A non-PCP Approach to Succinct Quantum-Safe Zero-Knowledge

Jonathan Bootle[1,3], Vadim Lyubashevsky[1], Ngoc Khanh Nguyen[1,2], and Gregor Seiler[1,2]

[1] IBM Research – Zurich, Switzerland
nkn@zurich.ibm.com, [0000-0001-8240-6167]
vad@zurich.ibm.com
[2] ETH Zurich, Switzerland
gseiler@inf.ethz.ch
[3] UC Berkeley, USA
jonathan.bootle@berkeley.edu, [0000-0003-3582-3368]

Abstract. Today's most compact zero-knowledge arguments are based on the hardness of the discrete logarithm problem and related classical assumptions. If one is interested in quantum-safe solutions, then all of the known techniques stem from the PCP-based framework of Kilian (STOC 92) which can be instantiated based on the hardness of any collision-resistant hash function. Both approaches produce asymptotically logarithmic sized arguments but, by exploiting extra algebraic structure, the discrete logarithm arguments are a few orders of magnitude more compact in practice than the generic constructions.

In this work, we present the first (poly)-logarithmic, potentially *post-quantum* zero-knowledge arguments that deviate from the PCP approach. At the core of succinct zero-knowledge proofs are succinct commitment schemes (in which the commitment and the opening proof are sub-linear in the message size), and we propose two such constructions based on the hardness of the (Ring)-Short Integer Solution (Ring-SIS) problem, each having certain trade-offs. For commitments to N secret values, the communication complexity of our first scheme is $\tilde{O}(N^{1/c})$ for any positive integer c, and $O(\log^2 N)$ for the second. Both of these are a significant theoretical improvement over the previously best lattice construction by Bootle et al. (CRYPTO 2018) which gave $O(\sqrt{N})$-sized proofs.

Keywords. Lattices, Zero-Knowledge Proofs, SNARKS

1 Introduction

Zero-knowledge proofs are a crucial component in many cryptographic protocols. They are essential to electronic voting, verifiable computation, cryptocurrencies, and for adding stronger security and privacy guarantees to digital signature and encryption schemes. Across almost all applications, it is important to be able to prove in zero-knowledge that one knows how to open a cryptographic commitment, and to prove that the committed values have particular properties or satisfy some relations.

Recent years have seen an explosion of new zero-knowledge proof techniques, each with improvements in proof-size, proving time, or verification time. These new constructions are based on a variety of cryptographic assumptions, including the discrete logarithm assumption [14], various pairing-based assumptions in the Generic and Algebraic Group Models [28,29], collision-resistant hash functions [8,7], and lattice-based assumptions such as (R)SIS and (R)LWE [13,21,3,20].

Of these, only constructions from hash-functions and lattices stand any chance of being post-quantum secure. At this point in time, general-purpose lattice-based proof systems still lag far behind, both asymptotically and in practice, in proof-size and usability. This may seem somewhat surprising, since unlike hash-functions, lattices are endowed with algebraic structure that allows for constructions of rather efficient encryption [37], signature [18,40], and identity-based encryption schemes [25,19]. One could hope that the additional lattice structure can be also exploited for succinct zero-knowledge proofs as well.

1.1 Our Contribution

In this paper, we present two novel lattice-based commitment schemes with associated zero-knowledge opening protocols which prove knowledge of N secret integers with $\tilde{\mathcal{O}}(N^{1/c})$ and $\tilde{\mathcal{O}}(\log^2 N)$ communication complexity, for any constant c. For the former argument, we sketch out a method for constructing an argument of knowledge of a satisfying assignment for an arithmetic circuit with N gates, with $\tilde{\mathcal{O}}(N^{1/c})$ communication complexity (see Appendix B). Both arguments of knowledge follow the same basic methodology, which is to replace the homomorphic commitment schemes used in earlier classically-secure protocols with a commitment scheme based on the (Ring)-SIS problem, and adapt the security proofs to match.

Our constructions follow the usual framework of being interactive schemes converted to non-interactive ones using the Fiat-Shamir transform. As with many schemes constructed in this fashion, their security is proven in the ROM rather than the QROM. The latter would be very strong evidence of quantum security,[4] but the former also appears to give strong evidence of quantum security in practice. To this day, there is no example of a practical scheme that has been proven secure in the ROM based on a quantum-safe computational assumption that has shown any weakness when the adversary was given additional quantum access to the random oracle. A recent line of works (e.g. [31,17,33,16]) that prove security in the QROM of schemes using the Fiat-Shamir transform which previously only known to be secure in the ROM give further evidence that security in the ROM based on a quantum-safe assumption is a meaningful security notion in practice.

Our first construction extends the classical interactive argument of [27] in which the prover commits to message values using Pedersen commitments, and then commits to those commitments using a pairing-based commitment scheme. The two-level structure means that a clever commitment-opening procedure is possible, giving $\tilde{\mathcal{O}}(N^{1/3})$ communication costs. With a d-level commitment scheme, one could hope to extend the technique and construct an argument with $\tilde{\mathcal{O}}(N^{1/(d+1)})$-sized proofs. However, in [27], Pedersen commitments map finite field elements to source group elements, which are mapped to target group elements by the second commitment scheme. In the classical setting, this is as far as the technique can take us, as no homomorphic, compressing commitment scheme for target group elements is known. In the lattice setting, however, the message space for SIS commitments are small integers and commitments are just made up of larger integers. So there is no fundamental reason not to continue. Using careful manipulation of matrices and moduli, our first new argument extends this technique to any constant number of levels.

