Efficient Zero-Knowledge Arguments from Two-Tiered Homomorphic Commitments

Jens Groth[⋆]

University College London
j.groth@ucl.ac.uk

Abstract. We construct practical and efficient zero-knowledge arguments with sublinear communication complexity. The arguments have perfect completeness, perfect special honest verifier zero-knowledge and computational soundness. Our zero-knowledge arguments rely on two-tiered homomorphic commitments for which pairing-based constructions already exist.
As a concrete application of our new zero-knowledge techniques, we look at the case of range proofs. To demonstrate a committed value belongs to a specific N-bit integer interval we only need to communicate $O(N^{\frac{1}{3}})$ group elements.

Keywords: Zero-knowledge arguments, sublinear communication, circuit satisfiability, range proofs, two-tiered homomorphic commitments.

1 Introduction

Zero-knowledge proofs introduced by Goldwasser, Micali and Rackoff [21] are fundamental building blocks in cryptography that are used in secure multi-party computation and numerous other protocols. Zero-knowledge proofs enable a prover to convince a verifier of the truth of a statement without leaking any other information. The central properties are captured in the notions of completeness, soundness and zero-knowledge.

Completeness: The prover can convince the verifier if the prover knows a witness testifying to the truth of the statement.

Soundness: A malicious prover cannot convince the verifier if the statement is false. We distinguish between computational soundness that protects against polynomial time cheating provers and statistical or perfect soundness where even an unbounded prover cannot convince the verifier of a false statement. We will call computationally sound proofs for *arguments*.

Zero-knowledge: A malicious verifier learns nothing except that the statement is true. We distinguish between computational zero-knowledge, where a polynomial time verifier learns nothing from the proof and statistical or perfect zero-knowledge, where even a verifier with unlimited resources learns nothing from the proof.

Recent works on zero-knowledge proofs [27] give us proofs with a communication complexity that grows linearly in the size of the statement to be proven and [27, 28] also give us proofs where the communication complexity depends quasi-linearly on the witness-length. These works rely on standard assumptions; if one is willing to assume the existence of fully homomorphic encryption [17] the communication complexity can be reduced to the witness-length plus a small additive overhead [16, 18].

For zero-knowledge *arguments* the communication complexity can be even lower. Kilian [29] gave a zero-knowledge argument for circuit satisfiability with polylogarithmic communication. His argument goes through the PCP-theorem [3, 2, 13] and uses a collision-free hash-function to

⋆ Supported by EPSRC grant number EP/G013829/1.

build a hash-tree that includes the entire PCP though. Even with the best PCP constructions known to date [5] Kilian's argument has high computational complexity for practical parameters. Goldwasser, Kalai and Rothblum [20] improve that state of affairs by constructing arguments that have both low communication complexity and highly efficient verification.

A large body of research starting with Schnorr's identification protocols [34] deals with zero-knowledge proofs and arguments over prime order groups. A class of zero-knowledge proofs and arguments known as Σ-protocols [9] is often used in practical applications. Groth [25] also used prime order groups to develop practical sublinear size zero-knowledge arguments for statements relating to linear algebra over $\mathbb{Z}_p$ for large primes p.

One particular example of zero-knowledge arguments that has appeared in several applications, e.g., e-voting [12] and auctions [32] are range proofs. Here the prover holds a commitment to a value w and wants to convince the verifier that the value belongs to a specific integer interval $[A; B)$. Boudot [6], Lipmaa [31] and Groth [23] have given constant size zero-knowledge argument for interval membership based on the strong RSA assumption.

In prime order groups the best range proof technique known was for a long time to commit to the bits of the value and use OR-proofs [9] to show that the committed bits were 0 or 1. For N-bit integers this communicates $O(N)$ group elements. Camenisch, Chaabouni and Shelat [7] improved this in the bilinear group setting by giving a zero-knowledge range proof with communication complexity $O(\frac{N}{\log N})$. Chaabouni, Lipmaa and Shelat [8] improved this complexity with a factor 2.

Our contribution. We construct zero-knowledge arguments for circuit satisfiability and range proofs that have perfect completeness and perfect zero-knowledge. For simplicity our constructions are in the common reference string model, but typically the common reference string can be chosen by the verifier at the cost of one extra round in the beginning to get zero-knowledge arguments in the plain model; we refer to the remarks at end of Section 2.2 for further discussion.

The circuit satisfiability argument has communication complexity $O(N^{\frac{1}{3}})$ group elements when the circuit has N gates. The range proof has a size of $O(N^{\frac{1}{3}})$ group elements for N-bit intervals. The arguments have quasi-linear computational complexity for the prover and very efficient verification. An efficiency comparison of the arguments can be found in Tables 1 and 2.

	Rounds	Comm.	Prover comp.	Verifier comp.	Assumption
Cramer et al. [9]	3	$O(N)$ G	$O(N)$ E	$O(N)$ E	Dlog
Groth [25]	5	$O(N^{\frac{1}{2}})$ G	$O(N \log^2 N)$ M	$O(N)$ M	DLog
This paper	7	$O(N^{\frac{1}{3}})$ G	$O(N \log^2 N)$ M	$O(N)$ M	DPair

Table 1. Zero-knowledge arguments for satisfiability of circuits with N NAND-gates measured in group elements G, exponentiations E, and multiplications M.

	Rounds	Comm.	Prover comp.	Verifier comp.	Assumption
Camenisch et al. [7]	3	$O(\frac{N}{\log N})$ G	$O(\frac{N}{\log N})$ E	$O(\frac{N}{\log N})$ E	q-SDH
Chaabouni et al [8]	3	$O(\frac{N}{\log N})$ G	$O(\frac{N}{\log N})$ E	$O(\frac{N}{\log N})$ E	q-SDH
This paper	7	$O(N^{\frac{1}{3}})$ G	$O(N \log^2 N)$ M	$O(N^{\frac{1}{3}})$ M	DPair

Table 2. Range proofs in prime order groups measured in group elements G, exponentiations E, and multiplications M.

In the tables we give the conservative estimate of $O(N \log^2 N)$ estimate for the prover's computation, but as we will discuss at the end of Section 3 it can often be reduced to $O(N \log N)$ using Fast Fourier Transform techniques. When comparing the range proofs, we are assuming a common reference string is available. This permits the incorporation of the initial messages in [7, 8] into the common reference string such that their range proofs only use 3 rounds instead of 4 rounds.

Our zero-knowledge arguments can be instantiated in asymmetric bilinear groups where the computational double pairing assumption (Section 2.1) holds. In comparison, the range proofs [7, 8] are based on the q-SDH assumption in bilinear groups.

Techniques. Our main technical contribution is the batch product argument that can be found in Section 3. Using homomorphic commitments to group elements [1, 25] we can in combination with Pedersen commitments to multiple elements commit to N elements in $\mathbb{Z}_p$ using only $N^{\frac{1}{3}}$ group elements. Given $3N$ committed elements $u_i, v_i, w_i \in \mathbb{Z}_p$ we generalize techniques from [26, 25] to develop a communication-efficient zero-knowledge argument for proving that the committed values all satisfy $u_i v_i = w_i$.

Since the commitments are homomorphic we can now do both additions and multiplications on the committed elements. This enables the prover to commit to the wires in a circuit and prove that they respect the NAND-gates.

For the range proof we commit to the bits $w_1, \ldots, w_N$ of the committed value. Using the batch product argument we can show with a communication complexity of $O(N^{\frac{1}{3}})$ group elements that the committed bits satify $w_i w_i = w_i$, which can only be true if $w_i \in \{0, 1\}$. Once we have the committed bits, we can then use the homomorphic properties of the commitment schemes to compute $w = \sum_{i=1}^{N} w_i 2^{i-1}$. This shows that w belongs to the range $[0; 2^N)$ and can be generalized to a range of the form $[A; B)$.

2 Preliminaries

We write $y = A(x; r)$ when the algorithm A on input x and randomness r, outputs y. We write $y \leftarrow A(x)$ for the process of picking randomness r at random and setting $y = A(x; r)$. We also write $y \leftarrow S$ for sampling y uniformly at random from the set S.

We will give a security parameter λ written in unary as input to all parties in our protocols. Intuitively, the higher the security parameter the more secure the protocol. We say a function $f : \mathbb{N} \to [0, 1]$ is negligible if $f(\lambda) = O(\lambda^{-c})$ for every constant $c > 0$. We write $f \approx g$ when $|f(\lambda) - g(\lambda)|$ is negligible. We say f is overwhelming if $f \approx 1$.

2.1 Two-tiered homomorphic commitments

A commitment scheme allows Alice to compute and send a commitment to a secret message a. Later Alice may open the commitment and reveal to Bob that she committed to a. Commitments must be binding and hiding. Binding means that Alice cannot change her mind; a commitment can only be opened to one message a. Hiding means that Bob does not learn which message Alice committed to.

In the Pedersen commitment scheme [33] the public key contains the description of a group of prime order p and group elements g, h. A commitment to $a \in \mathbb{Z}_p$ is constructed by picking $r \leftarrow \mathbb{Z}_p$ and computing $c = g^a h^r$. This commitment scheme is very useful because it is homomorphic, i.e., the product of two commitments is $c \cdot c' = (g^a h^r)(g^b h^s) = g^{a+b} h^{r+s}$, which is a commitment to $a+b$. The Pedersen commitment can be generalized such that the public key contains $g_1, \ldots, g_n, h$ and a commitment to $(a_1, \ldots, a_n) \in \mathbb{Z}_p^n$ is computed as $h^r \prod_{k=1}^{n} g_k^{a_k}$.

Abe, Fuchsbauer, Groth, Haralambiev and Ohkubo [1, 24] proposed commitment schemes for group elements. One of the commitment schemes uses a bilinear group with a pairing $e : \mathbb{G} \times \hat{\mathbb{G}} \to \mathbb{T}$. Here $\mathbb{G}, \hat{\mathbb{G}}, \mathbb{T}$ are cyclic groups of prime order p where we call $\mathbb{G}, \hat{\mathbb{G}}$ the base groups and $\mathbb{T}$ the target group. The pairing is efficiently computable, non-trivial and bilinear, i.e., for all x, y, a, b we have $e(x^a, y^b) = e(x, y)^{ab}$. The commitment scheme specifies non-trivial group elements $v, u_1, \ldots, u_m \in \hat{\mathbb{G}}$ and a commitment to $(c_1, \ldots, c_m) \in \mathbb{G}$ is computed by picking at random $t \in \mathbb{G}$ and computing $C = e(t, v) \prod_{j=1}^{m} e(c_j, u_j)$. The commitment scheme is computationally binding under the computational double pairing assumption, which states that given random $u, v \in \hat{\mathbb{G}}$ it is hard to find non-trivial $s, t \in \mathbb{G}$ such that $e(s, u) = e(t, v)$. The hardness of the computational double pairing assumption is implied by the decision Diffie-Hellman assumption in $\hat{\mathbb{G}}$ [1, 24].[1] Furthermore, the bilinearity of the pairing means that the commitment scheme is homomorphic in the sense that

$$C \cdot C' = \left(e(t, v) \prod_{j=1}^{m} e(c_j, u_j) \right) \left(e(t', v) \prod_{j=1}^{m} e(c'_j, u_j) \right) = e(tt', v) \prod_{j=1}^{m} e(c_j c'_j, u_j)$$

is a commitment to the entry-wise product of the messages.

