbounded adversaries, then Π_{BARG} is statistically weak somewhere extractable in trapdoor mode.

Somewhere soundness. A BARG Π_{BARG} is *somewhere sound* if, for every PPT adversary $\mathcal{A}$, the following advantage is negligible:

$$\begin{aligned}&\mathsf{Adv}^{\mathsf{sSound}}_{\Pi_{\mathsf{BARG}},\mathcal{A}}(\lambda, \ell, t, s) \\ &:= \Pr\left[\begin{array}{l}\mathsf{Verify}(\mathsf{crs}, C, x_{1:\ell}, \pi) = 1 \\ \ \wedge \ \forall w \in \{0,1\}^h, \ C(x_{i^*}, w) = 0 : \\ i^* \xleftarrow{\$} [\ell]; \\ \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, \ell, t, s); \\ (C, x_{1:\ell}, \pi) \leftarrow \mathcal{A}(\mathsf{crs})\end{array}\right].\end{aligned}$$

We require that

$$\mathsf{Adv}^{\mathsf{sSound}}_{\Pi_{\mathsf{BARG}},\mathcal{A}}(\lambda, \ell, t, s) \leq \mathsf{negl}(\lambda).$$

Here, the instance index i^* is sampled uniformly and remains hidden from $\mathcal{A}$. Thus, an accepting proof cannot correspond to a false instance at the hidden index i^*, except with negligible probability.

Lemma 1 (Weak somewhere extractability implies somewhere soundness). *Let Π_{BARG} be a weak somewhere-extractable BARG for Boolean circuit satisfiability. Suppose that the extraction requirement includes the consistency of the extracted public-input, gate-local, gate-wire, and output-wire values. Then Π_{BARG} satisfies somewhere soundness.*

The proof of this lemma is deferred to Appendix C.

3.3 Hidden-Bits Generator

A hidden-bits generator with subset-dependent proofs is a tuple of efficient algorithms $\Pi_{\mathsf{HBG}} = (\mathsf{Setup}, \mathsf{GBits}, \mathsf{Prove}, \mathsf{Verify})$ with the following syntax:

- $\mathsf{Setup}(\lambda, m) \to \mathsf{crs}$. On input security parameter λ and output-length m, it outputs a common reference string crs.
- $\mathsf{GBits}(\mathsf{crs}) \to (\mathbf{r}, \mathsf{st})$. On input the common string crs, it outputs a hidden-bits bitstring $\mathbf{r} \in \{0,1\}^m$ and a generator state st.
- $\mathsf{Prove}(\mathsf{st}, I) \to \pi$. On input the state st and a subset of indices $I \subseteq [m]$, it outputs a succinct proof π that serves as an opening to the subset bits $\mathbf{r}_I \in \{0,1\}^{|I|}$.
- $\mathsf{Verify}(\mathsf{crs}, I, \mathbf{r}_I, \pi) \to b$. On input the common string crs, a subset I, a substring $\mathbf{r}_I \in \{0,1\}^{|I|}$ and a proofπ, it outputs a bit $b \in \{0,1\}$ where $b = 1$ if π is a valid opening to $\mathbf{r}_I$, and $b = 0$ otherwise.

Π_{HBG} is required to satisfy the following properties:

Correctness. For all $m, \lambda \in \mathbb{N}$ and all subsets $I \subseteq [m]$, we have

$$\Pr\left[\mathsf{Verify}(\mathsf{crs}, I, \mathbf{r}_I, \pi) = 1 \;\middle|\; \begin{array}{l} \mathsf{crs} \leftarrow \mathsf{Setup}(\lambda, m); \\ (\mathbf{r}, \mathsf{st}) \leftarrow \mathsf{GBits}(\mathsf{crs}); \\ \pi \leftarrow \mathsf{Prove}(\mathsf{st}, I) \end{array}\right] = 1.$$

Somewhat computational binding. For each $\mathsf{crs} \leftarrow \mathsf{Setup}(\lambda, m)$, there exists a sparse set $\mathcal{RV}^{\mathsf{crs}} \subset \{0,1\}^m$ where $|\mathcal{RV}^{\mathsf{crs}}| \leq 2^{m^{\gamma} \cdot \mathsf{poly}(\lambda)}$ for some constant $\gamma < 1$ and fixed polynomial $\mathsf{poly}(\cdot)$. Moreover, no efficient and stateful adversary $\mathcal{A}$ can generate an accepting proof π_I for $\mathbf{r}_I$ with $\mathbf{r}_I \notin \mathcal{RV}^{\mathsf{crs}}_I$, where $\mathcal{RV}^{\mathsf{crs}}_I = \{\mathbf{r}_I : \mathbf{r} \in \mathcal{RV}^{\mathsf{crs}}\}$. That is,

$$\Pr\left[\mathbf{r}_I \notin \mathcal{RV}^{\mathsf{crs}}_I \wedge \mathsf{Verify}(\mathsf{crs}, I, \mathbf{r}_I, \pi) = 1 \;\middle|\; \begin{array}{l} m \leftarrow \mathcal{A}(\lambda); \\ \mathsf{crs} \leftarrow \mathsf{Setup}(\lambda, m); \\ (I, \mathbf{r}_I, \pi) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right] = \varepsilon(\lambda).$$

Computationally hiding. For every polynomial $m = m(\lambda)$, $\mathsf{crs} \leftarrow \mathsf{Setup}(\lambda, m)$, $(\mathbf{r}, \mathsf{st}) \leftarrow \mathsf{GBits}(\mathsf{crs})$, $\pi \leftarrow \mathsf{Prove}(\mathsf{st}, I)$ every subset $I \subseteq [m]$, and all efficient adversary $\mathcal{A}$, we have

$$\left| \Pr\left[\mathcal{A}(\mathsf{crs}, \mathbf{r}_I, \pi, \mathbf{r}_{\bar{I}}) = 1\right] - \Pr\left[\mathcal{A}(\mathsf{crs}, \mathbf{r}_I, \pi, \{0,1\}^{|\bar{I}|}) = 1\right] \right| = \varepsilon(\lambda)$$

where $\bar{I} = [m] \backslash I$.

4 Efficient Pairing-Based Batch Arguments for NP

In this section, we show how to construct an efficient BARG with constant-size proof and constant number of pairing operations.

4.1 Assumption

The security of our BARG will rely on a variant of the subgroup decision exponent assumption [11]. We give the formal description below and prove this assumption in Appendix B.

Assumption 1 ((ℓ, t, s)-pairwise bivariate subgroup decision exponent) *Let $\ell, t, s \in \mathbb{N}$. We say that the (ℓ, t, s)-pairwise bivariate subgroup decision exponent assumption holds with respect to a composite-order bilinear group generator* CompGroupGen *if for all PPT adversaries $\mathcal{A}$, there exists a negligible function $\varepsilon(\cdot)$ such that*

$$\begin{aligned} &\left| \Pr\left[\mathcal{A}\left(\mathcal{G}_N, \{g_p^{\tau_X^a \tau_Y^b}\}_{(a,b)\in S_{XY}}, \{g_p^{\tau_X^a \tau_Z^c}\}_{(a,c)\in S_{XZ}}, \{g_p^{\tau_Y^b \tau_Z^c}\}_{(b,c)\in S_{YZ}}, g_p\right) = 1\right]\right. \\ &\left. - \Pr\left[\mathcal{A}\left(\mathcal{G}_N, \{g_p^{\tau_X^a \tau_Y^b}\}_{(a,b)\in S_{XY}}, \{g_p^{\tau_X^a \tau_Z^c}\}_{(a,c)\in S_{XZ}}, \{g_p^{\tau_Y^b \tau_Z^c}\}_{(b,c)\in S_{YZ}}, g_p g_q\right) = 1\right]\right| \\ &\leq \varepsilon(\lambda) \end{aligned}$$

where $\mathcal{G}_N = (\mathbb{G}, \mathbb{G}_T, N, g, e)$, $(\mathbb{G}, \mathbb{G}_T, p, q, g, e) \leftarrow \mathsf{CompGroupGen}(\lambda)$, $N = pq$, $r \leftarrow \mathbb{Z}_N$, $g_p = g^{qr}$, $g_q = g^{pr}$, $\tau_X, \tau_Y, \tau_Z \leftarrow \mathbb{Z}_N$, and

$$\begin{aligned} S_{XY} &= \{(a,b) : 0 \leq a \leq \ell,\ 0 \leq b \leq t,\ (a,b) \neq (0,0)\}, \\ S_{XZ} &= \{(a,c) : 0 \leq a \leq \ell,\ 0 \leq c \leq s,\ (a,c) \neq (0,0)\}, \\ S_{YZ} &= \{(b,c) : 0 \leq b \leq t,\ 0 \leq c \leq s,\ (b,c) \neq (0,0)\}. \end{aligned}$$

4.2 Protocol Description

We now construct our BARG for the language of circuit satisfiability from composite-order pairing groups.

– $\mathsf{Setup}(\lambda, \ell, t, s) \rightarrow \mathsf{crs}$:

1. Run $\mathcal{G} := (\mathbb{G}, \mathbb{G}_T, p, q, e, g) \leftarrow \mathsf{CompGroupGen}(\lambda)$. Let $N = pq$, sample randomly $r \leftarrow \mathbb{Z}_N$. Set $g_p = g^{qr}$, $\mathcal{G}_N = (\mathbb{G}, \mathbb{G}_T, N, g, e)$.
2. Sample $\tau_X, \tau_Y, \tau_Z \leftarrow \mathbb{Z}_N$. For each $i \in [0, \ell-1], j \in [0, t-1], k \in [0, s-1]$, let $A_{i,j} = g_p^{\tau_X^i \tau_Y^j}$, $B_{i,k} = g_p^{\tau_X^i \tau_Z^k}$ and $C_{j,k} = g_p^{\tau_Y^j \tau_Z^k}$. Here, $A_{0,0} = B_{0,0} = C_{0,0} = g_p$.
3. Denote three multiplicative subgroups $\mathbb{L} = \{\omega_L, \ldots, \omega_L^\ell\}, \mathbb{V} = \{\omega_V, \ldots, \omega_V^t\}, \mathbb{H} = \{\omega_H, \ldots, \omega_H^s\}$ such that $|\mathbb{L}| = \ell, |\mathbb{V}| = t, |\mathbb{H}| = s$. Define the vanishing polynomials $Z_n(Y) = \prod_{j\in[n]}(Y - \omega_V^j), Z_\ell(X) = \prod_{i\in[\ell]}(X - \omega_L^i), Z_t(Y) = \prod_{j\in[t]}(Y - \omega_V^j), Z_s(Z) = \prod_{k\in[s]}(Z - \omega_H^k)$. Compute $Z_n = g_p^{Z_n(\tau_Y)}, Z_\ell = g_p^{Z_\ell(\tau_X)}, Z_t = g_p^{Z_t(\tau_Y)}$ and $Z_s = g_p^{Z_s(\tau_Z)}$. Particularly, set $Z_{t,t}(Y) = Y - \omega_V^t$ and set $Z_{t,t} = g_p^{Z_{t,t}(\tau_Y)}$.
4. Output $\mathsf{crs} = (\mathcal{G}_N, \{A_{i,j}, B_{i,k}, C_{j,k}\}_{i\in[0,\ell-1], j\in[0,t-1], k\in[0,s-1]}, Z_n, Z_\ell, Z_t, Z_s, Z_{t,t})$.

– $\mathsf{Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_\ell), (\mathbf{w}_1, \ldots, \mathbf{w}_\ell)) \to \pi$:

1. **Interpolate wire values.** For each $i \in [\ell], j \in [t], k \in [s]$, define the Lagrange basis polynomials $L_i(X) = \frac{\omega_L^i}{|\mathbb{L}|} \cdot \frac{X^{|\mathbb{L}|}-1}{X-\omega_L^i}, \bar{L}_j(Y) = \frac{\omega_V^j}{|\mathbb{V}|} \cdot \frac{Y^{|\mathbb{V}|}-1}{Y-\omega_V^j}, \tilde{L}_k(Z) = \frac{\omega_H^k}{|\mathbb{H}|} \cdot \frac{Z^{|\mathbb{H}|}-1}{Z-\omega_H^k}$. Let $v_{i,j}$ denote the value of j-th wire in $C(\mathbf{x}_i, \mathbf{w}_i)$. Let $\Phi(X, Y) = \sum_{j\in[t]} v_{i,j} L_i(X) \bar{L}_j(Y)$ be the interpolation polynomials of degrees $\deg_X < \ell$ and $\deg_Y < t$ that evaluates over $\mathbb{L} \times \mathbb{V}$.
   Particularly, define the public-input interpolation polynomial
   $$\Phi_{\mathsf{in}}(X, Y) = \sum_{i\in[\ell]} \sum_{j\in[n]} x_{i,j} L_i(X) \bar{L}_j^{(n)}(Y).$$
   $\{\bar{L}_j^{(n)}(Y)\}_{j\in[n]}$ is the Lagrange basis over $V_n = \{\omega_V, \ldots, \omega_V^n\}$. Thus, for all $i \in [\ell], j \in [n]$, $\Phi_{\mathsf{in}}(\omega_L^i, \omega_V^j) = x_{i,j}$.
   For each NAND gate $G_k = (a_k, b_k, c_k)$ where $k \in [s]$, let $\Phi_a(X, Z), \Phi_b(X, Z), \Phi_c(X, Z)$ be interpolation polynomials of degree $\deg_X < \ell$ and $\deg_Z < s$ over $\mathbb{L} \times \mathbb{H}$ respectively, where the values $\Phi_a(\omega_L^i, \omega_H^k) = v_{i,a_k}, \Phi_b(\omega_L^i, \omega_H^k) = v_{i,b_k}$, and $\Phi_c(\omega_L^i, \omega_H^k) = v_{i,c_k}$ represent the values of the left input, right input, and output wires of the k-th gate in $C(\mathbf{x}_i, \mathbf{w}_i)$, respectively.
2. **Encoding.** Rewrite $\Phi(X, Y) = \sum_{i\in[0,\ell-1]} \sum_{j\in[0,t-1]} \phi_{i,j} X^i Y^j$ where $\phi_{i,j}$ is the coefficient of $\Phi(X, Y)$. Similarly, $\Phi_{\mathsf{in}}(X, Y) = \sum_{i\in[0,\ell-1]} \sum_{j\in[0,n-1]} \phi_{\mathsf{in},i,j} X^i Y^j, \Phi_a(X, Z) = \sum_{i\in[0,\ell-1]} \sum_{k\in[0,s-1]} \phi_{a,i,k} X^i Z^k, \Phi_b(X, Z) = \sum_{i\in[0,\ell-1]} \sum_{k\in[0,s-1]} \phi_{b,i,k} X^i Z^k, \Phi_c(X, Z) = \sum_{i\in[0,\ell-1]} \sum_{k\in[0,s-1]} \phi_{c,i,k} X^i Z^k$. Set $\Phi_{\mathsf{in}} = \prod_{i\in[0,\ell-1], j\in[0,n-1]} A_{i,j}^{\phi_{\mathsf{in},i,j}}, \Phi = \prod_{i\in[0,\ell-1], j\in[0,t-1]} A_{i,j}^{\phi_{i,j}}, \Phi_a = \prod_{i\in[0,\ell-1], k\in[0,s-1]} B_{i,k}^{\phi_{a,i,k}}$, $\Phi_b = \prod_{i\in[0,\ell-1], k\in[0,s-1} B_{i,k}^{\phi_{b,i,k}}, \Phi_c = \prod_{i\in[0,\ell-1], k\in[0,s-1]} B_{i,k}^{\phi_{c,i,k}}$.
3. **Validity of wire assignments.** Define the polynomial
   $$Q_\ell(X, Y) \cdot Z_\ell(X) + Q_t(X, Y) \cdot Z_t(Y) = \Phi(X, Y)^2 - \Phi(X, Y)$$
   where $Q_\ell(X, Y) = \sum_{i\in[0,\ell-1]} \sum_{j\in[0,t-1]} q_{\ell,i,j} X^i Y^j$ and $Q_t(X, Y) = \sum_{i\in[0,\ell-1]} \sum_{j\in[0,t-1]} q_{t,i,j} X^i Y^j$. Compute $Q_\ell = \prod_{i\in[0,\ell-1], j\in[0,t-1]} A_{i,j}^{q_{\ell,i,j}}$ and $Q_t = \prod_{i\in[0,\ell-1], j\in[0,t-1]} A_{i,j}^{q_{t,i,j}}$.