Combining the two types of commitment schemes it is possible to commit to commitments. If we compute $c_j = h^{r_j} \prod_{k=1}^{n} g_k^{a_{jk}}$ and $C = e(t, v) \prod_{j=1}^{m} e(c_j, u_j)$ we have a single target group element that is a commitment to mn values $\{a_{jk}\}_{j=1,k=1}^{m,n}$. Since both commitment schemes are homomorphic the product of two commitments $C \cdot C'$ is a commitment to the sums of the messages $a_{jk} + a'_{jk}$. In our zero-knowledge arguments both the homomorphic and the length-reducing properties will be crucial because it allows the prover to do computations on committed values in a verifiable manner and with little communication.

The commitment schemes described above provide an example of what we will call a two-tiered commitment scheme. With the Pedersen commitment scheme in mind we will for simplicity assume the randomness is drawn from $\mathbb{Z}_p$ but it would be easy to generalize to other randomizer spaces. Furthermore, in the example given above the Pedersen commitments are perfectly hiding and we can therefore use trivial randomness $t = 1$ in the commitments to Pedersen commitments. This observation is incorporated in the following definition of a two-tiered commitment scheme.

A two-tiered commitment scheme consists of three polynomial time algorithms $(\mathcal{K}, \mathrm{com}, \mathrm{com}^{(2)})$. $\mathcal{K}$ is a key generator that on security parameter λ and integers m, n returns a public key ck. The commitment key specifies cyclic groups $\mathbb{Z}_p$, $\mathbb{G}$ and $\mathbb{T}$ of prime order p. It also specifies how to efficiently compute $\mathrm{com}_{ck} : \mathbb{Z}_p^n \times \mathbb{Z}_p \to \mathbb{G}$ and $\mathrm{com}_{ck}^{(2)} : \mathbb{G}^m \to \mathbb{T}$.

Definition 1 (Homomorphic). *We say the two-tiered commitment scheme is homomorphic, when the maps* com_{ck} *and* $\mathrm{com}_{ck}^{(2)}$ *are* $\mathbb{Z}_p$*-linear.*

Definition 2 (Computationally binding). *The two-tiered commitment scheme* $(\mathcal{K}, \mathrm{com}, \mathrm{com}^{(2)})$ *is computationally binding if for all non-uniform polynomial time adversaries* $\mathcal{A}$ *and for all* $m, n = \lambda^{O(1)}$

$$\Pr\Big[ck \leftarrow \mathcal{K}(1^\lambda, m, n); (\boldsymbol{a}, \boldsymbol{b}, r, s, \boldsymbol{c}, \boldsymbol{d}) \leftarrow \mathcal{A}(ck) : \ \boldsymbol{a} \neq \boldsymbol{b} \in \mathbb{Z}_p^n \ r, s \in \mathbb{Z}_p \ \boldsymbol{c} \neq \boldsymbol{d} \in \mathbb{G}^m$$
$$\mathrm{com}_{ck}(\boldsymbol{a}; r) = \mathrm{com}_{ck}(\boldsymbol{b}; s) \quad \text{or} \quad \mathrm{com}_{ck}^{(2)}(\boldsymbol{c}) = \mathrm{com}_{ck}^{(2)}(\boldsymbol{d})\Big] \approx 0.$$

[1] Galbraith, Paterson and Smart [14] classified bilinear groups into 3 types. The commitment scheme described above uses type II or type III bilinear groups. In a type I bilinear group we could instead use the decisional linear assumption based commitment scheme from [24].

Definition 3 (Perfectly hiding). *The two-tiered commitment scheme* $(\mathcal{K}, \mathrm{com}, \mathrm{com}^{(2)})$ *is perfectly hiding if for all stateful adversaries* $\mathcal{A}$ *and all* $m, n \in \lambda^{O(1)}$

$$\Pr\left[ck \leftarrow \mathcal{K}(1^\lambda, m, n); \boldsymbol{a}_0, \boldsymbol{a}_1 \leftarrow \mathbb{Z}_p^n; b \leftarrow \{0,1\}; c \leftarrow \mathrm{com}_{ck}(\boldsymbol{a}_b) : \mathcal{A}(ck, \boldsymbol{a}_0, \boldsymbol{a}_1, c) = b\right] = \frac{1}{2}.$$

The zero-knowledge arguments we describe will work over any two-tiered homomorphic commitment scheme with a large prime p. When giving concrete efficiency estimates we will assume we are using the bilinear group based scheme described earlier in this section. The public key for this commitment scheme consists of a description of a bilinear group $(p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{T}, e)$ and $m+n+2$ group elements in $\mathbb{G}$ and $\hat{\mathbb{G}}$. We will be looking at statements of size N and the minimal communication complexity will be obtained when $m = O(N^{\frac{1}{3}})$ and $n = O(N^{\frac{1}{3}})$ giving a public key size of $O(N^{\frac{1}{3}})$ group elements.

2.2 Special honest verifier zero-knowledge arguments of knowledge

We will for simplicity describe how our arguments work in the common reference string model and how to obtain zero-knowledge against honest-but-curious verifiers. Both of these restrictions can be removed at very small cost to get full zero-knowledge in the plain model as described in the remarks at the end.

Consider a triple of probabilistic polynomial time interactive algorithms $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ called the common reference string generator, the prover and the verifier. The common reference string generator takes the security parameter λ as input in unary and some auxilliary input m, n that specifies the size of the statements we are interested in and generates a common reference string. In the zero-knowledge arguments in this paper, the common reference string will contain the public key ck for a two-tiered commitment scheme.

Let R be a polynomial time decidable ternary relation. For a statement x we call w a witness if $(ck, x, w) \in R$. We define a corresponding common reference string dependent language L_{ck} consisting of statements x that have a witness w such that $(ck, x, w) \in R$. This is a natural generalization of NP-languages; when R ignores ck we have the standard notion of an NP-language.

We write $\mathrm{tr} \leftarrow \langle \mathcal{P}(s), \mathcal{V}(t) \rangle$ for the public transcript produced by $\mathcal{P}$ and $\mathcal{V}$ when interacting on inputs s and t. This transcript ends with $\mathcal{V}$ either accepting or rejecting. We sometimes shorten the notation by saying $\langle \mathcal{P}(s), \mathcal{V}(t) \rangle = b$, where $b = 0$ corresponds to $\mathcal{V}$ rejecting and $b = 1$ corresponds to $\mathcal{V}$ accepting.

Definition 4 (Argument). *The triple* $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ *is an* argument *for relation* R *with perfect completeness if for all non-uniform polynomial time interactive adversaries* $\mathcal{A}$ *and all* $m, n = \lambda^{O(1)}$ *we have*

Perfect completeness:

$$\Pr\left[ck \leftarrow \mathcal{K}(1^\lambda, m, n); (x, w) \leftarrow \mathcal{A}(ck) : (ck, x, w) \notin R \text{ or } \langle \mathcal{P}(ck, x, w), \mathcal{V}(ck, x) \rangle = 1\right] = 1.$$

Computational soundness:

$$\Pr\left[ck \leftarrow \mathcal{K}(1^\lambda, m, n); x \leftarrow \mathcal{A}(ck) : x \notin L_{ck} \text{ and } \langle \mathcal{A}, \mathcal{V}(ck, x) \rangle = 1\right] \approx 0.$$

Definition 5 (Public coin argument). *An argument* $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ *is* public coin *if the verifier's messages are chosen uniformly at random independently of the messages sent by the prover.*

We shall define an argument of knowledge through witness-extended emulation [22, 30]. Informally, the definition says: given an adversary that produces an acceptable argument with probability ϵ, there exists an emulator that produces a similar argument with roughly the same probability ϵ and at the same time provides a witness.

Definition 6 (Witness-extended emulation). *We say the public coin argument $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ has computational witness-extended emulation if for all deterministic polynomial time $\mathcal{P}^*$ there exists an expected polynomial time emulator $\mathcal{X}$ such that for all non-uniform polynomial time adversaries $\mathcal{A}$ and all $m, n = \lambda^{O(1)}$*

$$\Pr\Big[ck \leftarrow \mathcal{K}(1^\lambda, m, n); (x, s) \leftarrow \mathcal{A}(ck); \mathrm{tr} \leftarrow \langle \mathcal{P}^*(ck, x, s), \mathcal{V}(ck, x)\rangle : \mathcal{A}(\mathrm{tr}) = 1\Big]$$
$$\approx \Pr\Big[ck \leftarrow \mathcal{K}(1^\lambda, m, n); (x, s) \leftarrow \mathcal{A}(ck); (\mathrm{tr}, w) \leftarrow \mathcal{X}^{\langle \mathcal{P}^*(ck,x,s), \mathcal{V}(ck,x)\rangle}(ck, x) :$$
$$\mathcal{A}(\mathrm{tr}) = 1 \text{ and if tr is accepting then } (ck, x, w) \in R\Big],$$

where $\mathcal{X}$ has access to a transcript oracle $\langle \mathcal{P}^*(ck, x, s), \mathcal{V}(ck, x)\rangle$ that can be rewound to a particular round and run again with the verifier using fresh randomness.

We think of s as being the state of $\mathcal{P}^*$, including the randomness. Then we have an argument of knowledge in the sense that the emulator can extract a witness whenever $\mathcal{P}^*$ is able to make a convincing argument. This shows that the definition implies soundness. We remark that the verifier's randomness is part of the transcript and the prover is deterministic. So combining the emulated transcript with ck, x, s gives us the view of both the prover and the verifier and at the same time gives us the witness.

Please note that the standard definition of *proofs* of knowledge by Bellare and Goldreich [4] does not apply in our setting, since we work in the common reference string model and are interested in *arguments* of knowledge; see Damgård and Fujisaki [11] for a discussion of this issue and an alternative definition of knowledge soundness. Witness-extended emulation implies knowledge soundness as defined by Damgård and Fujisaki [22].

We define special honest verifier zero-knowledge (SHVZK) [9] for a public coin argument as the ability to simulate the transcript for any set of challenges without access to the witness.

Definition 7 (Perfect special honest verifier zero-knowledge). *The public coin argument $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ is a perfect special honest verifier zero-knowledge argument for R if there exists a probabilistic polynomial time simulator $\mathcal{S}$ such that for all non-uniform polynomial time adversaries $\mathcal{A}$ and all $m, n = \lambda^{O(1)}$*

$$\Pr\Big[ck \leftarrow \mathcal{K}(1^\lambda, m, n); (x, w, \rho) \leftarrow \mathcal{A}(ck); \mathrm{tr} \leftarrow \langle \mathcal{P}(ck, x, w), \mathcal{V}(ck, x; \rho)\rangle :$$
$$(ck, x, w) \in R \text{ and } \mathcal{A}(\mathrm{tr}) = 1\Big]$$
$$= \Pr\Big[ck \leftarrow \mathcal{K}(1^\lambda, m, n); (x, w, \rho) \leftarrow \mathcal{A}(ck); \mathrm{tr} \leftarrow \mathcal{S}(ck, x, \rho) : (ck, x, w) \in R \text{ and } \mathcal{A}(\mathrm{tr}) = 1\Big].$$

The plain model. We will describe our arguments in the common reference string model where the prover and verifier have a trusted setup. If we want to work in the plain model we can add an initial round where the verifier picks the common reference string and sends it to the prover. Provided it can be verified that the verifier's initial message describes a valid common reference string this will still be perfect SHVZK because we do not rely on the simulator knowing any trapdoor information associated with the common reference string.

Full zero-knowledge. For simplicity, we focus on SHVZK arguments in this paper. There are very efficient standard techniques [10, 15, 22] to convert an SHVZK argument into a public-coin full zero-knowledge argument with a cheating verifier when a common reference string is available.

If we work in the plain model and let the verifier choose the common reference string, we can use coin-flipping techniques (for the full zero-knowledge property the coin-flips should be simulatable against a dishonest verifier) for the challenges to get private-coin[2] full zero-knowledge arguments against a cheating verifier. Challenges in our SHVZK arguments are very short so both in the case with and without a common reference string the overhead of getting full zero-knowledge is insignificant compared to the cost of the SHVZK arguments.

3 Batch Product Argument

We will now present our main technical contribution, which is a batch product argument for committed values $\{u_{ijk}, v_{ijk}, w_{ijk}\}_{i=1,j=1,k=1}^{M,m,n}$ satisfying $u_{ijk}v_{ijk} = w_{ijk}$. More precisely, the statement consists of commitments $C_{U_1}, C_{V_1}, C_{W_1}, \ldots, C_{U_M}, C_{V_M}, C_{W_M} \in \mathbb{T}$. The prover will argue knowledge of openings $u_{ijk}, r_{ij}, v_{ijk}, s_{ij}, w_{ijk}, t_{ij} \in \mathbb{Z}_p$ such that

$$\begin{aligned} c_{u_{ij}} &= \mathrm{com}_{ck}(u_{ij1}, \ldots, u_{ijn}; r_{ij}) & C_{U_i} &= \mathrm{com}_{ck}^{(2)}(c_{u_{i1}}, \ldots, c_{u_{im}}) \\ c_{v_{ij}} &= \mathrm{com}_{ck}(v_{ij1}, \ldots, v_{ijn}; s_{ij}) & C_{V_i} &= \mathrm{com}_{ck}^{(2)}(c_{v_{i1}}, \ldots, c_{v_{im}}) \\ c_{w_{ij}} &= \mathrm{com}_{ck}(w_{ij1}, \ldots, w_{ijn}; t_{ij}) & C_{W_i} &= \mathrm{com}_{ck}^{(2)}(c_{w_{i1}}, \ldots, c_{w_{im}}) \end{aligned}$$
$$u_{ijk}v_{ijk} = w_{ijk}.$$

The argument will have communication complexity $O(M + m + n)$. In order to explain the idea behind the argument let us first focus on soundness and for now postpone the question of how to get SHVZK. In the argument, the prover will demonstrate that she knows openings of $C_{U_i}, C_{V_i}, C_{W_i}$ to $c_{u_{ij}}, c_{v_{ij}}, c_{w_{ij}}$ and that she knows openings of $c_{u_{ij}}, c_{v_{ij}}, c_{w_{ij}}$ using standard techniques. She will also know openings $a_\alpha, \rho_\alpha, b_\beta, \sigma_\beta \in \mathbb{Z}_p$ of intermediate commitments $c_{a_\alpha} = \mathrm{com}_{ck}(a_\alpha; \rho_\alpha), c_{b_\beta} = \mathrm{com}_{ck}(b_\beta, \sigma_\beta)$ that she sends during the argument and which will be specified later. The argument runs over 7 moves with the prover getting challenges $x, y, z \in \mathbb{Z}_p^*$ in round 2, 4 and 6. The commitments c_{a_α} are sent in round 3 and the commitments c_{b_β} are sent in round 5. This means a_α may depend on x but is independent of y and z, and b_β may depend on both x and y but is independent of z.

The prover will demonstrate to the verifier that

$$\sum_{i=1}^{M}\sum_{j=1}^{m}\sum_{k=1}^{n}(u_{ijk}v_{ijk} - w_{ijk})x^{i(m+1)n+jn+k} = 0. \tag{1}$$

Unless $u_{ijk}v_{ijk} = w_{ijk}$ for all choices of i, j, k this has negligible probability of holding over a randomly chosen challenge $x \in \mathbb{Z}_p^*$. Our main obstacle is to build up this polynomial and convince the verifier that the equality (1) holds true using only $O(M + m + n)$ communication.

We carefully choose appropriate linear combinations of the commitments and by the homomorphic property get corresponding linear combinations of the $u_{ijk}, v_{ijk}, w_{ijk}$ values such that the equality (1) emerges. During this process, we will also use exponentiations of some of the commitments to powers of x such that we get linear combinations of $u_{ijk}x^{i(m+1)n+jn+k}$ and

[2] Goldreich and Krawczyk [19] have shown that only languages in BPP have constant-round public-coin arguments.

$w_{ijk}x^{i(m+1)n+jn+k}$. Suppose for instance that the prover after seeing x computes and opens

$$\prod_{i=1}^{M} C_{U_i}^{x^{i(m+1)n}} = \mathrm{com}^{(2)}_{ck}(c_{u_1},\ldots,c_{u_m}) \qquad \text{where} \qquad c_{u_j} = \prod_{i=1}^{M} c_{u_{ij}}^{x^{i(m+1)n}}$$
$$\prod_{i=1}^{M} C_{V_i} = \mathrm{com}^{(2)}_{ck}(c_{v_1},\ldots,c_{v_m}) \qquad \text{where} \qquad c_{v_j} = \prod_{i=1}^{M} c_{v_{ij}}$$
$$\prod_{i=1}^{M} C_{W_i}^{x^{i(m+1)n}} = \mathrm{com}^{(2)}_{ck}(c_{w_1},\ldots,c_{w_m}) \qquad \text{where} \qquad c_{w_j} = \prod_{i=1}^{M} c_{w_{ij}}^{x^{i(m+1)n}}$$

and at the same time computes and opens

$$\prod_{j=1}^{m} c_{u_j}^{x^{jn}} = \mathrm{com}_{ck}(u_1,\ldots,u_n;r) \qquad \text{where} \qquad u_k = \sum_{i=1}^{M}\sum_{j=1}^{m} u_{ijk}x^{i(m+1)n+jn}$$
$$\prod_{j=1}^{m} c_{v_j} = \mathrm{com}_{ck}(v_1,\ldots,v_n;s) \qquad \text{where} \qquad v_k = \sum_{i=1}^{M}\sum_{j=1}^{m} v_{ijk}$$
$$\prod_{j=1}^{m} c_{w_j}^{x^{jn}} = \mathrm{com}_{ck}(w_1,\ldots,w_n;t) \qquad \text{where} \qquad w_k = \sum_{i=1}^{M}\sum_{j=1}^{m} w_{ijk}x^{i(m+1)n+jn}$$

Using only $3m$ commitments and $3m+3$ elements in $\mathbb{Z}_p$ this tells the verifier

$$u_kx^k = \sum_{i=1}^{M}\sum_{j=1}^{m} u_{ijk}x^{i(m+1)n+jn+k} \qquad v_k = \sum_{i=1}^{M}\sum_{j=1}^{m} v_{ijk}$$

$$w_kx^k = \sum_{i=1}^{M}\sum_{j=1}^{m} w_{ijk}x^{i(m+1)n+jn+k}.$$

We now have that

$$\sum_{k=1}^{n}(u_kv_k - w_k)x^k$$
$$= \sum_{k=1}^{n}\left((\sum_{i=1}^{M}\sum_{j=1}^{m} u_{ijk}x^{i(m+1)n+jn+k})(\sum_{i'=1}^{M}\sum_{j'=1}^{m} v_{i'j'k}) - \sum_{i=1}^{M}\sum_{j=1}^{m} w_{ijk}x^{i(m+1)n+jn+k} \right)$$

contains the desired polynomial from (1) but there are some cross-terms corresponding to $i \neq i'$ or $j \neq j'$ so the polynomial given above may be non-zero.

We will choose the a_α and b_β values such that they cancel out the cross-terms. However, we have to be careful that there are only $O(M+m+n)$ of them and that they are feasible to compute. We will therefore use an interactive technique that will enable the verifier to pick a_α and b_α after seeing x. This introduces a second concern, namely to choose them in a way such that they do not affect the original equality we wish to get. We accomplish this by making sure that a_α and b_β are modified by factors y^α and z^β for $\alpha, \beta \neq 0$ while the desired equality does not contain any such factors. To make this happen we will modify the opening process of the commitments C_{U_i} and C_{V_i} described above to open

$$\prod_{i=1}^{M} C_{U_i}^{x^{i(m+1)n}y^i} = \mathrm{com}_{ck}^{(2)}(c_{u_1},\ldots,c_{u_m}) \qquad \prod_{j=1}^{m} c_{u_j}^{x^{jn}z^j} = \mathrm{com}_{ck}(u_1,\ldots,u_n;r)$$

$$\prod_{i=1}^{M} C_{V_i}^{y^{-i}} = \mathrm{com}_{ck}^{(2)}(c_{u_1},\ldots,c_{u_m}) \qquad \prod_{j=1}^{m} c_{v_j}^{z^{-j}} = \mathrm{com}_{ck}(v_1,\ldots,v_n;r)$$

This gives us

$$u_k x^k = \sum_{i=1}^{M}\sum_{j=1}^{m} u_{ijk}x^{i(m+1)n+jn+k}y^i z^j \qquad\qquad v_k = \sum_{i=1}^{M}\sum_{j=1}^{m} v_{ijk}y^{-i}z^{-j}.$$

We now have

$$\begin{aligned}\sum_{k=1}^{n} u_k x^k v_k &= \sum_{k=1}^{n}(\sum_{i=1}^{M}\sum_{j=1}^{m} u_{ijk}x^{i(m+1)n+jn+k}y^i z^j)(\sum_{i'=1}^{M}\sum_{j'=1}^{m} v_{i'j'k}y^{-i'}z^{-j'})\\ &= \sum_{k=1}^{n}\sum_{i=1}^{M}\sum_{i'=1}^{M}\sum_{j=1}^{m}\sum_{j'=1}^{m} u_{ijk}x^{i(m+1)n+jn+k}v_{i'j'k}y^{i-i'}z^{j-j'}\end{aligned}$$