4. **Validity of public-input consistency.** Define the polynomial

$$\Phi(X,Y) - \Phi_{\mathsf{in}}(X,Y) = Q_\ell^{\mathsf{in}}(X,Y)Z_\ell(X) + Q_n^{\mathsf{in}}(X,Y)Z_n(Y),$$

where $Q_\ell^{\mathsf{in}}(X,Y) = \sum_{i\in[0,\ell-1]}\sum_{j\in[0,n-1]} q_{\ell,i,j}^{\mathsf{in}} X^i Y^j$ and $Q_n^{\mathsf{in}}(X,Y) = \sum_{i\in[0,\ell-1]}\sum_{j\in[0,n-1]} q_{n,i,j}^{\mathsf{in}} X^i Y^j$.

Compute The proof contains commitments to $Q_\ell^{\mathsf{in}} = \prod_{i\in[0,\ell-1],j\in[0,n-1]} A_{i,j}^{q_{\ell,i,j}^{\mathsf{in}}}$, $Q_n^{\mathsf{in}} = \prod_{i\in[0,\ell-1],j\in[0,n-1]} A_{i,j}^{q_{n,i,j}^{\mathsf{in}}}$

5. **Validity of gate computation.** Define the polynomial identity

$$Q'_\ell(X,Z)\cdot Z_\ell(X) + Q'_s(X,Z)\cdot Z_s(Z) = 1 - \Phi_c(X,Z) - \Phi_a(X,Z)\Phi_b(X,Z)$$

where $Q'_\ell(X,Z) = \sum_{i\in[0,\ell-1]}\sum_{k\in[0,s-1]} q'_{\ell,i,k} X^i Z^k$ and $Q'_s(X,Z) = \sum_{i\in[0,\ell-1]}\sum_{k\in[0,s-1]} q'_{s,i,k} X^i Z^k$.

Compute $Q'_\ell = \prod_{i\in[0,\ell-1],k\in[0,s-1]} B_{i,k}^{q'_{\ell,i,k}}$ and $Q'_s = \prod_{i\in[0,\ell-1],k\in[0,s-1]} B_{i,k}^{q'_{s,i,k}}$.

6. **Validity of gate-wire consistency.** Rewrite the definition in a sum-check fashion

$$\begin{aligned}
\Phi_a(X,Z) &= \sum_{y\in\mathbb{H}} \Phi(X,y)\cdot \mathsf{eq}_a(Z,y),\\
\Phi_b(X,Z) &= \sum_{y\in\mathbb{H}} \Phi(X,y)\cdot \mathsf{eq}_b(Z,y),\\
\Phi_c(X,Z) &= \sum_{y\in\mathbb{H}} \Phi(X,y)\cdot \mathsf{eq}_c(Z,y)
\end{aligned}$$

where $\mathsf{eq}_a, \mathsf{eq}_b, \mathsf{eq}_c$ are defined as follows:

$$\mathsf{eq}_a(u,v) = \begin{cases} 1, & A(u) = v \\ 0, & A(u) \neq v \end{cases},\ \mathsf{eq}_a(u,v) = \begin{cases} 1, & B(u) = v \\ 0, & B(u) \neq v \end{cases},$$

$$\mathsf{eq}_c(u,v) = \begin{cases} 1, & C(u) = v \\ 0, & C(u) \neq v \end{cases}.$$

Here $\mathsf{eq}_a, \mathsf{eq}_b, \mathsf{eq}_c$ depend on C. The maps $A(Z), B(Z), C(Z) : [s] \to [t]$ satisfy

$$A(k) = a_k, \quad B(k) = b_k, \quad C(k) = c_k \quad \forall k \in [s].$$

Define the polynomial identities

$$\begin{aligned}
\Phi(X,Y)\cdot \mathsf{eq}_a(Z,Y) - \Phi_a(X,Z)\cdot t^{-1} &= H_a(Y)\cdot Z_t(Y) + Y\cdot F_a(Y),\\
\Phi(X,Y)\cdot \mathsf{eq}_b(Z,Y) - \Phi_b(X,Z)\cdot t^{-1} &= H_b(Y)\cdot Z_t(Y) + Y\cdot F_b(Y),\\
\Phi(X,Y)\cdot \mathsf{eq}_c(Z,Y) - \Phi_c(X,Z)\cdot t^{-1} &= H_c(Y)\cdot Z_t(Y) + Y\cdot F_c(Y)
\end{aligned}$$

where $H_a(Y) = \sum_{j\in[0,t-1]} h_{a,j} Y^j, F_a(Y) = \sum_{j\in[0,t-1]} f_{a,j} Y^j, H_b(Y) = \sum_{j\in[0,t-1]} h_{b,j} Y^j, F_b(Y) = \sum_{j\in[0,t-1]} f_{b,j} Y^j$ and $H_c(Y) = \sum_{j\in[0,t-1]} h_{c,j} Y^j$, $F_c(Y) = \sum_{j\in[0,t-1]} f_{c,j} Y^j$. Compute $H_a = \prod_{j\in[0,t-1]} A_{0,j}^{h_{a,j}}$, $F_a = \prod_{j\in[0,t-1]} A_{0,j}^{f_{a,j}}$, $H_b = \prod_{j\in[0,t-1]} A_{0,j}^{h_{b,j}}$, $F_b = \prod_{j\in[0,t-1]} A_{0,j}^{f_{b,j}}$ and $H_c = \prod_{j\in[0,t-1]} A_{0,j}^{h_{c,j}}$, $F_c = \prod_{j\in[0,t-1]} A_{0,j}^{f_{c,j}}$.

7. **Output satisfiability.** We prove that the output is a commitment to the all-ones vector by showing $\Phi(X, \omega_V^t) = \Phi_t(X) \equiv 1$. The constant function $\Phi_t(X)$ interpolates the vector of all-ones for the t-th wire across all ℓ instances. Equivalently, there exists a polynomial $q_t(X, Y)$ such that
$$\Phi(X, Y) - 1 = \hat{Q}_t(X, Y)\,(Y - \omega_V^t)$$
where $\hat{Q}_t(X, Y) = \sum_{i\in[0,\ell-1]} \sum_{j\in[0,t-1]} \hat{q}_{t,i,j} X^i Y^j$. Compute $\hat{Q}_t = \prod_{i\in[0,\ell-1], j\in[0,t-1]} A_{i,j}^{\hat{q}_{t,i,j}}$.
8. Output proof $\pi = (\Phi, \Phi_a, \Phi_b, \Phi_c, Q_\ell^{\mathsf{in}}, Q_n^{\mathsf{in}}, Q_\ell, Q_t, Q'_\ell, Q'_s, H_a, H_b, H_c, F_a, F_b, F_c, \hat{Q}_t)$.

– Verify$(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_\ell), \pi) \to \{0, 1\}$: This algorithm checks the following properties:

- **Validity of public-input consistency.** For $i \in [\ell]$, $j \in [n]$, $\phi(i, j) = x_{i,j}$. Recompute Φ_{in} using Lagrange interpolation modulo N and check that
$$e(\Phi, A_{0,0}) = e(\Phi_{\mathsf{in}}, A_{0,0}) \cdot e(Q_\ell^{\mathsf{in}}, Z_\ell) \cdot e(Q_n^{\mathsf{in}}, Z_n).$$
- **Validity of wire assignments.** Check that
$$e(\Phi, \Phi) = e(Q_\ell, Z_\ell) \cdot e(Q_t, Z_t) \cdot e(\Phi, A_{0,0}).$$
- **Validity of gate computation.** Check that
$$e(Q'_\ell, Z_\ell) \cdot e(Q'_s, Z_s) \cdot e(\Phi_c, B_{0,0}) \cdot e(\Phi_a, \Phi_b) = e(B_{0,0}, B_{0,0}).$$
- **Validity of gate permutation.** Compute $\mathsf{eq}_a, \mathsf{eq}_b, \mathsf{eq}_c$. Check that
$$\begin{aligned} e(\Phi, \mathsf{eq}_a) \cdot e(\Phi_a, B_{0,0}^{t^{-1}}) &= e(H_a, Z_t) \cdot e(A_{0,1}, F_a), \\ e(\Phi, \mathsf{eq}_b) \cdot e(\Phi_b, B_{0,0}^{t^{-1}}) &= e(H_b, Z_t) \cdot e(A_{0,1}, F_b), \\ e(\Phi, \mathsf{eq}_c) \cdot e(\Phi_c, B_{0,0}^{t^{-1}}) &= e(H_c, Z_t) \cdot e(A_{0,1}, F_c) \end{aligned}$$
- **Output satisfiability.** Check that
$$e(\Phi, A_{0,0}) = e(A_{0,0}, A_{0,0}) \cdot e(\hat{Q}_t, Z_{t,t}).$$
- Output 1 if all checks pass and 0 otherwise.

4.3 Analysis.

We analyze the completeness of and security of Construction 4.2 using the following theorems. More details can be found in Appendix D.

Theorem 4 (Completeness). *For any Boolean circuit C and any set of ℓ instance-witness pairs $\{(\mathbf{x}_i, \mathbf{w}_i)\}_{i=1}^{\ell}$ such that $C(\mathbf{x}_i, \mathbf{w}_i) = 1$ for all $i \in [\ell]$, if the prover follows the introduced protocol* BARG *correctly, then:*
$$\Pr\left[\mathsf{Verify}\left(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_\ell), \mathsf{Prove}(\mathsf{crs}, C, \mathbf{x}_{1:\ell}, \mathbf{w}_{1:\ell})\right) = 1\right] = 1.$$

Theorem 5 (Weak Somewhere Extractability). *Assume that the (ℓ, t, s)-pairwise bivariate subgroup decision exponent assumption holds over composite-order pairing groups. Then Construction 4.2 satisfies weak somewhere extractability.*

Corollary 1 (Somewhere soundness). *Assume that the (ℓ, t, s)-pairwise bivariate subgroup decision exponent assumption holds. Then Construction 4.2 satisfies somewhere soundness.*

Proof. This follows immediately from Theorem 5 and Lemma 1, since Construction 4.2 enforces public-input consistency, gate validity, gate–wire consistency, and output-wire consistency.

Lemma 2 (CRS Indistinguishability). *Assume that the (ℓ, t, s)-pairwise bivariate subgroup decision exponent assumption holds over composite-order pairing groups. Then the normal CRS and the trapdoor CRS generated for any target tuple (i^*, j^*, k^*) are computationally indistinguishable.*

Lemma 3 (Weak Extractability in Trapdoor Mode). *Construction 4.2 is statistically somewhere extractable in trapdoor mode.*

In the applications below, we use the somewhere-soundness property of Construction 4.2, rather than its extraction algorithm. The extraction algorithm is used only in the proof of Theorem 5 and in the derivation of Corollary 1.

5 Applications

5.1 Hidden-Bits Generator from BARG

In this subsection, we show how to construct a hidden-bits generator with subset-dependent proofs from our construction 4.2, together with a one-time dual mode commitment with extraction, and a leakage-resilient weak PRF. The construction description is provided bellow:

Protocol Description Let $\lambda \in \mathbb{N}$ be a security parameter and $m \in \mathbb{N}$ be an output length parameter. $|\mathsf{rk}| = \kappa$ be the size of the PRF key rk.

- $\mathsf{Setup}(\lambda, m, \kappa)$:
  1. Run $(\mathsf{crs}_{\mathsf{BC}}^{(j)}, \mathsf{td}_{\mathsf{BC}}^{(j)}) \leftarrow \mathsf{BC.SetupBind}(\lambda)$ for each $j \in [\kappa]$.
  2. Let $C_{\mathcal{R}}$ be be the circuit that computes the NP relation $\mathcal{R}$. Run $\mathsf{crs}_{\mathsf{BARG}} \leftarrow \mathsf{BARG.Setup}(\lambda, m, t, s)$.
  3. Sample $\mathbf{z}_1, \ldots, \mathbf{z}_m \leftarrow \{0,1\}^d$.
  4. Output $\mathsf{crs} = (\{\mathsf{crs}_{\mathsf{BC}}^{(j)}\}_{j\in[\kappa]}, \mathsf{crs}_{\mathsf{BARG}}, \mathbf{z}_1, \ldots, \mathbf{z}_m)$.
- $\mathsf{GBits}(\mathsf{crs})$:
  1. Sample a PRF key $\mathsf{rk} \leftarrow \mathsf{LRwPRF.Setup}(\lambda, \mathsf{leak})$.
  2. Compute $r_i \leftarrow \mathsf{LRwPRF.Eval}(\mathsf{rk}, \mathbf{z}_i)$ for each $i \in [m]$.
  3. Output $\mathbf{r} = r_1 || \cdots || r_m$ and $\mathsf{st} = (\mathsf{crs}, \mathsf{rk})$.

– $\mathsf{Prove}(I, \mathsf{st})$:

1. Commit to each PRF key bit $\mathsf{com}_{\mathsf{rk}_j} \leftarrow \mathsf{BC.Commit}(\mathsf{crs}_{\mathsf{BC}}^{(j)}, \mathsf{rk}_j)$ for each $j \in [\kappa]$.
2. Construct a BARG proof validity. Define the relation $\mathcal{R}$

$$\begin{aligned} \text{statement:} \quad & \mathbf{x}_i = \left(C_{\mathsf{LRwPRF}}(\cdot), \mathbf{z}_i, \{\mathsf{com}_{\mathsf{rk}_j}\}_{j\in[\kappa]},\ r_i\right) \\ \text{witness:} \quad & \mathbf{w}_i = \left(\{\mathsf{rk}_j,\ \sigma_j\}_{j\in[\kappa]}\right) \end{aligned}$$

   Run $\pi_{\mathsf{BARG}} \leftarrow \mathsf{BARG.Prove}(\mathsf{crs}_{\mathsf{BARG}}, C_{\mathcal{R}}, \{\mathbf{x}_i, \mathbf{w}_i\}_{i\in I})$.
3. Output $\pi_{\mathsf{HBG}} = (\pi_{\mathsf{BARG}}, \{\mathsf{com}_{\mathsf{rk}_j}\}_{j\in[\kappa]})$.

– $\mathsf{Verify}(\mathsf{crs}, I, \mathbf{r}_I, \pi_{\mathsf{HBG}})$:

1. For each $i \in I$, construct statement $\mathbf{x}_i$ from crs and π_{HBG}.
2. Output $\mathsf{BARG.Verify}(\mathsf{crs}_{\mathsf{BARG}}, C_{\mathcal{R}}, (\mathbf{x}_1, \ldots, \mathbf{x}_{|I|}), \pi_{\mathsf{BARG}})$.

Analysis Throughout our analysis, we use the following theorems which we prove in Appendix E.

Theorem 6 (Correctness). *If Π_{BARG} is complete and Π_{BC} is correct, then construction 5.1 is correct.*

By Corollary 1, Construction 4.2 satisfies somewhere soundness under the (ℓ, t, s)-pairwise bivariate subgroup decision exponent assumption. Therefore, the binding analysis below uses the somewhere-soundness property of the underlying BARG directly.