By splitting the sum into three parts corresponding to the three cases $j=j', i=i'$ and $j=j', i\neq i'$ and $j\neq j'$ and subtracting the $w_k x^k$'s we get

$$\begin{aligned}\sum_{k=1}^{n}(u_k v_k - w_k)x^k &= \sum_{k=1}^{n}\sum_{i=1}^{M}\sum_{j=1}^{m}(u_{ijk}v_{ijk}-w_{ijk})x^{i(m+1)n+jn+k}\\ &\quad + \sum_{k=1}^{n}\sum_{i=1}^{M}\sum_{\substack{i'=1\\ i'\neq i}}^{M}\sum_{j=1}^{m} u_{ijk}x^{i(m+1)n+jn+k}v_{i'jk}y^{i-i'}\\ &\quad + \sum_{k=1}^{n}\sum_{i=1}^{M}\sum_{i'=1}^{M}\sum_{j=1}^{m}\sum_{\substack{j'=1\\ j'\neq j}}^{m} u_{ijk}x^{i(m+1)n+jn+k}v_{i'j'k}y^{i-i'}z^{j-j'} \qquad (2)\\ &= \sum_{k=1}^{n}\sum_{i=1}^{M}\sum_{j=1}^{m}(u_{ijk}v_{ijk}-w_{ijk})x^{i(m+1)n+jn+k}\\ &\quad + \sum_{\substack{\alpha=-M\\ \alpha\neq 0}}^{M}\sum_{\substack{i=1,i'=1\\ i-i'=\alpha}}^{M,M}\sum_{k=1}^{n}\sum_{j=1}^{m} u_{ijk}x^{i(m+1)n+jn+k}v_{i'jk}y^{\alpha}\\ &\quad + \sum_{\substack{\beta=-m\\ \beta\neq 0}}^{m}\sum_{\substack{j=1,j'=1\\ j-j'=\beta}}^{m,m}\sum_{k=1}^{n}(\sum_{i=1}^{M} u_{ijk}x^{i(m+1)n+jn+k}y^i)(\sum_{i'=1}^{M} v_{i'j'k}y^{-i'})z^{\beta}\end{aligned}$$

The prover will select

$$a_\alpha = \sum_{\substack{i=1,i'=1\\ i-i'=\alpha}}^{M,M}\sum_{k=1}^{n}\sum_{j=1}^{m} u_{ijk}x^{i(m+1)n+jn+k}v_{i'jk}$$

$$b_\beta = \sum_{\substack{j=1,j'=1\\ j-j'=\beta}}^{m,m} \sum_{k=1}^{n} (\sum_{i=1}^{M} u_{ijk} x^{i(m+1)n+jn+k} y^i)(\sum_{i'=1}^{M} y^i v_{i'j'k} y^{-i'})$$

and send the commitments $\{c_{a_\alpha}\}_\alpha$ before seeing y and send $\{c_{b_\beta}\}_\beta$ before seeing z. She will reveal randomness $R \in \mathbb{Z}_p$ such that

$$\prod_{\substack{\alpha=-M\\ \alpha\neq 0}}^{M} c_{a_\alpha}^{y^\alpha} \cdot \prod_{\substack{\beta=-m\\ \beta\neq 0}}^{m} c_{b_\beta}^{z^\beta} = \mathrm{com}_{ck}(\sum_{k=1}^{n}(u_k v_k - w_k)x^k; R).$$

This corresponds to the values in the commitments satisfying

$$\sum_{\substack{\alpha=-M\\ \alpha\neq 0}}^{M} a_\alpha y^\alpha + \sum_{\substack{\beta=-m\\ \beta\neq 0}}^{m} b_\beta z^\beta = \sum_{k=1}^{n}(u_k v_k - w_k)x^k.$$

Keeping in mind the expansion of the right hand side (2) we get that with overwhelming probability over y, z this can only be true if equation (1) holds.

In order to make the protocol SHVZK we add some commitments and values such that $c_{u_j}, c_{v_j}, c_{w_j}$ and u_k, v_k, w_k cannot reveal anything about $u_{ijk}, v_{ijk}, w_{ijk}$. Furthermore, we add some d_k values and c_{d_k} commitments to cancel out new cross-terms arising from the added values. This gives us the full batch product argument below.

Common reference string: Two-tiered commitment key ck.
Statement: Commitments $C_{U_1}, C_{V_1}, C_{W_1} \ldots, C_{U_M}, C_{V_M}, C_{W_M} \in \mathbb{T}$.
Prover's witness: Values $u_{111}, v_{111}, w_{111}, \ldots, u_{Mmn}, v_{Mmn}, w_{Mmn} \in \mathbb{Z}_p$ and randomness $r_{11}, s_{11}, t_{11}, \ldots, r_{Mm}, s_{Mm}, t_{Mm} \in \mathbb{Z}_p$ such that for all $i \in \{1, \ldots, M\}, j \in \{1, \ldots, m\}, k \in \{1, \ldots, n\}$:

$$\begin{aligned} c_{u_{ij}} &= \mathrm{com}_{ck}(u_{ij1}, \ldots, u_{ijn}; r_{ij}) \qquad & C_{U_i} &= \mathrm{com}_{ck}^{(2)}(c_{u_{i1}}, \ldots, c_{u_{im}}) \\ c_{v_{ij}} &= \mathrm{com}_{ck}(v_{ij1}, \ldots, v_{ijn}; s_{ij}) \qquad & C_{V_i} &= \mathrm{com}_{ck}^{(2)}(c_{v_{i1}}, \ldots, c_{v_{im}}) \\ c_{w_{ij}} &= \mathrm{com}_{ck}(w_{ij1}, \ldots, w_{ijn}; t_{ij}) \qquad & C_{W_i} &= \mathrm{com}_{ck}^{(2)}(c_{w_{i1}}, \ldots, c_{w_{im}}) \end{aligned}$$
$$u_{ijk} v_{ijk} = w_{ijk}.$$

1. $\mathcal{P} \to \mathcal{V}$**:** Pick $u_{00k}, v_{00k}, w_{00k} \leftarrow \mathbb{Z}_p$ and set $u_{0jk} = v_{0jk} = w_{0jk} = 0$ and $u_{i0k} = v_{i0k} = w_{i0k} = 0$ for $i \neq 0$ and $j \neq 0$. Pick $r_{00}, s_{00}, t_{00}, \tau_1, \ldots, \tau_n \leftarrow \mathbb{Z}_p$ and pick $r_{0j}, s_{0j}, t_{0j} \leftarrow \mathbb{Z}_p$. Compute for $j \in \{0, \ldots, m\}$ and $k \in \{1, \ldots, n\}$

$$\begin{aligned} c_{u_{0j}} &= \mathrm{com}_{ck}(u_{0j1}, \ldots, u_{0jn}; r_{0j}) \qquad & C_{U_0} &= \mathrm{com}_{ck}^{(2)}(c_{u_{01}}, \ldots, c_{u_{0m}}) \\ c_{v_{0j}} &= \mathrm{com}_{ck}(v_{0j1}, \ldots, v_{0jn}; s_{0j}) \qquad & C_{V_0} &= \mathrm{com}_{ck}^{(2)}(c_{v_{01}}, \ldots, c_{v_{0m}}) \\ c_{w_{0j}} &= \mathrm{com}_{ck}(w_{0j1}, \ldots, w_{0jn}; t_{0j}) \qquad & C_{W_0} &= \mathrm{com}_{ck}^{(2)}(c_{w_{01}}, \ldots, c_{w_{0m}}) \\ d_k &= u_{00k} v_{00k} - w_{00k} \qquad & c_{d_k} &= \mathrm{com}_{ck}(d_k; \tau_k) \end{aligned}$$

Send: $c_{u_{00}}, c_{v_{00}}, c_{w_{00}}, C_{U_0}, C_{V_0}, C_{W_0}, \{c_{d_k}\}_{k=1}^n$.
2. $\mathcal{P} \leftarrow \mathcal{V}$**:** $x \leftarrow \mathbb{Z}_p^*$.

3. $\mathcal{P} \to \mathcal{V}$**:** For $\alpha \in \{-M, \ldots, -1, 1, \ldots, M\}$ pick $\rho_\alpha \leftarrow \mathbb{Z}_p$ and compute

$$a_\alpha = \sum_{\substack{i=0,i'=0\\ i-i'=\alpha}}^{M,M} \sum_{j=0}^{m} \sum_{k=1}^{n} (u_{ijk} x^{i(m+1)n+jn+k}) v_{i'jk} \qquad c_{a_\alpha} = \mathrm{com}_{ck}(a_\alpha; \rho_\alpha).$$

Send: $\{c_{a_\alpha}\}_{\alpha \in \{-M,\ldots,-1,1,\ldots,M\}}$.

4. $\mathcal{P} \leftarrow \mathcal{V}$**:** $y \leftarrow \mathbb{Z}_p^*$.

5. $\mathcal{P} \to \mathcal{V}$**:** For $\beta \in \{-m, \ldots, -1, 1, \ldots, m\}$ pick $\sigma_\beta \leftarrow \mathbb{Z}_p$ and compute

$$b_\beta = \sum_{\substack{j=0,j'=0\\ j-j'=\beta}}^{m,m} \sum_{k=1}^{n} \left(\sum_{i=0}^{M} u_{ijk} x^{i(m+1)n+jn+k} y^i\right) \left(\sum_{i'=0}^{M} v_{i'j'k} y^{-i'}\right) \qquad c_{b_\beta} = \mathrm{com}_{ck}(b_\beta; \sigma_\beta).$$

Compute also for $j \in \{1, \ldots, m\}$

$$c_{u_j} = \prod_{i=0}^{M} c_{u_{ij}}^{x^{i(m+1)n} y^i} \qquad c_{v_j} = \prod_{i=0}^{M} c_{v_{ij}}^{y^{-i}} \qquad c_{w_j} = \prod_{i=0}^{M} c_{w_{ij}}^{x^{i(m+1)n}}.$$

Send: $\{c_{b_\beta}\}_{\beta \in \{-m,\ldots,-1,1,\ldots,m\}}, \{c_{u_j}, c_{v_j}, c_{w_j}\}_{j=1}^m$.

6. $\mathcal{P} \leftarrow \mathcal{V}$**:** $z \leftarrow \mathbb{Z}_p^*$.

7. $\mathcal{P} \to \mathcal{V}$**:** Compute for $k \in \{1, \ldots, n\}$

$$u_k = u_{00k} + \sum_{j=1}^{m} \sum_{i=0}^{M} u_{ijk} x^{i(m+1)n+jn} y^i z^j \qquad r = r_{00} + \sum_{j=1}^{m} \sum_{i=0}^{M} r_{ij} x^{i(m+1)n+jn} y^i z^j$$

$$v_k = v_{00k} + \sum_{j=1}^{m} \sum_{i=0}^{M} v_{ijk} y^{-i} z^{-j} \qquad s = s_{00} + \sum_{j=1}^{m} \sum_{i=0}^{M} s_{ij} y^{-i} z^{-j}$$

$$w_k = w_{00k} + \sum_{j=1}^{m} \sum_{i=0}^{M} w_{ijk} x^{i(m+1)n+jn} \qquad t = t_{00} + \sum_{j=1}^{m} \sum_{i=0}^{M} t_{ij} x^{i(m+1)n+jn}$$

$$R = \sum_{k=1}^{n} \tau_k x^k + \sum_{\substack{\alpha=-M\\ \alpha\neq 0}}^{M} \rho_\alpha y^\alpha + \sum_{\substack{\beta=-m\\ \beta\neq 0}}^{m} \sigma_\beta z^\beta$$

Send: $\{u_k, v_k, w_k\}_{k=1}^n, r, s, t, R$.

Verification: Accept the argument if the following holds

$$c_{u_{00}} \prod_{j=1}^{m} c_{u_j}^{x^{jn} z^j} = \mathrm{com}_{ck}(u_1, \ldots, u_n; r) \qquad \prod_{i=0}^{M} C_{U_i}^{x^{i(m+1)n} y^i} = \mathrm{com}_{ck}^{(2)}(c_{u_1}, \ldots, c_{u_m})$$

$$c_{v_{00}} \prod_{j=1}^{m} c_{v_j}^{z^{-j}} = \mathrm{com}_{ck}(v_1, \ldots, v_n; s) \qquad \prod_{i=0}^{M} C_{V_i}^{y^{-i}} = \mathrm{com}_{ck}^{(2)}(c_{v_1}, \ldots, c_{v_m})$$

$$c_{w_{00}} \prod_{j=1}^{m} c_{w_j}^{x^{jn}} = \mathrm{com}_{ck}(w_1, \ldots, w_n; t) \qquad \prod_{i=0}^{M} C_{W_i}^{x^{i(m+1)n}} = \mathrm{com}_{ck}^{(2)}(c_{w_1}, \ldots, c_{w_m})$$

$$\prod_{k=1}^{n} c_{d_k}^{x^k} \cdot \prod_{\substack{\alpha=-M\\ \alpha\neq 0}}^{M} c_{a_\alpha}^{y^\alpha} \cdot \prod_{\substack{\beta=-m\\ \beta\neq 0}}^{m} c_{b_\beta}^{z^\beta} = \mathrm{com}_{ck}(\sum_{k=1}^{n} (u_k v_k - w_k) x^k; R)$$

Theorem 1. *The argument given above has perfect completeness, perfect SHVZK and witness-extended emulation if the two-tiered commitment scheme is binding.*

Proof. We will first show the argument has perfect completeness. Looking at the committed values in the last equation we need

$$\sum_{k=1}^{n} d_k x^k + \sum_{\substack{\alpha=-M\\ \alpha\neq 0}}^{M} a_\alpha y^\alpha + \sum_{\substack{\beta=-m\\ \beta\neq 0}}^{m} b_\beta z^\beta = \sum_{k=1}^{n}(u_k v_k - w_k)x^k.$$

To see this holds, observe that since $u_{i0k} = v_{i0k} = w_{i0k} = 0$ for $i \neq 0$ we have

$$u_k = \sum_{j=0}^{m}\sum_{i=0}^{M} u_{ijk} x^{i(m+1)n+jn} y^i z^j \qquad v_k = \sum_{j=0}^{m}\sum_{i=0}^{M} v_{ijk} y^{-i} z^{-j} \qquad w_k = \sum_{j=0}^{m}\sum_{i=0}^{M} w_{ijk} x^{i(m+1)n+jn}.$$

$$\begin{aligned}
&\sum_{k=1}^{n}(u_k v_k - w_k)x^k\\
=&\sum_{k=1}^{n}\left((\sum_{j=0}^{m}\sum_{i=0}^{M} u_{ijk} x^{i(m+1)n+jn} y^i z^j)(\sum_{j'=0}^{m}\sum_{i'=0}^{M} v_{i'j'k} y^{-i'} z^{-j'}) - w_k \right) x^k\\
=&\sum_{k=1}^{n}\left(\sum_{j=0}^{m}\sum_{j'=0}^{m}\sum_{i=0}^{M}\sum_{i'=0}^{M} u_{ijk} x^{i(m+1)n+jn+k} v_{i'j'k} y^{i-i'} z^{j-j'} - w_k x^k\right)\\
=&\sum_{k=1}^{n}\left(\sum_{j=0}^{m}\sum_{i=0}^{M} u_{ijk} x^{i(m+1)n+jn+k} v_{ijk} - \sum_{j=0}^{m}\sum_{i=0}^{M} w_{ijk} x^{i(m+1)n+jn+k}\right)\\
&+\sum_{k=1}^{n}\sum_{j=0}^{m}\sum_{i=0}^{M}\sum_{\substack{i'=0\\ i'\neq i}}^{M} u_{ijk} x^{i(m+1)n+jn+k} v_{i'jk} y^{i-i'}\\
&+\sum_{k=1}^{n}\sum_{j=0}^{m}\sum_{\substack{j'=0\\ j'\neq j}}^{m}\sum_{i=0}^{M}\sum_{i'=0}^{M} u_{ijk} x^{i(m+1)n+jn+k} v_{i'j'k} y^{i-i'} z^{j-j'}\\
=&\sum_{k=1}^{n}(u_{00k}v_{00k} - w_{00k})x^k \qquad \text{(because } u_{ijk}v_{ijk} - w_{ijk} = 0 \text{ unless } i = j = 0)\\
&+\sum_{k=1}^{n}\sum_{j=0}^{m}\sum_{\substack{\alpha=-M\\ \alpha\neq 0}}^{M}\sum_{\substack{i=0,i'=0\\ i-i'=\alpha}}^{M,M} u_{ijk} x^{i(m+1)n+jn+k} v_{i'jk} y^{\alpha}\\
&+\sum_{k=1}^{n}\sum_{\substack{\beta=-m\\ \beta\neq 0}}^{m}\sum_{\substack{j=0,j'=0\\ j-j'=\beta}}^{m,m}(\sum_{i=0}^{M} u_{ijk} x^{i(m+1)n+jn+k} y^i)(\sum_{i'=0}^{M} v_{i'j'k} y^{-i'}) z^{\beta}\\
=&\sum_{k=1}^{n} d_k x^k + \sum_{\substack{\alpha=-M\\ \alpha\neq 0}}^{M} a_\alpha y^\alpha + \sum_{\substack{\beta=-m\\ \beta\neq 0}}^{m} b_\beta z^\beta
\end{aligned}$$

The remaining equalities follow by straightforward verification giving us perfect completeness.

We will now describe SHVZK simulator $\mathcal{S}$. It gets as input the common reference string ck, the statement $C_{U_1}, \ldots, C_{W_M}$ and the challenges x, y, z. It picks at random $u_k, v_k, w_k \leftarrow \mathbb{Z}_p$ as well as $r, s, t, R \leftarrow \mathbb{Z}_p$. It picks $c_{u_j}, c_{v_j}, c_{w_j} \leftarrow \mathrm{com}_{ck}(0, \ldots, 0)$ and $c_{d_k}, c_{a_\alpha}, c_{b_\beta} \leftarrow \mathrm{com}_{ck}(0)$. It now computes

$$c_{u_{00}} = \prod_{j=1}^{m} c_{u_j}^{-x^{jn} z^j} \mathrm{com}_{ck}(u_1, \ldots, u_n; r) \qquad C_{U_0} = \prod_{i=1}^{M} C_{U_i}^{-x^{i(m+1)n} y^i} \mathrm{com}_{ck}^{(2)}(c_{u_1}, \ldots, c_{u_m})$$

$$c_{v_{00}} = \prod_{j=1}^{m} c_{v_j}^{-z^{-j}} \mathrm{com}_{ck}(v_1, \ldots, v_n; s) \qquad C_{V_0} = \prod_{i=1}^{M} C_{V_i}^{-y^{-i}} \mathrm{com}_{ck}^{(2)}(c_{v_1}, \ldots, c_{v_m})$$

$$c_{w_{00}} = \prod_{j=1}^{m} c_{w_j}^{-x^{jn}} \mathrm{com}_{ck}(w_1, \ldots, w_n; t) \qquad C_{W_0} = \prod_{i=1}^{M} C_{W_i}^{-x^{i(m+1)n}} \mathrm{com}_{ck}^{(2)}(c_{w_1}, \ldots, c_{w_m})$$

$$c_{d_1} = \left(\prod_{k=2}^{n} c_{d_k}^{-x^k} \cdot \prod_{\substack{\alpha=-M \\ \alpha \neq 0}}^{M} c_{a_\alpha}^{-y^\alpha} \cdot \prod_{\substack{\beta=-m \\ \beta \neq 0}}^{m} c_{b_\beta}^{-z^\beta} \mathrm{com}_{ck}\Big(\sum_{k=1}^{n} (u_k v_k - w_k) x^k; R\Big) \right)^{x^{-1}}$$

The simulated argument is

$$\{c_{u_{0j}}, c_{v_{0j}}, c_{w_{0j}}\}_{j=0}^{m}, C_{U_0}, C_{V_0}, C_{W_0}, \{c_{d_k}\}_{k=1}^{n},\ x\ , \{c_{a_\alpha}\}_{\alpha \in \{-M, \ldots, -1, 1, \ldots, M\}},$$
$$y\ , \{c_{b_\beta}\}_{\beta \in \{-m, \ldots, -1, 1, \ldots, m\}}, \{c_{u_j}, c_{v_j}, c_{w_j}\}_{j=1}^{m},\ z\ , \{u_k, v_k, w_k\}_{k=1}^{n}, r, s, t, R.$$

By construction, it is a valid argument just as a real argument is valid. Both simulated arguments and real arguments give uniformly random $u_k, v_k, w_k, r, s, t, R \in \mathbb{Z}_p$. Furthermore, since the commitments are perfectly hiding simulated arguments and real arguments have identical distributions of commitments $c_{u_j}, c_{v_j}, c_{w_j}, c_{d_k}, c_{a_\alpha}, c_{b_\beta}$ for $k \neq 1$. Finally, both in simulated arguments and in real arguments the remaining commitments $c_{u_{00}}, c_{v_{00}}, c_{w_{00}}, C_{U_0}, C_{V_0}, C_{W_0}, c_{d_1}$ are uniquely determined. This means simulated arguments and real arguments have identical probability distributions and therefore the argument has perfect SHVZK.

Let us now describe the witness-extended emulator. It starts by running $\langle \mathcal{P}^*, \mathcal{V} \rangle$ to get a transcript that it will output as the emulated argument. If the verifier rejects on the transcript the emulator has a simulated transcript and it is done. However, if the verifier accepts the emulator will try to extract a witness. The emulator uses repeated rewinding until it has accepting arguments on challenges (x_i, y_{ij}, z_{ijk}) for $i \in \{1, \ldots, (M+1)(m+1)n\}, j \in \{1, \ldots, 2M+1\}, k \in \{1, \ldots, 2m+1\}$. If $\langle \mathcal{P}^*, \mathcal{V} \rangle$ has $\epsilon > 0$ chance of returning an accepting transcript, then it will on average rewind $\frac{(2M+1)(M+1)(2m+1)(m+1)n}{\epsilon}$ times. Since the emulator only rewinds when the transcript is accepting it uses an average of $(2M+1)(M+1)(2m+1)(m+1)n$ rewinds so it runs in expected polynomial time.