Theorem 7 (Somewhat computational binding). *Assume that:*

1. *Π_{BC} is statistically binding and extractable in the binding mode;*
2. *Π_{BARG} satisfies somewhere soundness;*
3. *$\kappa \le m^{\gamma} \cdot \mathsf{poly}(\lambda)$ for some constant $\gamma < 1$ and fixed polynomial $\mathsf{poly}(\cdot)$.*

Then Construction 5.1 is somewhat computationally binding.

Theorem 8 (Computational Hiding). *Assume that Π_{BC} satisfies mode indistinguishability and LRwPRF is a secure leakage-resilient weak PRF. If $\mathsf{leak}(\lambda, m) \ge |\pi_{\mathsf{BARG}}(\lambda, m)|$, construction 5.1 satisfies computational hiding.*

Since our BARG proof consists of a constant number of group elements, its bit-length is $O(\lambda)$ and is independent of the circuit size. Hence the hiding proof only requires the underlying weak PRF to tolerate $O(\lambda)$ bits of leakage.

5.2 Rate-1 Somewhere Extractable BARG

Instantiating the rate-1 flSEH with the construction from [8] and the BARG with one from Construction 4.2, we obtain a rate-1 BARG for NP relation with witnesses of h from pairing under a new q-type assumption. The construction description is provided bellow:

- $\mathsf{Setup}(\lambda, \ell, h, t_{\mathsf{loc}}, s_{\mathsf{loc}})$:
  1. Run $(\mathsf{hk}_{\mathsf{flSEH}}, \mathsf{vk}_{\mathsf{flSEH}}) \leftarrow \mathsf{flSEH.Gen}(\lambda, \ell \cdot h, h)$. Here, the last parameter h denotes the size of the binding set.
  2. Run $\mathsf{crs}_{\mathsf{BARG}} \leftarrow \mathsf{BARG.Setup}(\lambda, \ell, t_{\mathsf{loc}}, s_{\mathsf{loc}})$. Here, t_{loc} and s_{loc} are upper bounds on the number of wires and gates of the local-checking circuit C_{loc}.
  3. Output $\mathsf{crs} = (\mathsf{hk}_{\mathsf{flSEH}}, \mathsf{vk}_{\mathsf{flSEH}}, \mathsf{crs}_{\mathsf{BARG}})$.
- $\mathsf{TrapSetup}(\lambda, \ell, h, t_{\mathsf{loc}}, s_{\mathsf{loc}}, i^*)$:
  1. Define the hidden binding set $I_{i^*} = \{(i^* - 1) \cdot h + 1, \ldots, i^* \cdot h\}$.
  2. Run $\mathsf{crs}_{\mathsf{BARG}} \leftarrow \mathsf{BARG.Setup}(\lambda, \ell, t_{\mathsf{loc}}, s_{\mathsf{loc}})$.
  3. Run $(\mathsf{hk}_{\mathsf{flSEH}}, \mathsf{vk}_{\mathsf{flSEH}}, \mathsf{td}_{\mathsf{flSEH}}) \leftarrow \mathsf{flSEH.TrapGen}(\lambda, \ell \cdot h, I_{i^*})$.
  4. Output $\mathsf{crs} = (\mathsf{hk}_{\mathsf{flSEH}}, \mathsf{vk}_{\mathsf{flSEH}}, \mathsf{crs}_{\mathsf{BARG}})$ and $\mathsf{td} = \mathsf{td}_{\mathsf{flSEH}}$.
- $\mathsf{Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_\ell), (\mathbf{w}_1, \ldots, \mathbf{w}_\ell))$:
  1. Let $W = (\mathbf{w}_1, \ldots, \mathbf{w}_\ell) \in \{0,1\}^{\ell \cdot h}$. Compute
  $$(\mathsf{hv}, \mathsf{rt}_w) \leftarrow \mathsf{flSEH.Hash}(\mathsf{hk}_{\mathsf{flSEH}}, W).$$
  2. For every $i \in [\ell]$ and every $j \in [h]$, compute the local opening
  $$(w_{i,j}, \rho_{i,j}) \leftarrow \mathsf{flSEH.Open}(\mathsf{hk}_{\mathsf{flSEH}}, W, (i-1) \cdot h + j).$$
  3. Let C_{loc} be the local-checking circuit. For every $i \in [\ell]$, define
  $$\mathbf{x}'_i = (C, \mathbf{x}_i, \mathsf{vk}_{\mathsf{flSEH}}, \mathsf{hv}, \mathsf{rt}_w, i),$$
  $$\mathbf{w}'_i = (\mathbf{w}_i, \{\rho_{i,j}\}_{j \in [h]}),$$
  4. Compute
  $$\pi_{\mathsf{BARG}} \leftarrow \mathsf{BARG.Prove}(\mathsf{crs}_{\mathsf{BARG}}, C_{\mathsf{loc}}, \{(\mathbf{x}'_i, \mathbf{w}'_i)\}_{i \in [\ell]}).$$
  5. Output $\pi = (\mathsf{hv}, \mathsf{rt}_w, \pi_{\mathsf{BARG}})$.
- $\mathsf{Verify}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_\ell), \pi)$: Output 1 if and only if
  $$\mathsf{flSEH.Validate}(\mathsf{vk}_{\mathsf{flSEH}}, \mathsf{hv}, \mathsf{rt}_w) = 1$$
  and
  $$\mathsf{BARG.Verify}(\mathsf{crs}_{\mathsf{BARG}}, C_{\mathsf{loc}}, \{\mathbf{x}'_i\}_{i \in [\ell]}, \pi_{\mathsf{BARG}}) = 1.$$

Efficiency. Since the underlying flSEH is rate-1, the main hash value has size $|\mathsf{hv}| = h + \mathsf{poly}(\lambda)$. The auxiliary digest rt_w has size $\mathsf{poly}(\lambda, \log \ell)$. Our weak seBARG proof size is constant. Therefore, $|\pi| = |\mathsf{hv}| + |\mathsf{rt}_w| + |\pi_{\mathsf{w}}| = h + \mathsf{poly}(\lambda, \log \ell)$. Thus the construction is rate-1 with respect to the witness size.

Analysis. Throughout our analysis, we use the following theorems which we prove in Appendix F.

Theorem 9 (Completeness). *If* Π_{BARG} *is complete and* Π_{flSEH} *is complete, then the construction 5.2 is complete.*

Theorem 10 (Somewhere extractability). *Assume that* Π_{BARG} *satisfies somewhere soundness and* Π_{flSEH} *has rate-1 somewhere extractability. The construction 5.2 has somewhere extractability.*

References

1. Gaspard Anthoine, David Balbás, and Dario Fiore. Fully-succinct multi-key homomorphic signatures from standard assumptions. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part III*, volume 14922 of *LNCS*, pages 317–351. Springer, Cham, August 2024. 2
2. Gal Arnon, Alessandro Chiesa, and Eylon Yogev. IOPs with inverse polynomial soundness error. In *64th FOCS*, pages 752–761. IEEE Computer Society Press, November 2023. 28
3. David Balbás, Dario Fiore, and Russell WF Lai. Circuit-succinct algebraic batch arguments from projective functional commitments. *Cryptology ePrint Archive*, 2025. 2, 4
4. Eli Bradley, Brent Waters, and David J. Wu. Batch arguments to NIZKs from one-way functions. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024, Part II*, volume 15365 of *LNCS*, pages 431–463. Springer, Cham, December 2024. 2, 3, 10, 11, 12
5. Zvika Brakerski, Maya Farber Brodsky, Yael Tauman Kalai, Alex Lombardi, and Omer Paneth. SNARGs for monotone policy batch NP. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part II*, volume 14082 of *LNCS*, pages 252–283. Springer, Cham, August 2023. 2
6. Zvika Brakerski, Justin Holmgren, and Yael Tauman Kalai. Non-interactive delegation and batch NP verification from standard computational assumptions. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *49th ACM STOC*, pages 474–482. ACM Press, June 2017. 1
7. Pedro Branco, Nico Döttling, and Akshayaram Srinivasan. Rate-1 statistical non-interactive zero-knowledge. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part VII*, volume 16006 of *LNCS*, pages 514–547. Springer, Cham, August 2025. 2
8. Pedro Branco, Nico Döttling, Akshayaram Srinivasan, and Riccardo Zanotto. Rate-1 fully local somewhere extractable hashing from DDH. In Qiang Tang and Vanessa Teague, editors, *PKC 2024, Part II*, volume 14603 of *LNCS*, pages 356–386. Springer, Cham, April 2024. 3, 4, 12, 13, 24
9. Maya Farber Brodsky, Arka Rai Choudhuri, Abhishek Jain, and Omer Paneth. Monotone-policy aggregate signatures. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part IV*, volume 14654 of *LNCS*, pages 168–195. Springer, Cham, May 2024. 2
10. Jeffrey Champion and David J. Wu. Non-interactive zero-knowledge from non-interactive batch arguments. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part II*, volume 14082 of *LNCS*, pages 38–71. Springer, Cham, August 2023. 2, 3, 10, 11, 12
11. Binyi Chen, Noel Elias, and David J Wu. Pairing-based batch arguments for np with a linear-size crs. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 554–577. Springer, 2025. 2, 5, 18

12. Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. Non-interactive batch arguments for NP from standard assumptions. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 394–423, Virtual Event, August 2021. Springer, Cham. 2, 14
13. Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. SNARGs for $\mathcal{P}$ from LWE. In *62nd FOCS*, pages 68–79. IEEE Computer Society Press, February 2022. 2
14. Pratish Datta, Abhishek Jain, Zhengzhong Jin, Alexis Korb, Surya Mathialagan, and Amit Sahai. Incrementally verifiable computation for NP from standard assumptions. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part VII*, volume 16006 of *LNCS*, pages 611–642. Springer, Cham, August 2025. 2
15. Lalita Devadas, Rishab Goyal, Yael Kalai, and Vinod Vaikuntanathan. Rate-1 non-interactive arguments for batch-NP and applications. In *63rd FOCS*, pages 1057–1068. IEEE Computer Society Press, October / November 2022. 2, 3, 4, 12
16. Lalita Devadas, Brent Waters, and David J. Wu. Batching adaptively-sound SNARGs for NP. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024, Part II*, volume 15365 of *LNCS*, pages 339–370. Springer, Cham, December 2024. 1
17. Chaya Ganesh, Sikhar Patranabis, Shubh Prakash, and Nitin Singh. GAPP: Generic aggregation of polynomial protocols. Cryptology ePrint Archive, Report 2024/1685, 2024. 6
18. Rachit Garg, Kristin Sheridan, Brent Waters, and David J. Wu. Fully succinct batch arguments for np from indistinguishability obfuscation. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 526–555. Springer, Cham, November 2022. 1
19. Sanjam Garg, Dimitris Kolonelos, Guru-Vamsi Policharla, and Mingyuan Wang. Threshold encryption with silent setup. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part VII*, volume 14926 of *LNCS*, pages 352–386. Springer, Cham, August 2024. 28
20. Alexander Golovnev, Jonathan Lee, Srinath T. V. Setty, Justin Thaler, and Riad S. Wahby. Brakedown: Linear-time and field-agnostic SNARKs for R1CS. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part II*, volume 14082 of *LNCS*, pages 193–226. Springer, Cham, August 2023. 29
21. Rishab Goyal. Mutable batch arguments and applications. *Cryptology ePrint Archive*, 2024. 2
22. Zhengzhong Jin, Yael Tauman Kalai, Alex Lombardi, and Surya Mathialagan. Universal SNARGs for NP from proofs of correctness. In Michal Koucký and Nikhil Bansal, editors, *57th ACM STOC*, pages 933–943. ACM Press, June 2025. 2
23. Yael Kalai, Alex Lombardi, Vinod Vaikuntanathan, and Daniel Wichs. Boosting batch arguments and RAM delegation. In Barna Saha and Rocco A. Servedio, editors, *55th ACM STOC*, pages 1545–1552. ACM Press, June 2023. 2, 13
24. Yael Tauman Kalai, Omer Paneth, and Lisa Yang. How to delegate computations publicly. In Moses Charikar and Edith Cohen, editors, *51st ACM STOC*, pages 1115–1124. ACM Press, June 2019. 1, 2
25. Yael Tauman Kalai, Vinod Vaikuntanathan, and Rachel Yun Zhang. Somewhere statistical soundness, post-quantum security, and SNARGs. In Kobbi Nissim and Brent Waters, editors, *TCC 2021, Part I*, volume 13042 of *LNCS*, pages 330–368. Springer, Cham, November 2021. 2
26. Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 177–194. Springer, Berlin, Heidelberg, December 2010. 5
27. Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 146–162. Springer, Berlin, Heidelberg, April 2008. 33
28. Weihan Li, Zongyang Zhang, Yun Li, Pengfei Zhu, Cheng Hong, and Jianwei Liu. Soloist: Distributed snark for r1cs with constant proof size. 2026. 4, 6
29. Shafik Nassar, Brent Waters, and David J. Wu. Monotone policy BARGs from BARGs and additively homomorphic encryption. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024, Part II*, volume 15365 of *LNCS*, pages 399–430. Springer, Cham, December 2024. 2
30. Shafik Nassar, Brent Waters, and David J. Wu. Monotone-policy BARGs and more from BARGs and quadratic residuosity. In Tibor Jager and Jiaxin Pan, editors, *PKC 2025, Part IV*, volume 15677 of *LNCS*, pages 283–313. Springer, Cham, May 2025. 2
31. Omer Paneth and Rafael Pass. Incrementally verifiable computation via rate-1 batch arguments. In *63rd FOCS*, pages 1045–1056. IEEE Computer Society Press, October / November 2022. 2, 4

32. Brent Waters and David J. Wu. Batch arguments for NP and more from standard bilinear group assumptions. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 433–463. Springer, Cham, August 2022. 2, 5, 14
33. Mark Zhandry. To label, or not to label (in generic groups). In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part III*, volume 13509 of *LNCS*, pages 66–96. Springer, Cham, August 2022. 33

Appendix

A Additional Preliminaries

A.1 Generalized Sumcheck

Polynomials and Lagrange Basis. Let $\mathbb{F}$ be a field and $\ell \in \mathbb{N}$. Let ω_L be a primitive ℓ-th root of unity that generates multiplicative subgroup $\mathbb{L} = \{\omega_L^1, \omega_L^2, \ldots, \omega_L^\ell\}$. We define the set of *Lagrange basis polynomials* $\{L_i(X)\}_{i\in[\ell]}$, where each $L_i(X)$ is the unique polynomial satisfying

$$\deg(L_i) = \ell - 1 \quad \text{and} \quad L_i(\omega_L^j) = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}.$$

Let $Z_{\mathbb{L}}(X) = \prod_{i=1}^{\ell}(X - \omega_L^i)$ be the *vanishing polynomial* on $\mathbb{L}$. We have

$$Z_{\mathbb{L}}(X) = X^\ell - 1, \quad L_i(X) = \frac{\omega_L^i}{\ell} \cdot \frac{X^\ell - 1}{X - \omega_L^i}, \quad L_i(0) = \frac{1}{\ell}.$$