Since the emulator runs in expected polynomial time there is negligible probability of breaking the binding property of the commitment scheme, so we will in the analysis assume this does not happen. Also, it is possible that the emulator has to rewind and that two accepting transcripts will use the same challenge, for instance $y_{ij} = y_{ij'}$. However, since the emulator runs in expected polynomial time there is negligible probability that we encounter this event. We can therefore in our analysis focus on the case where all the challenges x_i, y_{ij}, z_{ijk} in the accepting transcripts differ from each other.

Consider now accepting transcripts with challenges $x_1, \ldots, x_{M(m+1)n+1}$. The vectors $(1, \ldots, x_\ell^{M(m+1)n})$ are rows in a Vandermonde matrix. Since a Vandermonde matrix is invertible when the x-values are different, we can for any i' find a linear combination $(\phi_1, \ldots, \phi_{M(m+1)n+1})$ of the vectors that yields $(0, \ldots, 1, 0, \ldots, 0)$ where the single 1 is in position i'. For each x_ℓ we have the verification equation $\prod_{i=0}^{M} C_{W_i}^{x_j^{i(m+1)n}} = \mathrm{com}_{ck}^{(2)}(c_{w_1}^{(j)}, \ldots, c_{w_1}^{(j)})$. By taking linear combinations of the verification equations, we get that $C_{W_{i'}}$ is

$$\prod_{j=1}^{M(m+1)n+1} \left(\prod_{i=0}^{M} C_{W_i}^{x_j^{i(m+1)n}} \right)^{\phi_j} = \mathrm{com}_{ck}^{(2)} \left(\prod_{j=1}^{M(m+1)n+1} (c_{w_1}^{(j)})^{\phi_j}, \ldots, \prod_{j=1}^{M(m+1)n+1} (c_{w_m}^{(j)})^{\phi_j} \right).$$

This gives us an opening $c_{w_{i'1}}, \ldots, c_{w_{i'm}}$ of $C_{W_{i'}}$. Using similar types of calculations we obtain openings of all the commitments, i.e., for each transcript we have openings $u_{ijk}, r_{ij}, v_{ijk}, s_{ij}, w_{ijk}, t_{ij}, d_k, \tau_k, a_\alpha, \rho_\alpha, b_\beta, \sigma_\beta$. The remaining question is whether the extracted $u_{ijk}, v_{ijk}, w_{ijk}$ satisfy $u_{ijk}v_{ijk} = w_{ijk}$ for $i \in \{1, \ldots, M\}, j \in \{1, \ldots, m\}, k \in \{1, \ldots, n\}$.

For each transcript, the binding property of the commitment scheme implies

$$c_{u_j} = \prod_{i=0}^{M} c_{u_{ij}}^{x^{i(m+1)n} y^i} \qquad c_{v_j} = \prod_{i=0}^{M} c_{v_{ij}}^{y^{-i}} \qquad c_{w_j} = \prod_{i=0}^{M} c_{w_{ij}}^{x^{i(m+1)n}}.$$

Define $u_{i0k} = v_{i0k} = w_{i0k} = 0$ for $i \neq 0$ to get for each transcript

$$u_k = \sum_{j=0}^{m} \sum_{i=0}^{M} u_{ijk} x^{i(m+1)n+jn} y^i z^j \qquad v_k = \sum_{j=0}^{m} \sum_{i=0}^{M} v_{ijk} y^{-i} z^{-j} \qquad w_k = \sum_{j=0}^{m} \sum_{i=0}^{M} w_{ijk} x^{i(m+1)n+jn}.$$

The last verification equation now shows that for each transcript

$$\sum_{k=1}^{n} d_k x^k + \sum_{\substack{\alpha=-M \\ \alpha \neq 0}}^{M} a_\alpha y^\alpha + \sum_{\substack{\beta=-m \\ \beta \neq 0}}^{m} b_\beta z^\beta$$
$$= \sum_{k=1}^{n} \left((\sum_{j=0}^{m} \sum_{i=0}^{M} u_{ijk} x^{i(m+1)n+jn+k} y^i z^j)(\sum_{j=0}^{m} \sum_{i=0}^{M} v_{ijk} y^{-i} z^{-j}) - \sum_{j=0}^{m} \sum_{i=0}^{M} w_{ijk} x^{i(m+1)n+jn+k} \right).$$

Multiply both sides by $y^M z^m$ to get a polynomial identity. Since for each transcript with challenges x, y we have $2m+1$ different z-values where the equality holds we get by polynomial interpolation

$$\sum_{k=1}^{n} d_k x^k + \sum_{\substack{\alpha=-M \\ \alpha \neq 0}}^{M} a_\alpha y^\alpha$$
$$= \sum_{k=1}^{n} \sum_{j=0}^{m} \left(\sum_{i=0}^{M} u_{ijk} x^{i(m+1)n+jn+k} y^i)(\sum_{i=0}^{M} v_{ijk} y^{-i}) - \sum_{i=0}^{M} w_{ijk} x^{i(m+1)n+jn+k} \right).$$

For each transcript with challenge x there are $2M+1$ choices of y-values where the equality holds so we get

$$\sum_{k=1}^{n} d_k x^k = \sum_{k=1}^{n} \sum_{j=0}^{m} \sum_{i=0}^{M} (u_{ijk} x^{i(m+1)n+jn+k} v_{ijk} - w_{ijk} x^{i(m+1)n+jn+k}).$$

This equality holds for $(M+1)(m+1)n$ different x-values and therefore the two polynomials are identical so we have for each $i, j, k \neq 0$ that $u_{ijk}v_{ijk} - w_{ijk} = 0$. □

Complexity. The communication complexity of the batch product argument is 3 elements in $\mathbb{T}$, $2M + 5m + n + 1$ elements in $\mathbb{G}$ and $3n + 7$ elements in $\mathbb{Z}_p$.

Let us estimate the computation complexity assuming that we use the two-tiered commitment scheme we described in Section 2.1 in an asymmetric bilinear group with base groups $\mathbb{G}, \hat{\mathbb{G}}$ and target group $\mathbb{T}$. The verifier's computation is $3m$ pairings and exponentiations in the target group $\mathbb{T}$ and $5M + 2m + 4n$ exponentiations in the base group $\mathbb{G}$. Using standard techniques for batch verification some of the equations can be combined in a randomized manner and we may also use multi-exponentiation techniques to reduce the complexity further to $O(\frac{M+m+n}{\log(M+m+n)})$ exponentiations.

A naïve implementation of the prover would require $3m$ pairings and $O(M + m + n)$ exponentiations and $O(N(M + m))$ multiplications in $\mathbb{Z}_p$, where $N - Mmn$. When M or m are large the latter complexity dominates. However, we observe that the commitments $C_{U_0}, C_{V_0}, C_{W_0}$ are commitments to $c_{u_{0j}}, c_{v_{0j}}, c_{w_{0j}}$, which themselves are commitments to 0. Therefore, each of the latter is a power of h, and therefore we can use the bilinear properties of the pairing to rewrite the computation to allow us to compute $C_{U_0}, C_{V_0}, C_{W_0}$ using 3 pairings and $3m$ exponentiations.

We can use techniques for polynomial multiplication to reduce the prover's computation. Consider as an example the computation in round 3, where the prover computes

$$a_\alpha = \sum_{\substack{i=0,i'=0\\ i-i'=\alpha}}^{M,M} \sum_{j=0}^{m} \sum_{k=1}^{n} (u_{ijk} x^{i(m+1)n+jn+k}) v_{i'jk}$$

for $\alpha = -M, \ldots, -1, 1, \ldots, M$. Define $\boldsymbol{u}_i = (u_{i01} x^{i(m+1)n+0n+1}, \ldots, u_{imn} x^{i(m+1)n+mn+n})$ and $\boldsymbol{v}_{i'} = (v_{i'01}, \ldots, v_{i'mn})$, which allows us to rewrite it as

$$a_\alpha = \sum_{\substack{i=0,i'=0\\ i-i'=\alpha}}^{M,M} \boldsymbol{u}_i \boldsymbol{v}_{i'}^\top.$$

Observe that a_α is the $M + \alpha$'th coefficient of the polynomial

$$p(\omega) = \left(\sum_{i=0}^{M} \omega^i \boldsymbol{u}_i \right) \left(\sum_{i'=0}^{M} \omega^{M-i'} \boldsymbol{v}_{i'}^\top \right) \in \mathbb{Z}_p[\omega].$$

The degree of the polynomial is $2M$ so if we evaluate it in $2M+1$ different points $\omega_1, \ldots, \omega_{2M+1} \in \mathbb{Z}_p$ we can use polynomial interpolation to recover the coefficients. The evaluation of $\sum_{i=0}^{M} \omega^i \boldsymbol{u}_i$ and $\sum_{i'=0}^{M} \omega^{M-i'} \boldsymbol{v}_{i'}^\top$ in $2M + 1$ different points can be done using $O(N \log^2 M)$ multiplications. If $2M | p - 1$ and M is a power of 2 we can pick $\omega_1, \ldots, \omega_{2M}$ as $2M$-roots of unity, i.e., $\omega_k^{2M} = 1$ and use the Fast Fourier Transform to reduce the cost further down to $O(N \log M)$ multiplications.[3] Similarly, we can compute $b_{-m}, \ldots, b_{-1}, b_1, \ldots, b_m$ using $O(N \log^2 m)$ multiplications or $O(N \log m)$ multiplications if $2m | p - 1$ and m is a power of 2.

Known values. Sometimes it will be useful to use publicly known values u_{ijk} in the argument. The trivial way to handle this is to use commitments $c_{u_{ij}} = \mathrm{com}_{ck}(u_{ij1}, \ldots, u_{ijn}; 0)$. Since they use trivial randomness, the verifier can check directly that $C_{U_1}, \ldots, C_{U_M}$ contain the correct values. A more careful inspection reveals that some efficiency savings can be made by abandoning the

[3] It takes a while before the assymptotic behaviour kicks in, so for small M it may be better to use Toom-Cook related methods for computing the coefficients $a_{-M}, \ldots, a_M$.

commitments $c_{u_{ij}}$ altogether. Since the u_{ijk} values are public we do not need to hide them, so the prover may choose $u_{0jk} = 0$. The verifier can now herself compute the resulting u_k values without using the commitments at all.

A similar analysis reveals that when w_{ijk} are known the prover does not need to communicate any C_{W_i} or c_{w_j} commitments since the verifier can compute w_k himself. In the special case where $w_{ijk} = 0$ this simplifies to fixing $w_k = 0$.

3.1 Inner product argument

A slight modification of the batch product argument allows the prover to demonstrate instead $\sum_{i=1}^{M}\sum_{j=1}^{m}\sum_{k=1}^{n} u_{ijk}v_{ijk} = \sum_{i=1}^{M}\sum_{j=1}^{m}\sum_{k=1}^{n} w_{ijk}$. The main observation is that we can fix $x = 1$ instead of letting the verifier choose it, in which case equation (1) gives us the desired equality.

The only issue in following this idea is the cross-terms arising from $u_{0jk}, v_{0jk}, w_{0jk}$. We therefore compute $C_{U_0}^x, C_{V_0}^x, C_{W_0}^x, c_{u_{00}}^x, c_{v_{00}}^x, c_{w_{00}}^x$ giving us commitments to $u_{0jk}x, v_{0jk}x, w_{0jk}x$. Since $x \in \mathbb{Z}_p^*$ these values will still ensure that $c_{u_j}, c_{v_j}, c_{w_j}, u_k, v_k, w_k$ do not leak any information about $u_{ijk}, v_{ijk}, w_{ijk}$. But since they are modified by a random factor x throughout the argument they will not interfere with the equation $\sum_{i=1}^{M}\sum_{j=1}^{m}\sum_{k=1}^{n} u_{ijk}v_{ijk} = \sum_{i=1}^{M}\sum_{j=1}^{m}\sum_{k=1}^{n} w_{ijk}$. To get perfect completeness, we use two commitments to d_1 and d_2 values to cancel out cross-terms corresponding to x and x^2.

4 Arguments for Circuit Satisfiability

Using the batch product argument from Section 3 we can give a 7-move SHVZK argument for circuit satisfiability. Consider a boolean circuit consisting of $N - 1$ NAND-gates where the prover wants to convince the verifier that there is a satisfying assignment making the circuit output 1. If the output wire is w, we can add a new variable u and add a self-looping gate of the form $w = \neg(w \wedge u)$, which can only be satisfied if $w = 1$. The prover now has a circuit with N NAND-gates and no output and wants to demonstrate that there is an internally consistent assignment to the wires that respects all gates.

Let us without loss of generality consider a circuit with $N = Mmn$ NAND-gates for which the prover wants to demonstrate that there is a consistent assignment. The prover enumerates the two inputs and the output of each gate as $u_{ijk}, v_{ijk}, w_{ijk}$. The task is now to show that the committed values correspond to a satisfying assignment for the circuit.

The prover first shows that all the committed values are either 0 or 1 corresponding to truth values. This is done by using batch product arguments to show $u_{ijk}u_{ijk} = u_{ijk}, v_{ijk}v_{ijk} = v_{ijk}$ and $w_{ijk}w_{ijk} = w_{ijk}$, which can only be true if $u_{ijk}, v_{ijk}, w_{ijk} \in \{0, 1\}$.

The prover then uses the homomorphic property of the commitment scheme to compute commitments to $1 - w_{ijk}$. Using another batch product argument it can show $u_{ijk}v_{ijk} = 1 - w_{ijk}$, which means the committed values respect the NAND-gates.

Finally, using a technique from [25] it uses an inner product argument to show that all committed values u_{ijk}, v_{ijk} and w_{ijk} corresponding to the same wire x_ℓ are consistent with each other. We describe this technique in the full circuit satisfiability argument below.

Common reference string: Two-tiered commitment key ck.
Statement: $N = Mmn$ NAND-gates $x_{\ell_2} = \neg(x_{\ell_0} \wedge x_{\ell_1})$ over variables x_ℓ.
Prover's witness: An assigment to $\{x_\ell\}$ respecting all NAND-gates.

Argument: Label the inputs and outputs of the gates $\{u_{ijk}, v_{ijk}, w_{ijk}\}_{i=1,j=1,k=1}^{M,m,n}$. Pick $r_{ij}, s_{ij}, t_{ij} \leftarrow \mathbb{Z}_p$ and compute the commitments

$$\begin{aligned} c_{u_{ij}} &= \mathrm{com}_{ck}(u_{ij1}, \dots, u_{ijn}; r_{ij}) & C_{U_i} &= \mathrm{com}_{ck}^{(2)}(c_{u_{i1}}, \dots, c_{u_{im}}) \\ c_{v_{ij}} &= \mathrm{com}_{ck}(v_{ij1}, \dots, v_{ijn}; s_{ij}) & C_{V_i} &= \mathrm{com}_{ck}^{(2)}(c_{v_{i1}}, \dots, c_{v_{im}}) \\ c_{w_{ij}} &= \mathrm{com}_{ck}(w_{ij1}, \dots, w_{ijn}; t_{ij}) & C_{W_i} &= \mathrm{com}_{ck}^{(2)}(c_{w_{i1}}, \dots, c_{w_{im}}) \end{aligned}$$

Send $\{C_{U_i}, C_{V_i}, C_{W_i}\}_{i=1}^{M}$ to the verifier.
Engage in three batch product arguments with statements $\{C_{U_i}, C_{U_i}, C_{U_i}\}_{i=1}^{M}$, $\{C_{V_i}, C_{V_i}, C_{V_i}\}_{i=1}^{M}$ and $\{C_{W_i}, C_{W_i}, C_{W_i}\}_{i=1}^{M}$ in order to show that $u_{ijk}, v_{ijk}, w_{ijk} \in \{0, 1\}$.
Define $c_1 = \mathrm{com}_{ck}(1, \dots, 1; 0)$ and $C_1 = \mathrm{com}_{ck}^{(2)}(c_1, \dots, c_1)$. Engage in a batch product proof with statement $\{C_{U_1}, C_{V_1}, C_1 C_{W_i}^{-1}\}_{i=1}^{M}$ to show that the NAND-gates are respected.
There are $3N = 3Mmn$ committed values $u_{ijk}, v_{ijk}, w_{ijk}$. Let us rename them $\{b_i\}_{i=1}^{3N}$ and the corresponding commitments to $\{C_{B_i}\}_{i=1}^{3M}$. The same variable x_ℓ may appear n_ℓ times in the circuit as $b_{i_1}, \dots, b_{i_{n_\ell}}$. Define π as the permutation in S_{3N} such that for each variable x_ℓ appearing n_ℓ times in the circuit the permutation makes a complete cycle $i_1 \to i_2 \to \dots \to i_{n_\ell} \to i_1$ corresponding to those appearances.
The prover receives a challenge y from the verifier and defines $a_i = y^i - y^{\pi(i)}$. It uses the inner product argument[4] from Section 3.1 to demonstrate $\sum_{i=1}^{3N} a_i b_i = 0$. This shows that for random y

$$\sum_{i=1}^{3N} a_i b_i = \sum_{i=1}^{3N} (y^i - y^{\pi(i)}) b_i = \sum_{i=1}^{3N} y^i (b_i - b_{\pi^{-1}(i)}) = 0.$$

With overwhelming probability over y this shows $b_{\pi(i)} = b_i$ for all i thus proving that the values b_i and hence the values $u_{ijk}, v_{ijk}, w_{ijk}$ are consistent with the wires x_ℓ.
Verification: Verify the 4 batch product proofs and the inner product argument.

Theorem 2. *The argument for circuit satisfiability has perfect completeness, perfect SHVZK and witness-extended emulation.*

Proof. Perfect completeness follows from the perfect completeness of the batch product argument and the inner product argument.

The SHVZK simulator starts by generating trapdoor commitments $c_{u_{ij}}, c_{v_{ij}}, c_{w_{ij}} \leftarrow \mathrm{com}_{ck}(\mathbf{0})$ and setting $C_{U_i} = \mathrm{com}_{ck}^{(2)}(c_{u_{i1}}, \dots, c_{u_{im}}), C_{V_i} = \mathrm{com}_{ck}^{(2)}(c_{v_{i1}}, \dots, c_{v_{im}}), C_{W_i} = \mathrm{com}_{ck}^{(2)}(c_{w_{i1}}, \dots, c_{w_{im}})$. It then runs the SHVZK simulation to simulate the four batch product commitments and the last inner product argument.

To see this is a perfect SHVZK simulation consider a hybrid argument where we have commitments $c_{u_{ij}}, c_{v_{ij}}, c_{w_{ij}}$ to a real witness $u_{ijk}, v_{ijk}, w_{ijk}$ and then proceed to simulate the batch product and the inner product arguments. By the perfect hiding property of the commitment scheme the hybrid argument is perfectly indistinguishable from a simulated argument. At the same time, the perfect SHVZK property implies that for all choices of challenges the hybrid argument and the real argument look identical.

The witness-extended emulator runs the batch product argument emulator and the inner product argument emulator to extract openings of $C_{U_i}, C_{V_i}, C_{W_i}$. The binding property of the commitment scheme implies that the extracted openings are consistent with each other in the four batch product arguments and the inner product argument. The first three batch product

[4] The first round of the inner product argument can be run independently of y such that the total round complexity remains 7.

arguments show that $u_{ijk}, v_{ijk}, w_{ijk} \in \{0,1\}$ and the fourth batch product argument shows $1 - w_{ijk} = u_{ijk}v_{ijk}$, which implies $w_{ijk} = \neg(u_{ijk} \wedge v_{ijk})$. Finally, the inner product argument shows with overwhelming probability over the choice of y that all committed vaues corresponding to the same wire x_ℓ are the same. This shows that the extracted witness satisfies the circuit. □

Arithmetic circuits. Using similar techniques as in the circuit satisfiability argument, we can also get an argument for the satisfiability of arithmetic circuits consisting of addition and multiplication gates over $\mathbb{Z}_p$. The prover commits to the values and uses the homomorphic property of the commitment scheme to show that addition gates are respected and the batch product argument to show that multiplication gates are respected. If there are publicly known constants (without loss of generality a multiple of mn) involved in the circuit, the prover commits to these using randomness 0 so the verifier can check directly that they are correct. As in the circuit satisfiability argument the prover also demonstrates that the committed values are consistent with the wiring of the arithmetic circuit. This gives an arithmetic circuit argument with communication complexity $O(M + m + n)$.

5 Range Arguments

As a concrete application of our batch product argument we will give a communication-efficient range proof. The prover has a commitment c and wants to convince the verifier that she knows an opening w, t such that $c = \mathrm{com}_{ck}(w;t)$ and $w \in [A;B)$. Since the commitment is homomorphic, the problem can be simplified to demonstrating that she knows an opening of $c \cdot \mathrm{com}_{ck}(-A;0)$ in the range $[0; B-A)$. Let $N = \lfloor \log(B-A) \rfloor$. The prover can construct a commitment $c_{0/1} = \mathrm{com}_{ck}(b;s)$ and show that it contains 0 or 1 using standard techniques. By showing that $c \cdot \mathrm{com}_{ck}(-A;0) \cdot c_{0/1}^{A-B+2^N}$ contains a value in the range $[0;2^N)$ she convinces the verifier that $w \in [A;B)$.

We can therefore without loss of generality focus on demonstrating that a committed value w belongs to the interval $[0;2^N)$. We will now give such a range argument that only communicates $O(N^{\frac{1}{3}})$ elements. The idea is that the prover will commit to the bit representation of w. Using a batch product argument the prover can demonstrate that the committed bits are 0 or 1. Furthermore, using techniques similar to the buildup of w_k in the batch product argument the prover will demonstrate that $w = \sum_{i=1}^{M}\sum_{j=1}^{m}\sum_{k=1}^{n} w_{ijk}2^{(i-1)mn+(j-1)n+k-1}$ using $O(M+m+n)$ communication, where $N = Mmn$. If $M = O(N^{\frac{1}{3}}), m = O(N^{\frac{1}{3}}), n = O(N^{\frac{1}{3}})$ the communication complexity is $O(N^{\frac{1}{3}})$ elements.