We use bivariate polynomials to encode matrices in our scheme. Given a matrix $\mathbf{V} \in \mathbb{Z}_p^{\ell\times t}$, denote by $\mathbf{v}_j = (\mathbf{V}[1, j], \ldots, \mathbf{V}[\ell, j])$ its j-th column. We encode each column vector $\mathbf{v}_j$ to a univariate polynomial

$$v_j(X) = \sum_{i\in[\ell]} \mathbf{V}[i, j]L_i(X).$$

We then aggregate all column polynomials into a bivariate polynomial

$$V(X, Y) = \sum_{j\in[t]} v_j(X)\bar{L}_j(Y)$$

where $\{\bar{L}_j(Y)\}_{j\in[t]}$ are the Lagrange basis polynomials defined over the multiplicative subgroup $\mathbb{V} = \{\omega_V^1, \ldots, \omega_V^t\}$ generated by a primitive t-th root of unity ω_V. Each $\bar{L}_j(Y)$ has the explicit form

$$\bar{L}_j(Y) = \frac{\omega_V^j}{\ell} \cdot \frac{Y^t - 1}{Y - \omega_V^j}.$$

Lemma 4 (Weighted Univariate Sumcheck[19,2]). *Denote a multiplicative subgroup* $\mathbb{L} = \{\omega_L^1, \omega_L^2, \ldots, \omega_L^\ell\}$ *with* $|\mathbb{L}| = \ell$. *Let* $L_i(X) = \frac{\omega_L^i}{|\mathbb{L}|} \cdot \frac{X^{|\mathbb{L}|}-1}{X-\omega_L^i}$, $Z_{\mathbb{L}}(X) = \prod_{i=1}^{|\mathbb{L}|}(X - \omega_L^i)$, $v(X) = \sum_{i=1}^{|\mathbb{L}|} v_i L_i(X)$ *and* $r(X) = \sum_{i=1}^{|\mathbb{L}|} r_i L_i(X)$. *We have* $\sum_{i=1}^{|\mathbb{L}|} v(\omega_L^i)r(\omega_L^i) = v$, *if and only if there exists polynomials* $Q_X(X)$ *and* $D_{Z_{\mathbb{L}}}(X)$ *satisfying*

$$v(X) \cdot r(X) = \frac{v}{|\mathbb{L}|} + Q_X(X) \cdot X + D_{Z_{\mathbb{L}}}(X) \cdot Z_{\mathbb{L}}(X)$$

where both $Q_X(X)$ *and* $D_{Z_{\mathbb{L}}}(X)$ *are polynomials with degree* $< |\mathbb{L}| - 1$ *defined as*

$$Q_X(X) = \sum_{i=1}^{|\mathbb{L}|} v_i r_i \frac{L_i(X) - L_i(0)}{X},$$
$$D_{Z_{\mathbb{L}}}(X) = \sum_{i=1}^{|\mathbb{L}|} v_i r_i \frac{L_i^2(X) - L_i(X)}{Z_{\mathbb{L}}(X)} + \sum_{i \neq i'} v_i r_{i'} \frac{L_{i'}(X) L_i(X)}{Z_{\mathbb{L}}(X)}.$$

A.2 Polynomial Commitment Scheme

A *Polynomial Commitment Scheme* (PCS) [20] is defined by the following algorithms:

- $\mathsf{KeyGen}(\lambda, d, m) :\rightarrow \mathsf{par}$: It takes as input a security number $\lambda \in \mathbb{N}$, variate-degree bound $d \in \mathbb{N}$ and variate $m \in \mathbb{N}$, generates public parameter par.
- $\mathsf{Commit}(\mathsf{par}, f) \rightarrow \mathsf{com}_f$: It takes as input a polynomial $f(\mathbf{X})$ where $\mathbf{X} = (X_0, \ldots, X_{m-1})$ and par, outputs commitment com_f.
- $\mathsf{Open}(\mathsf{par}, \mathsf{com}_f, \mathbf{x}) \rightarrow (\pi_f, z)$: It takes as input par, com_f and the evaluation point $\mathbf{x}$, generates a proof π_f and the value $z = f(\mathbf{x})$.
- $\mathsf{Verify}(\mathsf{par}, \mathsf{com}_f, \mathbf{x}, \pi_f, z) \rightarrow b$: It is a protocol between the prover and the verifier. The verifier checks whether $z = f(\mathbf{x})$ through input par, $\mathsf{com}_f, \mathbf{x}, \pi_f$ and outputs $b \in \{0, 1\}$.

This polynomial commitment scheme has the following properties:

Completeness. For all $\lambda, d, m \in \mathbb{N}$, all evaluation points $\mathbf{x} \in \mathbb{Z}_p^m$, and any polynomial f with degree bound d and variate m, it holds that

$$\Pr\left[\mathsf{Verify}(\mathsf{par}, \mathsf{com}_f, \mathbf{x}, \pi_f, z) = 1 \;\middle|\; \begin{array}{l} \mathsf{par} \leftarrow \mathsf{KeyGen}(\lambda, d, m); \\ \mathsf{com}_f \leftarrow \mathsf{Commit}(\mathsf{par}, f); \\ (\pi_f, z) \leftarrow \mathsf{Open}(\mathsf{par}, \mathsf{com}_f, \mathbf{x}) \end{array}\right] = 1.$$

Knowledge soundness. For every PPT adversary $\mathcal{A}_{\mathsf{PCS}}$, there exists a PPT extractor $\mathcal{E}_{\mathsf{PCS}}$ with access to $\mathcal{A}_{\mathsf{PCS}}$'s messages during the protocol, it holds that

$$\Pr\left[\begin{array}{l} \mathsf{Verify}(\mathsf{par}, \mathsf{com}, \mathbf{x}, \pi, z) = 1 \wedge \\ \mathsf{com} = \mathsf{Commit}(\mathsf{par}, f) \wedge \\ f(\mathbf{x}) \neq z \end{array} \;\middle|\; \begin{array}{l} \mathsf{par} \leftarrow \mathsf{KeyGen}(\lambda, d, m); \\ (z, \mathbf{x}) \leftarrow \mathcal{A}_{\mathsf{PCS}}(\mathsf{par}, \lambda); \\ (\mathsf{com}, \pi) \leftarrow \mathcal{A}_{\mathsf{PCS}}(\mathsf{par}, \lambda); \\ f \leftarrow \mathcal{E}_{\mathsf{PCS}}^{\mathcal{A}_{\mathsf{PCS}}(\cdot)}(\mathsf{par}, \lambda) \end{array}\right] \leq \varepsilon(\lambda).$$

A.3 Leakage-Resilient Weak Pseudorandom Function

A leakage-resilient weak pseudorandom function (LRwPRF) is a pair of efficient algorithms $\Pi_{\mathsf{LRwPRF}} = (\mathsf{Setup}, \mathsf{Eval})$ with the following syntax:

– $\mathsf{Setup}(\lambda, \mathsf{leak}) \to \mathsf{rk}$. On input security parameter λ and a leakage parameter leak, it outputs a PRF key rk with bit-length $|\mathsf{rk}| = \kappa(\lambda, \mathsf{leak})$.
– $\mathsf{Eval}(\mathsf{rk}, z) \to y$. On input a PRF key rk and an input $z \in \mathcal{Z}$, it deterministically outputs a value $y \in \mathcal{Y}$. Here,

Leakage-resilient weak pseudorandomness. For an efficient adversary $\mathcal{A}$, define

$$\mathsf{Adv}^{\mathsf{LR}}_{\Pi_{\mathsf{LRwPRF}},\mathcal{A}}(\lambda) = \left|\Pr\left[\mathrm{E}^{\mathsf{LR}}_{\Pi_{\mathsf{LRwPRF}},\mathcal{A}}(\lambda, 0) = 1\right] - \Pr\left[\mathrm{E}^{\mathsf{LR}}_{\Pi_{\mathsf{LRwPRF}},\mathcal{A}}(\lambda, 1) = 1\right]\right|,$$

where the experiment $\mathrm{E}^{\mathsf{LR}}_{\Pi_{\mathsf{LRwPRF}},\mathcal{A}}(\lambda, \beta)$ is defined as follows:

1. **Initialization:** The adversary receives the security parameter λ and outputs a leakage parameter leak and a number $\mathsf{qt}_{\mathrm{pre}}$ of pre-challenge queries it wishes to make. The challenger runs to obtain $\mathsf{rk} \leftarrow \mathsf{Setup}(\lambda, \mathsf{leak})$.
2. **Pre-challenge queries:** The chalenger samples $z_1, \ldots, z_{\mathsf{qt}_{\mathrm{pre}}} \xleftarrow{\$} \mathcal{Z}$. It computes $y_i = \mathsf{Eval}(\mathsf{rk}, z_i))$ for each $i \in \mathsf{qt}_{\mathrm{pre}}$. It passes $\{(z_i, \mathsf{Eval}(\mathsf{rk}, z_i))\}_{i \in [\mathsf{qt}_{\mathrm{pre}}]}$ to the adversary.
3. **Leakage queries:** The adversary specifies a Boolean circuit $C_{\mathsf{leak}} : \{0,1\}^{\kappa} \to \{0,1\}^{\mathsf{leak}}$. The challenger returns $C_{\mathsf{leak}}(\mathsf{rk})$ to the adversary.
4. **Challenge.** The adversary outputs a number $\mathsf{qt}_{\mathsf{chall}}$ of challenge queries it wishes to make. The challenger samples $z'_1, \ldots, z'_{\mathsf{qt}_{\mathsf{chall}}} \xleftarrow{\$} \mathcal{Z}$. For each $i \in [\mathsf{qt}_{\mathsf{chall}}]$,
   – if $\beta = 0$, sets $y_i = \mathsf{Eval}(\mathsf{rk}, z'_i)$.
   – if $\beta = 1$, samples $y_i \xleftarrow{\$} \mathcal{Y}$.
   The challenger gives the list $\{(z'_i, y_i)\}_{i \in [\mathsf{qt}_{\mathsf{chall}}]}$ to the adversary.
5. **Output:** Finally, the adversary outputs a bit b'. The experiment returns b'.

We say that Π_{LRwPRF} satisfies leakage-resilient weak pseudorandomness if for every PPT adversary $\mathcal{A}$,

$$\mathsf{Adv}^{\mathsf{LR}}_{\Pi_{\mathsf{LRwPRF}},\mathcal{A}}(\lambda) \leq \varepsilon(\lambda).$$

A.4 One-Time Dual-Mode Bit Commitment

A one-time dual-mode bit commitment with extraction Π_{BC} is a tuple of algorithms $\Pi_{\mathsf{DMC}} = (\mathsf{SetupBind}, \mathsf{SetupEquiv}, \mathsf{Commit}, \mathsf{Verify}, \mathsf{Extract})$ with the following syntax:

– $\mathsf{SetupBind}(\lambda) \to (\mathsf{crs}, \mathsf{td})$: On input security parameter λ, it outputs a common reference string crs for the binding mode and trapdoor td.
– $\mathsf{SetupEquiv}(\lambda) \to (\mathsf{crs}, \tilde{\mathsf{com}}, \tilde{\sigma}_0, \tilde{\sigma}_1)$: On input security parameter λ, it outputs a common reference string crs for the equivocable mode, a commitment $\tilde{\mathsf{com}}$ and two openings $\tilde{\sigma}_0, \tilde{\sigma}_1$.
– $\mathsf{Commit}(\mathsf{crs}, b) \to (\mathsf{com}, \sigma)$: On input security parameter λ and a bit $\{0,1\}$, it outputs a commitment com and the opening σ.

– $\mathsf{Verify}(\mathsf{crs}, \mathsf{com}, b, \sigma) \to \{0,1\}$: On input security parameter λ and a bit $\{0,1\}$, a commitment com and opening σ, it outputs a bit $b' \in \{0,1\}$.
– $\mathsf{Extract}(\mathsf{td}, \mathsf{com}) \to \{0,1\}$: On input the trapdoor td and a commitment com, it outputs a bit $b \in \{0,1\}$.

Π_{DMC} is required to satisfy the following properties:

Correctness. For security parameters $\lambda \in \mathbb{N}$, common reference string crs generated by $\mathsf{SetupBind}(\lambda)$, and all bits $b \in \{0,1\}$

$$\Pr[\mathsf{Verify}(\mathsf{crs}, \mathsf{com}, b, \sigma) = 1 | (\mathsf{com}, \sigma) \leftarrow \mathsf{Commit}(\mathsf{crs}, b))] = 1.$$

Mode indistinguishability. For security parameters $\lambda \in \mathbb{N}$, it is hard to distinguish the binding mode and the equivocable mode, that is the advantage function

$$\mathsf{Adv}^{\mathsf{mod\text{-}ind}}_{\Pi_{\mathsf{DMC}},\mathcal{A}}(\lambda) = \left| \Pr\left[\mathcal{A}(\mathsf{com}, \sigma) = 1 \;\middle|\; \begin{array}{l} \mathsf{crs} \leftarrow \mathsf{SetupBind}(\lambda), \\ b \leftarrow \mathcal{A}(\lambda, \mathsf{crs}), \\ (\mathsf{com}, \sigma) \leftarrow \mathsf{Commit}(\mathsf{crs}, b) \end{array} \right] \right.$$
$$\left. - \Pr\left[\mathcal{A}(\mathsf{com}, \sigma) = 1 \;\middle|\; \begin{array}{l} (\mathsf{crs}, \tilde{\mathsf{com}}, \tilde{\sigma}_0, \tilde{\sigma}_1) \leftarrow \mathsf{SetupEquiv}(\lambda), \\ b \leftarrow \mathcal{A}(\lambda, \mathsf{crs}), \\ (\mathsf{com}, \sigma) = (\tilde{\mathsf{com}}, \tilde{\sigma}_b) \end{array} \right] \right|$$

is negligible in λ.

Extractability. For all parameters $\lambda \in \mathbb{N}$, and every adversary $\mathcal{A}$, we have

$$\Pr\left[b \neq b' \wedge \mathsf{Verify}(\mathsf{crs}, \mathsf{com}, b, \sigma) = 1 \;\middle|\; \begin{array}{l} (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{SetupBind}(\lambda); \\ (\mathsf{com}, \sigma, b) \leftarrow \mathcal{A}(\lambda, \mathsf{crs}); \\ b' \leftarrow \mathsf{Extract}(\mathsf{td}, \mathsf{com}) \end{array} \right] = \varepsilon(\lambda).$$

A.5 Rate-1 Somewhere Extractable Hash

A rate-1 somewhere extractable hash family is a tuple algorithms $\Pi_{\mathsf{flSEH}} = (\mathsf{Gen}, \mathsf{TrapGen}, \mathsf{Hash}, \mathsf{Validate}, \mathsf{Open}, \mathsf{Verify}, \mathsf{Extract})$ with the following syntax:

– $\mathsf{Gen}(\lambda, N) \to (\mathsf{hk}, \mathsf{vk})$: On input security parameter λ and the message length N, it outputs a (long) hashing key hk and a (short) verification key vk.
– $\mathsf{TrapGen}(\lambda, N, I) \to (\mathsf{hk}, \mathsf{vk}, \mathsf{td})$: On input security parameter λ, the message length N and a subset of indices $I \subset [N]$, it outputs a (long) hashing key hk, a (short) verification key vk and a trapdoor td.

– $\mathsf{Hash}(\mathsf{hk}, \mathbf{z}) \to (\mathsf{hv}, \mathsf{rt})$: On input hash key hk and a message $\mathbf{z} \in \{0,1\}^N$, it outputs a (long) hash value hv and a (short) digest rt.
– $\mathsf{Open}(\mathsf{hk}, \mathbf{z}, i) \to (b, \sigma)$: On input hash key hk, a message $\mathbf{z}$, an index i, it outputs a bit $b \in \{0,1\}$ and an opening σ.
– $\mathsf{Verify}(\mathsf{vk}, \mathsf{rt}, i, b, \sigma) \to \{0,1\}$: On input a verification key vk, a short digest rt, an index i, a bit b and an opening σ, it outputs 1 if (b, σ) is a valid opening against rt and outputs 0 otherwise.
– $\mathsf{Validate}(\mathsf{vk}, \mathsf{hv}, \mathsf{rt}) \to \{0,1\}$: On input a verification key vk, a hash value hv and a digest rt, it outputs 1 if hv and rt are consistent and outputs 0 otherwise.
– $\mathsf{Extract}(\mathsf{td}, \mathsf{hv}) \to \mathbf{z}_I$: On input a trapdoor td and a hash value hv, it outputs an extracted message $\mathbf{z}_I \in \{0,1\}^{|I|}$.