Common reference string: ck.
Statement: $c \in \mathbb{G}$.
Prover's witness: $w, t \in \mathbb{Z}_p$ such that $w \in [0;2^N)$ and $c = \mathrm{com}_{ck}(w;t)$.
Argument: Let $\{w_{ijk}\}_{i=1,j=1,k=1}^{M,m,n}$ be the bits of w. Pick $r_{ij} \leftarrow \mathbb{Z}_p$ and compute

$$c_{w_{ij}} = \mathrm{com}_{ck}(w_{ij1},\ldots,w_{ijn};r_{ij}) \qquad C_{W_i} = \mathrm{com}_{ck}^{(2)}(c_{w_{i1}},\ldots,c_{w_{im}}) \qquad c_{w_j} = \prod_{i=1}^{M} c_{w_{ij}}^{2^{(i-1)mn}}.$$

Pick $w_{01},\ldots,w_{0n} \leftarrow \mathbb{Z}_p$ and $r_0, s_d \leftarrow \mathbb{Z}_p$ and compute $c_{w_0} = \mathrm{com}_{ck}(w_{01},\ldots,w_{0n};r_0)$ and $c_d = \mathrm{com}_{ck}(\sum_{k=1}^{n} w_{0k}2^{k-1};s_d)$.
Send $\{C_{W_i}\}_{i=1}^{M}$, $\{c_{w_j}\}_{j=0}^{m}$ and c_d to the verifier and get a challenge $x \leftarrow \mathbb{Z}_p^*$ back. Compute

$$w_k = xw_{0k} + \sum_{i=1}^{M}\sum_{j=1}^{m} w_{ijk}2^{(i-1)mn+(j-1)n} \qquad r = xr_0 + \sum_{i=1}^{M}\sum_{j=1}^{m} r_{ij}2^{(i-1)mn+(j-1)n} \qquad s = s_d x + t$$

and send them to the verifier.

In parallel, engage in a batch product argument with statement $\{C_{W_i}, C_{W_i}, C_{W_i}\}_{i=1}^M$ to show that each w_{ijk} satisfies $w_{ijk}w_{ijk} = w_{ijk}$, which implies $w_{ijk} \in \{0,1\}$.

Verification: Verify that the batch product argument is valid and

$$\prod_{i=1}^{M} C_{W_i}^{2^{(i-1)mn}} = \mathrm{com}_{ck}^{(2)}(c_{w_1}, \ldots, c_{w_m}) \qquad c_{w_0}^x \prod_{j=1}^{m} c_{w_j}^{2^{(j-1)n}} = \mathrm{com}_{ck}(w_1, \ldots, w_n; r)$$

$$c_d^x c = \mathrm{com}_{ck}(\sum_{k=1}^{n} w_k 2^{k-1}; s).$$

Theorem 3. *The range argument given above has perfect completeness, perfect SHVZK and witness-extended emulation.*

Proof. Perfect completeness follows from the perfect completeness of the batch product argument and straightforward verification.

The SHVZK simulator on challenge $x \in \mathbb{Z}_p^*$ generates commitments $c_{w_{ij}} \leftarrow \mathrm{com}_{ck}(0, \ldots, 0)$ and picks $w_1, \ldots, w_n, r, s \leftarrow \mathbb{Z}_p$ at random. It then computes C_{w_i}, c_{w_j} as in the real argument and sets

$$c_{w_0} = \left(\prod_{j=1}^{m} c_{w_j}^{-2^{(j-1)n}} \mathrm{com}_{ck}(w_1, \ldots, w_n; r)\right)^{x^{-1}} \qquad c_d = \left(c^{-1}\mathrm{com}_{ck}(\sum_{k=1}^{n} w_k 2^{k-1}; s)\right)^{x^{-1}}.$$

It runs the SHVZK simulator for the batch product argument.

To see this is a perfect simulation note that the perfect hiding property of the commitment scheme means that the commitments $c_{w_{ij}}$ are distributed just as in a real argument. Also, both in simulated arguments and in real arguments $w_1, \ldots, w_n, r, s$ are uniformly random in $\mathbb{Z}_p$. The remaining components are now uniquely determined by the verification equations and therefore simulated arguments and real arguments on challenge x are perfectly indistinguishable.

The witness-extended emulator runs the emulator for the batch product argument to get w_{ijk} and r_{ij} when seeing a succesful transcript. It also rewinds and runs the argument again with fresh challenges x' until getting another acceptable transcript. The two transcripts with extracted w_{ijk} and r_{ij} give us $c_d^x c = \mathrm{com}_{ck}(\sum_{k=1}^n w_k 2^{k-1}; s)$ and $c_d^{x'} c = \mathrm{com}_{ck}(\sum_{k=1}^n w'_k 2^{k-1}; s')$. If $x \neq x'$ we can take appropriate linear combinations to get openings w, t and d, s_d of c and c_d. Similarly, we can by taking appropriate linear combinations of $c_{w_0}^x \prod_{j=1}^m c_{w_j}^{2^{(j-1)n}} = \mathrm{com}_{ck}(w_1, \ldots, w_n; r)$ and $c_{w_0}^{x'} \prod_{j=1}^m c_{w_j}^{2^{(j-1)n}} = \mathrm{com}_{ck}(w'_1, \ldots, w'_n; r')$ get an opening $w_{01}, \ldots, w_{0n}, r_0$ of c_{w_0}. By the binding property of the commitment scheme the last verification equality implies

$$dx + w = \sum_{k=1}^{n} w_k 2^{k-1} = \sum_{k=1}^{n} \left(w_{0k}x + \sum_{i=1}^{M}\sum_{j=1}^{m} w_{ijk} 2^{(i-1)mn+(j-1)n}\right) 2^{k-1}.$$

Since this holds for two different x, we have $w = \sum_{i=1}^M \sum_{j=1}^m \sum_{k=1}^n w_{ijk} 2^{(i-1)mn+(j-1)n+k-1}$. The batch product argument guarantees $w_{ijk} \in \{0,1\}$ giving us $w \in [0; 2^{Mmn})$. □

Acknowledgment

We thank Rafik Chaabouni and Benedikt Bünz for helpful corrections.

References

1. M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, and M. Ohkubo. Structure-preserving signatures and commitments to group elements. In *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 209–236, 2010.
2. S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, May 1998.
3. S. Arora and S. Safra. Probabilistic checking of proofs: a new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998.
4. M. Bellare and O. Goldreich. On defining proofs of knowledge. In *CRYPTO*, volume 740 of *Lecture Notes in Computer Science*, pages 390–420, 1992.
5. E. Ben-Sasson, O. Goldreich, P. Harsha, M. Sudan, and S. P. Vadhan. Short PCPs verifiable in polylogarithmic time. In *IEEE Conference on Computational Complexity*, pages 120–134, 2005.
6. F. Boudot. Efficient proofs that a committed number lies in an interval. In *EUROCRYPT*, volume 1807 of *Lecture Notes in Computer Science*, pages 431–444, 2002.
7. J. Camenisch, R. Chaabouni, and A. Shelat. Efficient protocols for set membership and range proofs. In *ASIACRYPT*, volume 5350 of *Lecture Notes in Computer Science*, pages 234–252, 2008.
8. R. Chaabouni, H. Lipmaa, and A. Shelat. Additive combinatorics and discrete logarithm based range protocols. In *ACISP*, volume 6168 of *Lecture Notes in Computer Science*, pages 336–351, 2010.
9. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO*, volume 893 of *Lecture Notes in Computer Science*, pages 174–187, 1994.
10. I. Damgård. Efficient concurrent zero-knowledge in the auxiliary string model. In *EUROCRYPT*, volume 1807 of *Lecture Notes in Computer Science*, pages 418–430, 2000.
11. I. Damgård and E. Fujisaki. A statistically-hiding integer commitment scheme based on groups with hidden order. In *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 125–142, 2002.
12. I. Damgård and M. J. Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In *PKC*, volume 1992 of *Lecture Notes in Computer Science*, 2001.
13. I. Dinur. The PCP theorem by gap amplification. *Journal of the ACM*, 54(3), 2007.
14. S. D. Galbraith, K. G. Paterson, and N. P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.
15. J. A. Garay, P. D. MacKenzie, and K. Yang. Strengthening zero-knowledge protocols using signatures. *Journal of Cryptology*, 19(2):169–209, 2006.
16. C. Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009.
17. C. Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178, 2009.
18. C. Gentry, J. Groth, Y. Ishai, C. Peikert, A. Sahai, and A. D. Smith. Using fully homomorphic hybrid encryption to minimize non-interactive zero-knowledge proofs. *J. Cryptology*, 28(4):820–843, 2015.
19. O. Goldreich and H. Krawczyk. On the composition of zero-knowledge proof systems. *SIAM Journal on Computing*, 25(1):169–192, 1996.
20. S. Goldwasser, Y. T. Kalai, and G. N. Rothblum. Delegating computation: Interactive proofs for muggles. *Journal of the ACM*, 62(4):27, 2015.
21. S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proofs. *SIAM Journal on Computing*, 18(1):186–208, 1989.
22. J. Groth. Honest verifier zero-knowledge arguments applied. Dissertation Series DS-04-3, BRICS, 2004. PhD thesis. xii+119 pp.
23. J. Groth. Non-interactive zero-knowledge arguments for voting. In *ACNS*, volume 3531 of *Lecture Notes in Computer Science*, 2005.
24. J. Groth. Homomorphic trapdoor commitments to group elements. Cryptology ePrint Archive, Report 2009/007, 2009.
25. J. Groth. Linear algebra with sub-linear zero-knowledge arguments. In *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 192–208, 2009.
26. J. Groth and Y. Ishai. Sub-linear zero-knowledge argument for correctness of a shuffle. In *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 379–396, 2008.
27. Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Zero-knowledge proofs from secure multiparty computation. *SIAM Journal on Computing*, 39(3):1121–1152, 2009.
28. Y. T. Kalai and R. Raz. Interactive pcp. In *ICALP*, volume 5126 of *Lecture Notes in Computer Science*, pages 536–547, 2008.
29. J. Kilian. A note on efficient zero-knowledge proofs and arguments. In *STOC*, pages 723–732, 1992.
30. Y. Lindell. Parallel coin-tossing and constant-round secure two-party computation. *Journal of Cryptology*, 16(3):143–184, 2003.

31. H. Lipmaa. On diophantine complexity and statistical zero-knowledge arguments. In *ASIACRYPT*, volume 2894 of *Lecture Notes in Computer Science*, pages 398–415, 2003.
32. H. Lipmaa, N. Asokan, and V. Niemi. Secure Vickrey auctions without threshold trust. In *Financial Cryptography*, volume 2357 of *Lecture Notes in Computer Science*, pages 87–101, 2002.
33. T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140, 1991.
34. C.-P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.