Π_{flSEH} is required to satisfy the following properties:

Completeness. For all $\lambda, N \in \mathbb{N}$, all subsets $I \subset [N]$ and any index $i \in [N]$, we have

$$\Pr\left[\begin{array}{l} b = z_i \wedge \mathsf{Verify}(\mathsf{vk}, \mathsf{rt}, i, b, \sigma) = 1 \\ \wedge \mathsf{Validate}(\mathsf{vk}, \mathsf{hv}, \mathsf{rt}) = 1 \end{array} \middle| \begin{array}{l} (\mathsf{hk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(\lambda, N); \\ (\mathsf{hv}, \mathsf{rt}) \leftarrow \mathsf{Hash}(\mathsf{hk}, \mathbf{z}); \\ (b, \sigma) \leftarrow \mathsf{Open}(\mathsf{hk}, \mathbf{z}, i) \end{array}\right] = 1.$$

Key indistinguishability. For $\lambda, N \in \mathbb{N}$, an adversary $\mathcal{A}$, it is hard to distinguish the keys generated from Gen and the keys generated from TrapGen, that is, the advantage function

$$\begin{aligned} \mathsf{Adv}^{\mathsf{key\text{-}ind}}_{\Pi_{\mathsf{flSEH}},\mathcal{A}}(\lambda) = \Bigg| & \Pr\left[\mathcal{A}(\mathsf{hk}, \mathsf{vk}) = 1 \;\middle|\; (\mathsf{hk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(\lambda, N)\right] \\ & - \Pr\left[\mathcal{A}(\mathsf{hk}, \mathsf{vk}) = 1 \;\middle|\; (\mathsf{hk}, \mathsf{vk}, \mathsf{td}) \leftarrow \mathsf{TrapGen}(\lambda, N, I)\right] \Bigg| \end{aligned}$$

is negligible in λ.

Extractability. For every $\lambda, N \in \mathbb{N}$, every subset $I \subseteq [N]$, and every PPT adversary $\mathcal{A}$, consider the following experiment:

$$\begin{aligned} (\mathsf{hk}, \mathsf{vk}, \mathsf{td}) &\leftarrow \mathsf{TrapGen}(1^\lambda, N, I), \\ \left(\mathsf{hv}, rt, \{(b_j, \sigma_j)\}_{j \in [N]}\right) &\leftarrow \mathcal{A}(\mathsf{hk}, \mathsf{vk}), \\ \widehat{z}_I &\leftarrow \mathsf{Extract}(\mathsf{td}, \mathsf{hv}). \end{aligned}$$

The adversary does not receive I or td. We require that

$$\Pr\left[\begin{array}{l} \mathsf{Validate}(\mathsf{vk}, \mathsf{hv}, \mathsf{rt}) = 1 \\ \wedge \bigwedge\limits_{j \in I} \mathsf{Verify}(\mathsf{vk}, \mathsf{rt}, j, b_j, \sigma_j) = 1 \\ \wedge \; (b_j)_{j \in I} \neq \widehat{\mathbf{z}}_I \end{array}\right] \leq \varepsilon(\lambda).$$

Here, $(b_j)_{j \in I}$ denotes the string consisting of the bits output by $\mathcal{A}$ at the positions in I.

B Hardness of Assumption 1

We prove that Assumption 1 holds in the generic bilinear group model of composite order, relying on the "master theorems" from [33,27]. We start by recalling the generic group model.

Definition 1 (Generic Bilinear Group Model). *Let $N \in \mathbb{N}$ be a positive integer and let $S \subseteq \{0,1\}^*$ be a set of strings of cardinality at least N. In the generic symmetric bilinear group model, group elements are represented by random encodings $\mathsf{List}_G, \mathsf{List}_T : \mathbb{Z}_N \to S$ where $\mathsf{List}_G(x)$ and $\mathsf{List}_T(x)$ are interpreted as the encodings of $g^x \in \mathbb{G}, g_T^x \in \mathbb{G}_T$, respectively, for fixed generators $g \in \mathbb{G}, g_T \in \mathbb{G}_T$. A generic algorithm is given oracle access to the following operations:*

- $\mathsf{Label}(i, x)$ *with $i \in \{\mathsf{G}, \mathsf{T}\}$ and value $x \in \mathbb{Z}_N$: if $i = \mathsf{G}$, return $\mathsf{List}_G(x)$; otherwise return $\mathsf{List}_T(x)$.*
- $\mathsf{GroupOp}(i, a_1, a_2, u_1, u_2)$ *with $i \in \{\mathsf{G}, \mathsf{T}\}$, $a_1, a_2 \in \mathbb{Z}_N$ and labels $u_1, u_2 \in S$: If there exists $x_1, x_2 \in \mathbb{Z}_N$ such that $\mathsf{List}_i(x_1) = u_1$ and $\mathsf{List}_i(x_2) = u_2$, then return $\mathsf{List}_i(a_1x_1 + a_2x_2)$; otherwise, return $\perp$.*
- $\mathsf{PairingOp}(u_1, u_2)$ *with labels $u_1, u_2 \in S$: If there exist $x_1, x_2 \in \mathbb{Z}_N$ such that $\mathsf{List}_G(x_1) = u_1, \mathsf{List}_G(x_2) = u_2$, then return $\mathsf{List}_\mathsf{T}(x_1 \cdot x_2)$ in the target group; otherwise, return $\perp$.*

Theorem 11 (Generic hardness of Assumption 1). *Assume that factoring a product of two primes (each of size 2^λ) is computationally hard. Then, for all polynomials*

$$\ell = \ell(\lambda), \qquad t = t(\lambda), \qquad s = s(\lambda),$$

the (ℓ, t, s)-pairwise bivariate subgroup decision exponent assumption holds in the generic composite-order bilinear group model of order $N = pq$, where p, q are primes of size 2^λ.

Proof. Let

$$\begin{aligned} S_{XY} &= \{(a,b) : 0 \le a \le \ell,\ 0 \le b \le t,\ (a,b) \ne (0,0)\}, \\ S_{XZ} &= \{(a,c) : 0 \le a \le \ell,\ 0 \le c \le s,\ (a,c) \ne (0,0)\}, \\ S_{YZ} &= \{(b,c) : 0 \le b \le t,\ 0 \le c \le s,\ (b,c) \ne (0,0)\}. \end{aligned}$$

We represent the public elements in the assumption as polynomials over the formal variables $X, Y, Z, R \in \mathbb{Z}_N$, where X, Y, Z correspond to τ_X, τ_Y, τ_Z, and R corresponds to the subgroup generator randomness r used in $g_p = g^{qr}, g_q = g^{pr}$. For every $(a,b) \in S_{XY}$, $(a,c) \in S_{XZ}$, and $(b,c) \in S_{YZ}$ define the source polynomials

$$\begin{gathered} P_{a,b}^{XY}(X,Y,Z,R) := [X^aY^bR,\ 0],\ P_{a,c}^{XZ}(X,Y,Z,R) := [X^aZ^cR,\ 0], \\ P_{b,c}^{YZ}(X,Y,Z,R) := [Y^bZ^cR,\ 0]. \end{gathered}$$

Let $\mathcal{P} := \{P^{XY}_{a,b}\}_{(a,b)\in S_{XY}} \cup \{P^{XZ}_{a,c}\}_{(a,c)\in S_{XZ}} \cup \{P^{YZ}_{b,c}\}_{(b,c)\in S_{YZ}}$. The challenge polynomials are

$$T_0(X,Y,Z,R) := [R,0], \qquad T_1(X,Y,Z,R) := [R,R].$$

These correspond exactly to the two challenge distributions g_p and $g_p g_q$. Then verify the following conditions:

- Every polynomial in $\mathcal{P}$ has total degree at most $\max\{\ell+t+1,\ \ell+s+1,\ t+s+1\}$, while T_0, T_1 have total degree 1.
- By construction, every non-constant polynomial in $\mathcal{P}$ contains at least one of the variables X, Y, Z, whereas T_0 and T_1 are monomials in R only. Therefore, any non-zero linear combination of polynomials in $\mathcal{P}$ either remains non-zero in at least one of the variables X, Y, Z, or is constant. In particular, neither T_0 nor T_1 can be expressed as a linear combination of $\mathcal{P}$. Thus T_0 and T_1 are independent of $\mathcal{P}$.
- For every polynomial $P_i \in \mathcal{P}$, the second component of P_i is zero. Hence $T_0 P_i = T_1 P_i$ for all i.
- Define

  $$S^{(b)} = \mathcal{P}^2 \cup \{T_b P_i : P_i \in \mathcal{P}\}.$$

  Every polynomial in $\mathcal{P}^2$ contains at least one of the variables X, Y, Z, since each polynomial in $\mathcal{P}$ does. Likewise, every polynomial of the form $T_b P_i$ also contains at least one of X, Y, Z. On the other hand, $T_0^2 = [R^2, 0], T_1^2 = [R^2, R^2]$ are monomials in R only. Therefore, for each $b \in \{0,1\}$, the polynomial T_b^2 cannot lie in the linear span of $S^{(b)}$, and thus T_b^2 is independent of $S^{(b)}$.

C Proof of Lemma 1

Lemma 5 (Weak somewhere extractability implies somewhere soundness).

Let Π_{BARG} be a weak somewhere-extractable BARG for Boolean circuit satisfiability. Suppose that the extraction requirement includes the consistency of the extracted public-input, gate-local, gate-wire, and output-wire values. Then Π_{BARG} satisfies somewhere soundness.

Proof. Let $\mathcal{A}$ be a PPT adversary and define

$$\epsilon(\lambda) := \mathsf{Adv}^{\mathsf{sSound}}_{\Pi_{\mathsf{BARG}},\mathcal{A}}(\lambda, \ell, t, s).$$

Augment the somewhere-soundness experiment by independently sampling

$$j^* \xleftarrow{\$} [t] \qquad \text{and} \qquad k^* \xleftarrow{\$} [s].$$

This does not change the marginal distribution of the original experiment.

Consider an accepting proof for which the hidden instance $\mathbf{x}_{i^*}$ is false. Let

$$\widehat{\boldsymbol{\nu}}_{i^*} = (\widehat{\nu}_{i^*,1}, \ldots, \widehat{\nu}_{i^*,t})$$

denote the induced global Boolean wire assignment, and let

$$\widehat{\mathbf{w}} = (\widehat{\nu}_{i^*,n+1}, \ldots, \widehat{\nu}_{i^*,n+h})$$

be its restriction to the witness wires. Public-input consistency and output satisfiability imply

$$\widehat{\nu}_{i^*,j} = x_{i^*,j} \quad \text{for all } j \in [n], \qquad \widehat{\nu}_{i^*,o_C} = 1.$$

Since $\mathbf{x}_{i^*}$ is false,

$$C(\mathbf{x}_{i^*}, \widehat{\mathbf{w}}) = 0.$$

Consequently, there exists a gate

$$G_{k_{\mathsf{bad}}} = (a_{k_{\mathsf{bad}}}, b_{k_{\mathsf{bad}}}, c_{k_{\mathsf{bad}}})$$

such that

$$\widehat{\nu}_{i^*,c_{k_{\mathsf{bad}}}} \neq \mathrm{NAND}\left(\widehat{\nu}_{i^*,a_{k_{\mathsf{bad}}}}, \widehat{\nu}_{i^*,b_{k_{\mathsf{bad}}}}\right).$$

Let

$$\left(\widehat{\nu}^{a}_{i^*,k_{\mathsf{bad}}}, \widehat{\nu}^{b}_{i^*,k_{\mathsf{bad}}}, \widehat{\nu}^{c}_{i^*,k_{\mathsf{bad}}}\right)$$

be the corresponding gate-local values. Either these values violate the NAND relation, or they satisfy the NAND relation but differ from the corresponding global wire values in at least one position. The former violates gate validity when $k^* = k_{\mathsf{bad}}$; the latter violates gate–wire consistency when $k^* = k_{\mathsf{bad}}$ and j^* equals an inconsistent wire index. Therefore,

$$\Pr[\mathsf{Bad} \mid \text{an accepting proof for a false } \mathbf{x}_{i^*}] \geq \frac{1}{ts}.$$

It follows that, in the normal-CRS experiment,

$$\Pr[\mathsf{Verify}(\mathsf{crs}, C, x_{1:\ell}, \pi) = 1 \wedge \mathsf{Bad}] \geq \frac{\epsilon(\lambda)}{ts}.$$

Replacing the normal CRS by a trapdoor CRS changes this probability by at most the CRS-indistinguishability advantage of some PPT adversary $\mathcal{D}$. In the resulting trapdoor experiment, the same event is bounded by the weak somewhere-extractability advantage of a PPT adversary $\mathcal{B}$. Hence,

$$\frac{\epsilon(\lambda)}{ts} \leq \mathsf{Adv}^{\mathsf{wExt\text{-}td}}_{\Pi_{\mathsf{BARG}},\mathcal{B}}(\lambda, \ell, t, s) + \mathsf{Adv}^{\mathsf{crs\text{-}ind}}_{\Pi_{\mathsf{BARG}},\mathcal{D}}(\lambda, \ell, t, s).$$

Both terms on the right-hand side are negligible. Since t and s are polynomially bounded in λ, $\epsilon(\lambda)$ is negligible. Therefore, Π_{BARG} satisfies somewhere soundness.

D Analysis of Construction 4.2

Theorem 12 (Completeness). *For any Boolean circuit C and any set of ℓ instance-witness pairs $\{(\mathbf{x}_i, \mathbf{w}_i)\}_{i=1}^{\ell}$ such that $C(\mathbf{x}_i, \mathbf{w}_i) = 1$ for all $i \in [\ell]$, if the prover follows the introduced protocol* BARG *correctly, then:*

$$\Pr\left[\mathsf{Verify}\left(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_\ell), \mathsf{Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_\ell), (\mathbf{w}_1, \ldots, \mathbf{w}_\ell))\right) = 1\right] = 1.$$

Proof. We prove completeness by demonstrating that each verification check passes when the prover honestly executes the protocol with valid inputs. For each $i \in [\ell], j \in [t]$, $\Phi(\omega_L^i, \omega_V^j) = v_{i,j}$. Write $\Phi(X, Y) = \sum_{i\in[0,\ell-1]} \sum_{j\in[0,t-1]} \phi_{i,j} X^i Y^j$. Since $A_{i,j} = g_p^{\tau_X^i \tau_Y^j}$, we can write

$$\Phi = \prod_{i\in[0,\ell-1], j\in[0,t-1]} A_{i,j}^{\phi_{i,j}} = g_p^{\sum_{i\in[0,\ell-1]} \sum_{j\in[0,t-1]} \phi_{i,j} \tau_X^i \tau_Y^j} = g_p^{\Phi(\tau_X, \tau_Y)}.$$

Similarly, we can write

$$\Phi_a = \prod_{i\in[0,\ell-1], k\in[0,s-1]} B_{i,k}^{\phi_{a,i,k}} = g_p^{\sum_{i\in[0,\ell-1]} \sum_{j\in[0,s-1]} \phi_{a,i,k} \tau_X^i \tau_Z^k} = g_p^{\Phi_a(\tau_X, \tau_Z)},$$

$$\Phi_b = \prod_{i\in[0,\ell-1], k\in[0,s-1]} B_{i,k}^{\phi_{b,i,k}} = g_p^{\sum_{i\in[0,\ell-1]} \sum_{j\in[0,s-1]} \phi_{b,i,k} \tau_X^i \tau_Z^k} = g_p^{\Phi_b(\tau_X, \tau_Z)},$$

$$\Phi_c = \prod_{i\in[0,\ell-1], k\in[0,s-1]} B_{i,k}^{\phi_{c,i,k}} = g_p^{\sum_{i\in[0,\ell-1]} \sum_{j\in[0,s-1]} \phi_{c,i,k} \tau_X^i \tau_Z^k} = g_p^{\Phi_c(\tau_X, \tau_Z)}.$$

1.Validity of wire assignments. For each $i \in [\ell], j \in [t]$, $0 = v_{i,j}^2 - v_{i,j} = \Phi(\omega_L^i, \omega_V^j)^2 - \Phi(\omega_L^i, \omega_V^j)$. There exists two polynomials $Q_\ell(X, Y)$ and $Q_t(X, Y)$ such that

$$Q_\ell(X, Y) \cdot Z_\ell(X) + Q_t(X, Y) \cdot Z_t(Y) = \Phi(X, Y)^2 - \Phi(X, Y).$$

Here, $Q_\ell(X, Y) = \sum_{i\in[0,\ell-1]} \sum_{j\in[0,t-1]} q_{\ell,i,j} X^i Y^j$ and $Q_t(X, Y) = \sum_{i\in[0,\ell-1]} \sum_{j\in[0,t-1]} q_{t,i,j} X^i Y^j$, so

$$Q_\ell = g_p^{\sum_{i\in[0,\ell-1]} \sum_{j\in[0,t-1]} q_{\ell,i,j} \tau_X^i \tau_Y^j} = g_p^{Q_\ell(\tau_X, \tau_Y)},$$

$$Q_t = g_p^{\sum_{i\in[0,\ell-1]} \sum_{j\in[0,t-1]} q_{t,i,j} \tau_X^i \tau_Y^j} = g_p^{Q_t(\tau_X, \tau_Y)}.$$

Moreover, $A_{0,0} = g_p^{\tau_X^0 \tau_Y^0} = g_p$, $Z_\ell = g_p^{Z_\ell(\tau_X)}$ and $Z_t = g_p^{Z_t(\tau_Y)}$. We write

$$\begin{aligned} e(\Phi, \Phi) &= e(g_p^{\Phi(\tau_X, \tau_Y)}, g_p^{\Phi(\tau_X, \tau_Y)}) = e(g_p, g_p)^{\Phi(\tau_X, \tau_Y)^2} \\ &= e(g_p, g_p)^{\Phi(\tau_X, \tau_Y)} \cdot e(g_p, g_p)^{Q_\ell(\tau_X, \tau_Y) Z_\ell(\tau_X)} \cdot e(g_p, g_p)^{Q_t(\tau_X, \tau_Y) Z_t(\tau_Y)} \\ &= e(\Phi, A_{0,0}) \cdot e(Q_\ell, Z_\ell) \cdot e(Q_t, Z_t), \end{aligned}$$

so the verification check for wire assignment passes.

2. Public-input consistency. By construction,

$$\Phi_{\text{in}}(\omega_L^i, \omega_V^j) = x_{i,j} \qquad \forall i \in [\ell],\ j \in [n].$$

Moreover, the honest wire polynomial satisfies

$$\Phi(\omega_L^i, \omega_V^j) = v_{i,j} = x_{i,j} \qquad \forall i \in [\ell],\ j \in [n],$$

because the first n wires contain the public input. Hence,

$$\Phi - \Phi_{\text{in}} \in \langle Z_\ell(X), Z_n(Y) \rangle.$$

Therefore, there exist polynomials Q_ℓ^{in} and Q_n^{in} satisfying

$$\Phi - \Phi_{\text{in}} = Q_\ell^{\text{in}} Z_\ell + Q_n^{\text{in}} Z_n,$$

and the public-input consistency verification equation holds.

3. Validity of gate computation. For each $i \in [\ell]$, gate $(a_k, b_k, c_k) \in [t]^3$ where $k \in [s]$, $0 = 1 - v_{i,c_k} - v_{i,a_k} v_{i,b_k} = 1 - \Phi_c(\omega_L^i, \omega_H^k) - \Phi_a(\omega_L^i, \omega_H^k)\Phi_b(\omega_L^i, \omega_H^k)$. There exist two polynomials $Q'_\ell(X, Y)$ and $Q'_s(X, Z)$ such that

$$Q'_\ell(X, Z) \cdot Z_\ell(X) + Q'_s(X, Z) \cdot Z_s(Z) = 1 - \Phi_c(X, Z) - \Phi_a(X, Z)\Phi_b(X, Z).$$

Here

$$Q'_\ell(X, Z) = \sum_{i \in [0, \ell-1]} \sum_{k \in [0, s-1]} q'_{\ell,i,k} X^i Z^k$$

$$Q'_s(X, Z) = \sum_{i \in [0, \ell-1]} \sum_{k \in [0, s-1]} q'_{s,i,k} X^i Z^k,$$

so

$$Q'_\ell = g_p^{\sum_{i \in [0,\ell-1]} \sum_{k \in [0,s-1]} q'_{\ell,i,k} \tau_X^i \tau_Z^k} = g_p^{Q'_\ell(\tau_X, \tau_Z)},$$

$$Q'_s = g_p^{\sum_{i \in [0,\ell-1]} \sum_{k \in [0,s-1]} q'_{s,i,k} \tau_X^i \tau_Z^k} = g_p^{Q'_s(\tau_X, \tau_Z)}.$$

We write

$$\begin{aligned}
&e(B_{0,0}, B_{0,0}) = e(g_p, g_p) \\
&= e(g_p, g_p)^{\Phi_a(\tau_X,\tau_Z)\Phi_b(\tau_X,\tau_Z) + \Phi_c(\tau_X,\tau_Z) + Q'_\ell(\tau_X,\tau_Z) Z_\ell(\tau_X) + Q'_s(\tau_X,\tau_Z) Z_s(\tau_Z)} \\
&= e(g_p^{Q'_\ell(\tau_X,\tau_Z)}, g_p^{Z_\ell(\tau_X)}) e(g_p^{Q'_s(\tau_X,\tau_Z)}, g_p^{Z_s(\tau_Z)}) e(g_p^{\Phi_c(\tau_X,\tau_Z)}, g_p) e(g_p^{\Phi_a(\tau_X,\tau_Z)}, g_p^{\Phi_b(\tau_X,\tau_Z)}) \\
&= e(Q'_\ell, Z_\ell) \cdot e(Q'_s, Z_s) \cdot e(\Phi_c, B_{0,0}) \cdot e(g_p^{\Phi_a(\tau_X,\tau_Z)}, g_p^{\Phi_b(\tau_X,\tau_Z)}),
\end{aligned}$$

so the gate validity check passes.

4. Validity of gate-wire consistency. For each $i \in [\ell], k \in [s]$, $\Phi_a(\omega_L^i, \omega_H^k) = \Phi(\omega_L^i, A(\omega_H^k))$. Equivalently,

$$\Phi_a(\omega_L^i, \omega_H^k) = \sum_{y \in \mathbb{H}} \Phi(\omega_L^i, y) \cdot \mathsf{eq}_a(\omega_H^k, y),$$

which implies the following identity holds:

$$\Phi(X, Y) \cdot \mathsf{eq}_a(Z, Y) - \Phi_a(X, Z) \cdot t^{-1} = H_a(Y) \cdot Z_t(Y) + Y \cdot F_a(Y).$$

Here $H_a(Y) = \sum_{j \in [0,t-1]} h_{a,j} Y^j$, $F_a(Y) = \sum_{j \in [0,t-1]} f_{a,j} Y^j$. Thus,

$$H_a = g_p^{\sum_{j \in [0,t-1]} h_{a,j} \tau_Y^j} = g_p^{H_a(\tau_Y)}, F_a = g_p^{\sum_{j \in [0,t-1]} f_{a,j} \tau_Y^j} = g_p^{F_a(\tau_Y)}.$$

Moreover, $\mathsf{eq}_a(Z, Y) = \sum_{j \in [t], k \in [s]} eq_{j,k} Y^j Z^k$ and compute $\mathsf{eq}_a = g_p^{\sum_{j \in [t], k \in [s]} eq_{j,k} \tau_Y^j \tau_Z^k} = g_p^{\mathsf{eq}_a(\tau_Z, \tau_Y)}$. We write

$$\begin{aligned} e(\Phi, \mathsf{eq}_a) \cdot e(\Phi_a, B_{0,0}^{t^{-1}}) &= e(g_p^{\Phi(\tau_X, \tau_Y)}, g_p^{\mathsf{eq}_a(\tau_Z, \tau_Y)} \cdot e(g_p^{\Phi_a(\tau_X, \tau_Z)}, g_p^{t^{-1}}) \\ &= e(g_p^{H_a(\tau_Y)}, g_p^{Z_t(\tau_Y)}) \cdot e(g_p^{F_a(\tau_Y)}, g_p^{\tau_Y}) \\ &= e(H_a, Z_t) \cdot e(F_a, A_{0,1}) \end{aligned}$$

so this sumcheck relation check passes. Similarly, the other two sumcheck relations for $\Phi_b(X, Z)$ and $\Phi_c(X, Z)$ also pass.

5. Output satisfiability. For $i \in [\ell]$, $v_{i,t} = \Phi(\omega_L^i, \omega_V^t) = \Phi_t(\omega_L^i) = 1$, so there exists $\hat{Q}_t(X, Y)$ satisfying

$$\Phi(X, Y) - 1 = \hat{Q}_t(X, Y) \cdot (Y - \omega_V^t).$$

Here $\hat{Q}_t(X, Y) = \sum_{i \in [0,\ell-1]} \sum_{j \in [0,t-1]} \hat{q}_{t,i,j} X^i Y^j$. Thus $\hat{Q}_t = g_p^{\hat{Q}_t(\tau_X, \tau_Y)} = g_p^{\hat{Q}_t(\tau_X, \tau_Y)}$. We write

$$\begin{aligned} e(\Phi, A_{0,0}) &= e(g_p^{\Phi(\tau_X, \tau_Y)}, g_p) = e(g_p, g_p) \cdot e(g_p^{\hat{Q}_t(\tau_X, \tau_Y)}, g_p^{\tau_Y - \omega_V^t} \\ &= e(A_{0,0}, A_{0,0}) \cdot e(\hat{Q}_t, Z_{t,t}). \end{aligned}$$

so the verifier's output check passes.

All five categories of checks above are satisfied when the prover is honest and the underlying protocols are complete. Therefore the verifier accepts with probability 1. This completes the proof of completeness. □

Theorem 13 (Weak Somewhere Extractability). *Assume that the (ℓ, t, s)-pairwise bivariate subgroup decision exponent assumption holds over composite-order pairing groups. Then Construction 4.2 satisfies weak somewhere extractability.*

Proof. Define TrapSetup and Extract algorithms as follow:

– $\mathsf{TrapSetup}(\lambda, \ell, i^*, j^*, k^*) \to (\mathsf{crs}, \mathsf{td})$:

1. Run $\mathcal{G} := (\mathbb{G}, \mathbb{G}_T, p, q, e, g) \leftarrow \mathsf{CompGroupGen}(\lambda)$. Let $N = pq$, sample randomly $r \leftarrow \mathbb{Z}_N$. Set $g_p = g^{qr}$ and $\mathcal{G}_N = (\mathbb{G}, \mathbb{G}_T, N, g, e)$.
2. Sample $\tau_X, \tau_Y, \tau_Z \leftarrow \mathbb{Z}_N$. For each $i \in [0, \ell-1], j \in [0, t-1], k \in [0, s-1]$, let $A_{i,j} = g_p^{\tau_X^i \tau_Y^j} g_q^{(\omega_L^{i^*})^i (\omega_V^{j^*})^j}$, $B_{i,k} = g_p^{\tau_X^i \tau_Z^k} g_q^{(\omega_L^{i^*})^i (\omega_H^{k^*})^k}$ and $C_{j,k} = g_p^{\tau_Y^j \tau_Z^k}$.
3. Denote three multiplicative subgroups $\mathbb{L} = \{\omega_L, \ldots, \omega_L^\ell\}, \mathbb{V} = \{\omega_V, \ldots, \omega_V^t\}, \mathbb{H} = \{\omega_H, \ldots, \omega_H^s\}$ such that $|\mathbb{L}| = \ell, |\mathbb{V}| = t, |\mathbb{H}| = s$. Define the vanishing polynomials $Z_n(Y) = \prod_{j\in[n]}(Y - \omega_V^j), Z_\ell(X) = \prod_{i\in[\ell]}(X - \omega_L^i), Z_t(Y) = \prod_{j\in[t]}(Y - \omega_V^j), Z_s(Z) = \prod_{k\in[s]}(Z - \omega_H^k)$. Compute $Z_n = g_p^{Z_n(\tau_Y)}$, $Z_\ell = g_p^{Z_\ell(\tau_X)}$, $Z_t = g_p^{Z_t(\tau_Y)}$ and $Z_s = g_p^{Z_s(\tau_Z)}$. Particularly, set $Z_{t,t}(Y) = Y - \omega_V^t$ and set $Z_{t,t} = g_p^{Z_{t,t}(\tau_Y)}$.
4. Output the extraction trapdoor $\mathsf{td} = g_q$ and the CRS

$$\mathsf{crs} = (\mathcal{G}_N, \{A_{i,j}, B_{i,k}, C_{j,k}\}_{i\in[0,\ell-1], j\in[0,t-1], k\in[0,s-1]}, Z_n, Z_\ell, Z_t, Z_s, Z_{t,t}).$$

– $\mathsf{Extract}(\mathsf{td}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_\ell), \pi) \to (v_{i^*,j^*}, \mathbf{v}_{i^*,G_{k^*}})$: The extractor recovers the target wire value by checking whether the corresponding commitment contains a nontrivial component in the hidden subgroup. Concretely, it sets

$$v_{i^*,j^*} = 0 \quad \text{if } e(\Phi, g_q) = 1, \qquad v_{i^*,j^*} = 1 \quad \text{otherwise.}$$

If $j^* \in [n+1, n+h]$, then the extracted value v_{i^*,j^*} corresponds to a witness wire value. More generally, the extractor can also recover the wire values associated with a target gate. Let $G_{k^*} = (a_{k^*}, b_{k^*}, c_{k^*})$ be the target gate. The extractor recovers these wire values analogously:

$$v_{i^*,a_{k^*}} = 0 \quad \text{if } e(\Phi_a, g_q) = 1, \qquad v_{i^*,a_{k^*}} = 1 \quad \text{otherwise;}$$

$$v_{i^*,b_{k^*}} = 0 \quad \text{if } e(\Phi_b, g_q) = 1, \qquad v_{i^*,b_{k^*}} = 1 \quad \text{otherwise;}$$

$$v_{i^*,c_{k^*}} = 0 \quad \text{if } e(\Phi_c, g_q) = 1, \qquad v_{i^*,c_{k^*}} = 1 \quad \text{otherwise.}$$

Thus, besides extracting the target witness wire (when j^* lies in the witness range), the extractor is also able to recover the values $\mathbf{v}_{i^*,G_{k^*}} = (v_{i^*,a_{k^*}}, v_{i^*,b_{k^*}}, v_{i^*,c_{k^*}})$ of the three wires associated with the target gate G_{k^*}.

□

Lemma 6 (CRS Indistinguishability). *Assume that the (ℓ, t, s)-pairwise bivariate subgroup decision exponent assumption holds over composite-order pairing groups. Then the normal CRS and the trapdoor CRS generated for any target tuple (i^*, j^*, k^*) are computationally indistinguishable.*

Proof. Suppose there exists a PPT adversary $\mathcal{A}$ that distinguishes a normal CRS from a trapdoor CRS with non-negligible advantage. We use $\mathcal{A}$ to construct a reduction $\mathcal{B}$ that breaks the bivariate subgroup decision exponent assumption.

– $\mathcal{B}$ receives $\mathcal{G}_N = (\mathbb{G}, \mathbb{G}_T, N, g, e)$, $\{g_p^{\tau_X^a \tau_Y^b}, g_p^{\tau_X^a \tau_Z^c}, g_p^{\tau_Y^b \tau_Z^c}\}_{a\in[0,\ell],b\in[0,t],c\in[0,s]}$, and a challenge element $T \in \mathbb{G}$ where either $T = g_p$ or $T = g_p g_q$.

– $\mathcal{B}$ uses $\mathcal{A}$ to construct the CRS as follows:

- For each $(i, j) \in [0, \ell] \times [0, t]$, define

$$A_{i,j} = \prod_{a=0}^{i} \prod_{b=0}^{j} \left(G_{a,b}^{XY}\right)^{\binom{i}{a}\binom{j}{b}(\omega_L^{i^*})^{i-a}(\omega_V^{j^*})^{j-b}},$$

  where

$$G_{a,b}^{XY} = \begin{cases} T, & (a,b) = (0,0), \\ g_p^{\tau_X^a \tau_Y^b}, & \text{otherwise.} \end{cases}$$

  Similarly, it defines $B_{i,k}$ and $C_{j,k}$ using the pairs (X, Z) and (Y, Z).

- Define the vanishing polynomials $Z_\ell(X) = \prod_{i\in[\ell]}(X-\omega_L^i), Z_t(Y) = \prod_{j\in[t]}(Y-\omega_V^j), Z_s(Z) = \prod_{k\in[s]}(Z-\omega_H^k)$. Compute $Z_\ell = \prod_{i\in[0,\ell]} A_{i,0}^{z_{\ell,i}}$, $Z_t = \prod_{j\in[0,t]} A_{0,j}^{z_{t,j}}$ and $Z_s = \prod_{k\in[s]} B_{0,k}^{z_{s,k}}$.

– $\mathcal{B}$ sends crs to $\mathcal{A}$ and outputs whatever $\mathcal{A}$ outputs.

- If $T = g_p$, then

$$A_{i,j} = g_p^{(\tau_X+\omega_L^{i^*})^i(\tau_Y+\omega_V^{j^*})^j},$$

  for all $0 \le i \le \ell$, $0 \le j \le t$. Similarly,

$$B_{i,k} = g_p^{(\tau_X+\omega_L^{i^*})^i(\tau_Z+\omega_H^{k^*})^k}, \quad C_{j,k} = g_p^{(\tau_Y+\omega_V^{j^*})^j(\tau_Z+\omega_H^{k^*})^k}.$$

  Moreover, $Z_\ell = \prod_{i\in[0,\ell]} A_{i,0}^{z_{\ell,i}} = g_p^{Z_\ell(\tau_X+\omega_L^{i^*})}$, $Z_t = \prod_{j\in[0,t]} A_{0,j}^{z_{t,j}} = g_p^{Z_t(\tau_Y+\omega_V^{j^*})}$. Therefore, the resulting CRS is distributed exactly as a normal CRS.

- If $T = g_p g_q$, then this T-term contributes an additional g_q-component

$$A_{i,j} = g_p^{(\tau_X+\omega_L^{i^*})^i(\tau_Y+\omega_V^{j^*})^j} \cdot g_q^{(\omega_L^{i^*})^i(\omega_V^{j^*})^j},$$

$$B_{i,k} = g_p^{(\tau_X+\omega_L^{i^*})^i(\tau_Z+\omega_H^{k^*})^k} \cdot g_q^{(\omega_L^{i^*})^i(\omega_H^{k^*})^k},$$

  and

$$C_{j,k} = g_p^{(\tau_Y+\omega_V^{j^*})^j(\tau_Z+\omega_H^{k^*})^k} \cdot g_q^{(\omega_V^{j^*})^j(\omega_H^{k^*})^k}.$$

  Moreover,

$$Z_\ell = g_p^{Z_\ell(\tau_X+\omega_L^{i^*})}, \quad Z_t = g_p^{Z_t(\tau_Y+\omega_V^{j^*})}, \quad Z_s = g_p^{Z_s(\tau_Z+\omega_H^{k^*})}$$

since the g_q components vanish:

$$Z_\ell(\omega_L^{i^*}) = 0, \quad Z_t(\omega_V^{j^*}) = 0, \quad Z_s(\omega_H^{k^*}) = 0.$$

The resulting CRS is distributed exactly as the trapdoor CRS for the target tuple (i^*, j^*, k^*). Therefore, if $\mathcal{A}$ distinguishes the normal CRS from the trapdoor CRS with non-negligible advantage, then $\mathcal{B}$ distinguishes whether $T = g_p$ or $T = g_p g_q$ with the same non-negligible advantage, contradicting the (ℓ, t, s)-pairwise bivariate subgroup decision exponent assumption.

□

Lemma 7 (Weak Extractable in Trapdoor Mode). *Construction 4.2 is statistically weak somewhere extractable in trapdoor mode.*

Proof. Let Π_{BARG} be the above bivariate BARG construction 4.2 over a composite-order pairing group. Let $(i^*, j^*, k^*) \in [\ell] \times [t] \times [s]$ be a target tuple. Consider the following trapdoor CRS:

$$A_{i,j} = g_p^{\tau_X^i \tau_Y^j} g_q^{(\omega_L^{i^*})^i (\omega_V^{j^*})^j}, B_{i,k} = g_p^{\tau_X^i \tau_Z^k} g_q^{(\omega_L^{i^*})^i (\omega_H^{k^*})^k}, C_{j,k} = g_p^{\tau_Y^j \tau_Z^k} g_q^{(\omega_V^{j^*})^j (\omega_H^{k^*})^k}.$$

Moreover, let

$$Z_n = g_p^{Z_n(\tau_Y)}, \qquad Z_\ell = g_p^{Z_\ell(\tau_X)}, \qquad Z_t = g_p^{Z_t(\tau_Y)}, \qquad Z_s = g_p^{Z_s(\tau_Z)}$$

be pure $\mathbb{G}_p$ elements. Let $\pi = (\Phi, \Phi_a, \Phi_b, \Phi_c, Q_\ell^{\mathsf{in}}, Q_n^{\mathsf{in}}, Q_\ell, Q_t, Q'_\ell, Q'_s, H_a, H_b, H_c, F_a, F_b, F_c, \hat{Q}_t)$ be any proof accepted by the verifier under the trapdoor CRS $\mathsf{crs}^* = (\mathcal{G}_N, \{A_{i,j}, B_{i,k}, C_{j,k}\}_{i\in[0,\ell-1], j\in[0,t-1], k\in[0,s-1]}, Z_n, Z_\ell, Z_t, Z_s, Z_{t,t})$.

Since $\mathbb{G}$ is a composite-order group of order $N = pq$, every group element $U \in \mathbb{G}$ can be uniquely decomposed as $U = g_p^{u_p} g_q^{u_q}$ for some $u_p \in \mathbb{Z}_p$ and $u_q \in \mathbb{Z}_q$. Write

$$\Phi = g_p^{\rho} g_q^{\gamma}, \quad \Phi_a = g_p^{\rho_a} g_q^{\gamma_a}, \quad \Phi_b = g_p^{\rho_b} g_q^{\gamma_b}, \quad \Phi_c = g_p^{\rho_c} g_q^{\gamma_c}.$$

The extraction trapdoor is $\mathsf{td} = g_q$. By the definition of the trapdoor CRS, the $\mathbb{G}_q$ components of Φ, Φ_a, Φ_b, and Φ_c are exactly the evaluation of the committed polynomials at the target point:

$$\gamma = \Phi(\omega_L^{i^*}, \omega_V^{j^*}), \ \gamma_a = \Phi_a(\omega_L^{i^*}, \omega_H^{k^*}), \ \gamma_b = \Phi_b(\omega_L^{i^*}, \omega_H^{k^*}), \ \gamma_c = \Phi_c(\omega_L^{i^*}, \omega_H^{k^*}).$$

$\gamma = w_{\omega_L^{i^*}, \omega_V^{j^*}}$ if $\omega_V^{j^*} \in [n+1, n+h]$. Moreover, $\gamma = x_{\omega_L^{i^*}, \omega_V^{j^*}}$ if $\omega_V^{j^*} \in [n]$. Then we consider the following verification checks:

– Wire-assignments. We first prove that the extracted target wire value is Boolean. Since the wire-assignment verification equation is accepted, we have

$$e(\Phi, \Phi) = e(Q_\ell, Z_\ell) \cdot e(Q_t, Z_t) \cdot e(\Phi, A_{0,0}).$$

By construction, $Z_\ell, Z_t \in \mathbb{G}_p$, and in the trapdoor CRS $A_{0,0} = g_p g_q$. Projecting the above equation to the $\mathbb{G}_q$ component gives

$$\Phi^2 = \Phi \quad \Rightarrow \quad \gamma \in \{0,1\}.$$

Thus, the extractor

$$v_{i^*,j^*} = \begin{cases} 0, & e(g_q, \Phi) = 1, \\ 1, & e(g_q, \Phi) \neq 1. \end{cases}$$

– Gate-computation verification. We have

$$e(Q'_\ell, Z_\ell) \cdot e(Q'_s, Z_s) \cdot e(\Phi_c, B_{0,0}) \cdot e(\Phi_a, \Phi_b) = e(B_{0,0}, B_{0,0}).$$

Since $Z_\ell, Z_s \in \mathbb{G}_p$ and $B_{0,0} = g_p g_q$, the $\mathbb{G}_q$ component of this equation gives

$$\gamma_c + \gamma_a \gamma_b = 1 \quad \Rightarrow \quad \gamma_c = 1 - \gamma_a \gamma_b.$$

– Wire-gate consistency. $\gamma_a, \gamma_b, \gamma_c \in \{0,1\}$ are evaluations of the Boolean wire-assignment polynomial Φ at the corresponding input and output wires of the target gate, then by the sumcheck relation for fixed $(X, Z) = (\omega_L^{i^*}, \omega_H^{k^*})$

$$\Phi_a(\omega_L^{i^*}, \omega_H^{k^*}) = \sum_{y \in \mathbb{V}} \Phi(\omega_L^{i^*}, y) \mathsf{eq}_a(\omega_H^{k^*}, y).$$

Since $\mathsf{eq}_a(\omega_H^{k^*}), y)$ selects $y = \omega_V^{a_{k^*}}$, we obtain

$$\Phi_a(\omega_L^{i^*}, , \omega_H^{k^*}) = \Phi(\omega_L^{i^*}, \omega_V^{a_{k^*}}) \Rightarrow \gamma_a = \gamma.$$

Similarly, we have

$$\gamma_b = \Phi(\omega_L^{i^*}, \omega_V^{b_{k^*}}) = \gamma \quad \text{if} \quad y = \omega_V^{b_{k^*}}$$
$$\gamma_c = \Phi(\omega_L^{i^*}, \omega_V^{c_{k^*}}) = \gamma \quad \text{if} \quad y = \omega_V^{c_{k^*}}.$$

– Public-input consistency. Let $\gamma_{\mathsf{in}} = \Phi_{\mathsf{in}}(\omega_L^{i^*}, \omega_V^{j^*})$. Projecting the public-input consistency equation

$$e(\Phi, A_{0,0}) = e(\Phi_{\mathsf{in}}, A_{0,0}) \cdot e(Q_\ell^{\mathsf{in}}, Z_\ell) \cdot e(Q_n^{\mathsf{in}}, Z_n)$$

onto the G_q component gives

$$\gamma - \gamma_{\mathsf{in}} = Q_\ell^{\mathsf{in}}(\omega_L^{i^*}, \omega_V^{j^*}) Z_\ell(\omega_L^{i^*}) + Q_n^{\mathsf{in}}(\omega_L^{i^*}, \omega_V^{j^*}) Z_n(\omega_V^{j^*}).$$

If $j^* \in [n]$, then

$$Z_\ell(\omega_L^{i^*}) = 0 \qquad \text{and} \qquad Z_n(\omega_V^{j^*}) = 0.$$

Moreover, by the definition of Φ_{in},

$$\gamma_{\mathsf{in}} = \Phi_{\mathsf{in}}(\omega_L^{i^*}, \omega_V^{j^*}) = x_{i^*,j^*}.$$

Therefore,

$$j^* \in [n] \quad \Longrightarrow \quad \gamma = x_{i^*,j^*}.$$

– Output-wire consistency. The output-satisfiability equation enforces

$$\Phi(\omega_L^i, \omega_V^{o_C}) = 1 \qquad \text{for every } i \in [\ell].$$

Consequently,

$$j^* = o_C \quad \Longrightarrow \quad \gamma = \Phi(\omega_L^{i^*}, \omega_V^{o_C}) = 1.$$

Moreover, gate–wire consistency gives

$$\gamma_c = \Phi(\omega_L^{i^*}, \omega_V^{c_{k^*}}).$$

Hence,

$$c_{k^*} = o_C \quad \Longrightarrow \quad \gamma_c = \Phi(\omega_L^{i^*}, \omega_V^{o_C}) = 1.$$

Combining the above relations with the Booleanity and gate-validity conditions, the extracted values satisfy

$$\begin{aligned}
&j^* \in [n] \Longrightarrow \widehat{v}_{i^*,j^*} = x_{i^*,j^*},\\
&j^* = a_{k^*} \Longrightarrow \widehat{v}_{i^*,j^*} = \widehat{v}_{i^*,a_{k^*}},\\
&j^* = b_{k^*} \Longrightarrow \widehat{v}_{i^*,j^*} = \widehat{v}_{i^*,b_{k^*}},\\
&j^* = c_{k^*} \Longrightarrow \widehat{v}_{i^*,j^*} = \widehat{v}_{i^*,c_{k^*}},\\
&\widehat{v}_{i^*,c_{k^*}} = \mathrm{NAND}\left(\widehat{v}_{i^*,a_{k^*}}, \widehat{v}_{i^*,b_{k^*}}\right),\\
&j^* = o_C \Longrightarrow \widehat{v}_{i^*,j^*} = 1,\\
&c_{k^*} = o_C \Longrightarrow \widehat{v}_{i^*,c_{k^*}} = 1.
\end{aligned}$$

Thus, all conditions in the definition of weak somewhere extractability are satisfied.

E Analysis of Construction 5.1

Theorem 14 (Correctness). *If* Π_{BARG} *is complete and* Π_{BC} *is correct, then construction 5.1 is correct.*

Proof. Let λ be a security parameter, m be an output length, $I \subseteq [m]$ a set of indices. Let $\mathsf{crs} \leftarrow \mathsf{Setup}(\lambda, m)$, $(\mathsf{st}, \mathbf{r}) \leftarrow \mathsf{GBits}(\mathsf{crs})$, and $\pi_{\mathsf{HBG}} \leftarrow \mathsf{Prove}(\mathsf{st}, I)$. Consider the output of $\mathsf{Verify}(\mathsf{crs}, I, \mathbf{r}_I, \pi_{\mathsf{HBG}})$.

– Let $\pi_{\mathsf{HBG}} = (\pi_{\mathsf{BARG}}, \{\mathsf{com}_j^{\mathsf{rk}}\}_{j\in[\kappa]})$. By correctness of Π_{BC}, $(\mathbf{x}_i, \mathbf{w}_i) \in \mathcal{R}$ and thus

$$\mathsf{BC.Verify}(\mathsf{crs}_{\mathsf{BC}}, \mathsf{com}_{\mathsf{rk}_j}, \mathsf{rk}_j, \sigma_{\mathsf{rk}_j}) = 1, \quad \text{for each } j \in [\kappa].$$

– Since $(\mathbf{x}_i, \mathbf{w}_i) \in \mathcal{R}$ for each $i \in I$, completeness of BARG implies that

$$\mathsf{BARG.Verify}(\mathsf{crs}_{\mathsf{BARG}}, C_{\mathcal{R}}, (\mathbf{x}_1, \ldots, \mathbf{x}_{|I|}, \pi_{\mathsf{BARG}}) = 1.$$

Since both checks pass, we conclude that $\mathsf{Verify}(\mathsf{crs}, I, \mathbf{r}_I, \pi_{\mathsf{HBG}}) = 1$. □

Theorem 15 (Somewhat computational binding). *Assume that:*

1. Π_{BC} *is statistically binding and extractable in the binding mode;*
2. Π_{BARG} *satisfies somewhere soundness;*
3. $\kappa \le m^{\gamma} \cdot \mathsf{poly}(\lambda)$ *for some constant* $\gamma < 1$ *and fixed polynomial* $\mathsf{poly}(\cdot)$.

Then Construction 5.1 is somewhat computationally binding.

Proof. Let

$$V_{\mathsf{crs}} := \left\{ \left(\mathsf{LRwPRF.Eval}(\widehat{\mathsf{rk}}, z_1), \ldots, \mathsf{LRwPRF.Eval}(\widehat{\mathsf{rk}}, z_m) \right) : \widehat{\mathsf{rk}} \in \{0,1\}^{\kappa} \right\}.$$

Output sparsity. By $\kappa \le m^{\gamma} \cdot \mathsf{poly}(\lambda)$, we have

$$|V_{\mathsf{crs}}| \le 2^{\kappa} \le 2^{m^{\gamma} \cdot \mathsf{poly}(\lambda)}.$$

Thus, V_{crs} is sparse.

Computational binding. Suppose that an adversary outputs an accepting opening $(I, \mathbf{r}_I, \pi_{\mathsf{HBG}})$ such that $\mathbf{r}_I \notin V^I_{\mathsf{crs}}$ with non-negligible probability. By the extractability and statistical binding of the outer commitments, the commitments in π_{HBG} determine a unique key

$$\widehat{\mathsf{rk}} = (\widehat{\mathsf{rk}}_1, \ldots, \widehat{\mathsf{rk}}_{\kappa}).$$

Hence, there exists an index $i_{\mathsf{bad}} \in I$ such that

$$r_{i_{\mathsf{bad}}} \neq \mathsf{LRwPRF.Eval}(\widehat{\mathsf{rk}}, \mathbf{z}_{i_{\mathsf{bad}}}).$$

Therefore, the corresponding instance of the relation verified by $C_{\mathcal{R}}$ is false, while the BARG proof is accepting.

Construct a BARG adversary that runs the above adversary and outputs the resulting BARG proof together with the corresponding batch of instances. Since at least one instance in this batch is false, a uniformly sampled hidden instance is false with probability at least $1/|I|$. Consequently,

$$\Pr[\text{An accepting opening outside } V^I_{\mathsf{crs}}] \le |I| \cdot \mathsf{Adv}^{\mathsf{sSound}}_{\Pi_{\mathsf{BARG}}, \mathcal{B}}(\lambda, |I|, t, s).$$

Since $|I| \le m$ and Π_{BARG} satisfies somewhere soundness, the right-hand side is negligible. This contradicts the assumed existence of such an adversary. Therefore, Construction 5.1 is somewhat computationally binding.

Theorem 16 (Computational Hiding). *Assume that* Π_{BC} *satisfies mode indistinguishability and* LRwPRF *is a secure leakage-resilient weak PRF. If* $\mathsf{leak}(\lambda, m) \ge |\pi_{\mathsf{BARG}}(\lambda, m)|$, *construction 5.1 satisfies computational hiding.*

Proof. We prove computational hiding via a sequence of hybrids. Fix any PPT adversary $\mathcal{A}$ and any opened subset $I \subseteq [m]$. Let $\bar{I} = [m] \setminus I$. We need to show that

$$(\mathsf{crs}, \mathbf{r}_I, \pi_{\mathsf{HBG}}, \mathbf{r}_{\bar{I}}) \approx_c (\mathsf{crs}, \mathbf{r}_I, \pi_{\mathsf{HBG}}, \mathbf{u}),$$

where $\mathbf{u} \leftarrow \{0,1\}^{|\bar{I}|}$. Recall that

$$\pi_{\mathsf{HBG}} = (\pi_{\mathsf{BARG}}, \{\mathsf{com}^{\mathsf{rk}}_j\}_{j\in[\kappa]}).$$

- Hyb_0. This is the real hiding experiment. Namely,
  - for each $j \in [\kappa]$, sample $(\mathsf{crs}^{(j)}_{\mathsf{BC}}, \mathsf{td}^{(j)}_{\mathsf{BC}}) \leftarrow \mathsf{BC.SetupBind}(\lambda)$;
  - sample $\mathsf{rk} \leftarrow \mathsf{LRwPRF.Setup}(\lambda, \mathsf{leak})$;
  - compute $r_i \leftarrow \mathsf{LRwPRF.Eval}(\mathsf{rk}, \mathbf{z}_i) \qquad \forall i \in [m]$;
  - generate $\pi_{\mathsf{HBG}} = (\pi_{\mathsf{BARG}}, \{\mathsf{com}^{\mathsf{rk}}_j\}_{j\in[\kappa]})$ by running $\mathsf{Prove}(I, \mathsf{st})$.
- Hyb_1. This hybrid is identical to Hyb_0, except that each outer bit-commitment CRS is generated in the equivocable mode. That is, for every $j \in [\kappa]$, we sample

  $$(\widetilde{\mathsf{crs}}^{(j)}_{\mathsf{BC}}, \widetilde{\mathsf{com}}^{(j)}, \tilde{\sigma}^{(j)}_0, \tilde{\sigma}^{(j)}_1) \leftarrow \mathsf{BC.SetupEquiv}(\lambda),$$

  and use the equivocable commitment corresponding to the actual key bit rk_j in the witness of the i-th BARG instance. Since the distributions of the binding-mode CRS and the equivocable CRS are computationally indistinguishable for each bit, and the κ commitment instances are sampled independently, we have

  $$\left|\Pr[\mathsf{Hyb}_1(\mathcal{A}) = 1] - \Pr[\mathsf{Hyb}_0(\mathcal{A}) = 1]\right| \le \varepsilon(\lambda).$$

- Hyb_2. This hybrid is identical to Hyb_1, except that the unopened bits $\mathbf{r}_{\bar{I}}$ are replaced by a uniformly random string

  $$\mathbf{u} \leftarrow \{0,1\}^{|\bar{I}|}.$$

  In Hyb_1, the outer commitments to the key bits are generated in the equivocable mode and hence reveal no information about the underlying key rk. Therefore, conditioned on the public points $\mathbf{z}_1, \ldots, \mathbf{z}_m$ and the opened outputs $\mathbf{r}_I$, the only remaining key-dependent object in the adversary's view is the batch proof π_{BARG}. By assumption,

  $$|\pi_{\mathsf{BARG}}| \le \mathsf{leak}(\lambda, m).$$

  Hence, we may regard π_{BARG} as auxiliary leakage of length at most $\mathsf{leak}(\lambda, m)$ on the hidden key. By the leakage resilience of Π_{LRwPRF}, it follows that the unopened outputs remain pseudorandom even given the opened outputs $\mathbf{r}_I$ and leakage π_{BARG}. Therefore,

  $$\left|\Pr[\mathsf{Hyb}_2(\mathcal{A}) = 1] - \Pr[\mathsf{Hyb}_1(\mathcal{A}) = 1]\right| \le \varepsilon(\lambda).$$

– Hyb_3. This hybrid is identical to Hyb_2, except that $\mathsf{crs}_{\mathsf{BC}}$ is switched back from the equivocable mode to the binding mode. Since the unopened bits have already been replaced by uniform random bits, this step only changes the distribution of the outer commitments. By mode indistinguishability of Π_{BC},

$$\left|\Pr[\mathsf{Hyb}_3(\mathcal{A}) = 1] - \Pr[\mathsf{Hyb}_2(\mathcal{A}) = 1]\right| \leq \varepsilon(\lambda).$$

Observe that Hyb_3 is exactly the target distribution $(\mathsf{crs}, \mathbf{r}_I, \pi_{\mathsf{HBG}}, \mathbf{u})$

Combining the above hybrid transitions, we conclude that

$$(\mathsf{crs}, \mathbf{r}_I, \pi_{\mathsf{HBG}}, \mathbf{r}_{\bar{I}}) \approx_c (\mathsf{crs}, \mathbf{r}_I, \pi_{\mathsf{HBG}}, \mathbf{u}).$$

Therefore, Construction 5.1 satisfies computational hiding. □

F Analysis of Construction 5.2

Theorem 17 (Completeness). *If Π_{BARG} is complete and Π_{flSEH} is complete, then construction 5.2 is complete.*

Proof. Assume that

$$C(\mathbf{x}_i, \mathbf{w}_i) = 1 \qquad \forall i \in [\ell].$$

The prover honestly hashes the witnesses $W = (\mathbf{w}_1, \ldots, \mathbf{w}_\ell)$, so

$$\mathsf{flSEH.Validate}(\mathsf{vk}_{\mathsf{flSEH}}, \mathsf{hv}, \mathsf{rt}_w) = 1.$$

Moreover, all local openings of the witness bits are honestly generated and therefore accepted by flSEH.Verify. For every $i \in [\ell]$, $C(\mathbf{x}_i, \mathbf{w}_i) = 1$. Hence every pair $(\mathbf{x}'_i, \mathbf{w}'_i)$ is a valid instance-witness pair for the local-checking circuit C_{loc}.

By the completeness of Π_{BARG}, the BARG proof π_{BARG} is accepted. Therefore, the final proof $\Pi = (\mathsf{hv}, \mathsf{rt}_w, \pi_{\mathsf{BARG}})$ is accepted.

Theorem 18. *Assume that Π_{BARG} satisfies somewhere soundness and Π_{flSEH} has rate-1 somewhere extractability. The construction 5.2 has somewhere extractability.*

Proof. Let $\mathcal{A}$ be a PPT adversary in the somewhere-extractability experiment of Construction 5.2. Let $i^* \in [\ell]$ be the hidden instance index selected by the trapdoor setup of the composed construction, and define

$$I_{i^*} = \{(i^* - 1)h + 1, \ldots, i^* h\}.$$

The extractor of the composed construction computes

$$\widehat{\mathbf{w}}_{i^*} \leftarrow \mathsf{flSEH.Extract}(\mathsf{td}_{\mathsf{flSEH}}, hv),$$

where

$$\widehat{\mathbf{w}}_{i^*} \in \{0,1\}^h.$$

By the extractability of Π_{flSEH}, except with negligible probability, every collection of local openings at the positions I_{i^*} that is accepted by flSEH.Verify is consistent with $\widehat{\mathbf{w}}_{i^*}$. In particular, for every $j \in [h]$,

$$\mathsf{flSEH.Verify}\left(\mathsf{vk}_{\mathsf{flSEH}}, rt, (i^*-1)h + j, w_{i^*,j}, \rho_{i^*,j}\right) = 1$$

implies $w_{i^*,j} = \widehat{w}_{i^*,j}$. Suppose, towards a contradiction, that

$$C(\mathbf{x}_{i^*}, \widehat{\mathbf{w}}_{i^*}) = 0.$$

We show that the corresponding local-checking instance $\mathbf{x}'_{i^*}$ is false. Suppose otherwise that there exists a witness

$$\mathbf{w}'_{i^*} = \left(\mathbf{w}_{i^*}, \{\rho_{i^*,j}\}_{j\in[h]}\right)$$

such that

$$C_{\mathsf{loc}}\left(\mathbf{x}'_{i^*}, \mathbf{w}'_{i^*}\right) = 1.$$

By the definition of C_{loc}, this implies

$$\mathsf{flSEH.Verify}\left(\mathsf{vk}_{\mathsf{flSEH}}, rt, (i^*-1)h + j, w_{i^*,j}, \rho_{i^*,j}\right) = 1 \qquad \forall j \in [h].$$

Therefore, by the extractability of Π_{flSEH}, $\mathbf{w}_{i^*} = \widehat{\mathbf{w}}_{i^*}$. Moreover, C_{loc} verifies $C(\mathbf{x}_{i^*}, \mathbf{w}_{i^*}) = 1$. Consequently, $C(\mathbf{x}_{i^*}, \widehat{\mathbf{w}}_{i^*}) = 1$, which contradicts the assumption. Hence,

$$\forall \mathbf{w}'_{i^*}, \qquad C_{\mathsf{loc}}\left(\mathbf{x}'_{i^*}, \mathbf{w}'_{i^*}\right) = 0.$$

Thus, $\mathbf{x}'_{i^*}$ is a false instance of the local-checking relation. On the other hand, acceptance of the composed proof implies

$$\mathsf{BARG.Verify}\left(\mathsf{crs}_{\mathsf{BARG}}, C_{\mathsf{loc}}, \mathbf{x}'_{1:\ell}, \pi_{\mathsf{BARG}}\right) = 1.$$

Therefore, the adversary produces an accepting BARG proof for a false instance at the hidden index i^*. To reduce this event to somewhere soundness of Π_{BARG}, consider a BARG adversary $\mathcal{B}$ that receives a BARG CRS generated by the normal setup algorithm. It samples $\widetilde{i} \leftarrow [\ell]$ and generates

$$(\mathsf{hk}_{\mathsf{flSEH}}, \mathsf{vk}_{\mathsf{flSEH}}, \mathsf{td}_{\mathsf{flSEH}}) \leftarrow \mathsf{flSEH.TrapGen}\left(1^\lambda, \ell h, I_{\widetilde{i}}\right).$$

It gives $(\mathsf{hk}_{\mathsf{flSEH}}, \mathsf{vk}_{\mathsf{flSEH}}, \mathsf{crs}_{\mathsf{BARG}})$ to $\mathcal{A}$ and forwards the resulting $(C, \mathbf{x}'_{1:\ell}, \pi_{\mathsf{BARG}})$ to the BARG somewhere-soundness experiment. Let i^* be the hidden instance index sampled by that experiment. The indices i^* and $\widetilde{i}$ are independent and uniformly distributed over $[\ell]$, and therefore

$$\Pr[i^* = \widetilde{i}] = \frac{1}{\ell}.$$

Conditioned on the event $i^* = \widetilde{i}$, the preceding argument shows that, whenever the composed extractor outputs an invalid witness, the instance $\mathbf{x}'_{i^*}$ is false while the BARG proof is accepted. Consequently,

$$\mathsf{Adv}^{\mathsf{sExt}}_{\text{Construction 5.2},\mathcal{A}}(\lambda) \leq \ell \cdot \mathsf{Adv}^{\mathsf{sSound}}_{\Pi_{\mathsf{BARG}},\mathcal{B}}(\lambda, \ell, t_{\mathsf{loc}}, s_{\mathsf{loc}}) + \varepsilon(\lambda).$$

Since ℓ is polynomially bounded in λ and Π_{BARG} satisfies somewhere soundness, the right-hand side is negligible. Hence,

$$C(\mathbf{x}_{i^*}, \widehat{\mathbf{w}}_{i^*}) = 1$$

except with negligible probability. Therefore, Construction 5.2 satisfies somewhere extractability.