The second argument is based on the techniques in the Bulletproofs protocol [14], and an earlier protocol [12], which use Pedersen commitments to commit to long message vectors. The additional structure of the Pedersen commitment scheme allows a neat folding technique, which reduces the length of committed message vectors by a factor of two. The prover and verifier repeatedly employ the technique over logarithmically many rounds of interaction until message vectors are reduced to a single value, which the prover can then easily send to the verifier. This gives logarithmic proof sizes. Our new lattice protocol stems from the observation that a SIS-based commitment scheme has some structure similarity to the Pedersen commitment scheme, and thus can be made compatible with the same folding technique. A technical complication that is unique to the lattice setting is keeping the coefficients of the extracted values from growing (too much) during each fold, as a direct adaptation of the bulletproof technique would result in unconstrained growth for every fold which would make the proof meaningless.

Finally, we make a comparison of these two techniques in terms of commitment/proof sizes as well as sizes of the extracted solutions, alternatively called "slack". Our conclusion is that the Bulletproofs folding argument offers smaller poly-logarithmic proof size at the cost much larger slack. Hence, if one does not necessarily need their extracted solution to be very small, then using Bulletproofs appears to be more suitable. However, in many applications, such as group signatures or verifiable encryption, zero-knowledge proofs are just one part of a more complex scheme. If the extracted witnesses are large, then we must adjust parameters not only for the zero-knowledge proof but also for other components of the scheme. Thus, we believe the leveled commitments can be applied in such scenarios at the cost of slightly larger proofs than lattice-based Bulletproofs.

[4] Though still technically heuristic because of the assumption that a concrete hash function acts as a random oracle.

Discussion and Open Problems. The ultimate goal of the line of research that this paper is pursuing is constructing zero-knowledge proofs with concrete parameters smaller than those that can be achieved following the PCP approach. Our current results achieve parameters that are essentially the same asymptotically, but are larger in practice. The asymptotic equivalence comes from the fact that we succeeded in making the *dimension* of the vector(s) representing the proof be logarithmic in the message size. And while we have also somewhat restricted the *coefficient* growth of the proof vector, the coefficients still grow by some factor with each "folding" of the vector dimension. Finding a technique to even further restrict the coefficient growth is the main open problem stemming from this work.

From experience with other primitives, using the additional algebraic structure of concrete assumptions should (eventually) result in size advantages over the generic PCP-based approaches that have the implicit lower bounds (of around 100-200KB) posed by using Merkle-tree commitments. While lattice-based constructions may not achieve the extreme compactness of discrete logarithm based approaches (e.g. Bulletproofs, which have proofs sizes of a few kilobytes for reasonably-sized circuits), there is reason to hope that they can be shorter (and faster) than generic constructions. As an analogy, when lattice-based signatures first appeared [25,36], they were significantly larger than the generic quantum-safe signatures that one could construct using techniques, such as one-way functions and Merkle trees, dating back to the 1970s [32,38]. But expanding upon these early lattice constructions via novel algorithms and techniques exploiting the underlying mathematical structure of lattices, the current state-of-the-art lattice-based signatures [18,40] are currently an order of magnitude smaller and two orders of magnitude faster than those stemming from generic constructions [10]. We believe that the techniques of this paper can similarly be the beginning of the path to more practical succinct quantum-safe zero-knowledge.

1.2 Technical Overview

Levelled Commitments. The commitment scheme in [5] arranged N elements of $\mathbb{Z}_q$ to which one wants to commit to, into an $m \times k$ matrix $\mathbf{S}$ and created the commitment

$$\mathbf{A} \cdot \mathbf{S} = \mathbf{T} \bmod q \tag{1}$$

where $\mathbf{A} \leftarrow \mathbb{Z}_p^{n \times m}$ is a random matrix and $p < q$. Notice that $\mathbf{T} \in \mathbb{Z}_q^{n \times k}$, and [5] showed that the proof of knowledge of an $\mathbf{S}$ with small (but larger than the honest prover uses in the proof) coefficients satisfying (1) can be done with λm elements in $\mathbb{Z}_q$ [5] where λ is a security parameter. The total size of the proof is therefore the size of $\mathbf{T}$ and the size of the proof of (1), which is $nk + \lambda m$ elements in $\mathbb{Z}_q$. Since $n = \mathcal{O}(\lambda)$, the optimal way to commit to N elements in $\mathbb{Z}_q$ is to arrange them into a matrix $\mathbf{S} \in \mathbb{Z}_q^{m \times k}$, where $m = k = \tilde{\mathcal{O}}(\sqrt{N})$. This makes the total proof size $\tilde{\mathcal{O}}(\sqrt{N})$.

To illustrate our levelled commitment technique, we will describe a commitment scheme and a protocol for achieving a proof size of $\tilde{\mathcal{O}}(N^{1/3})$. We will commit to $\mathbf{S} \in \mathbb{Z}^{m_1 \cdot m_2 \times m_3}$ as

$$\mathbf{A}_1 \cdot ((\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \mathbf{S} \bmod q_2) \bmod q_1 = \mathbf{T} \ , \tag{2}$$

where $\mathbf{A}_1 \leftarrow \mathbb{Z}_{q_1}^{n \times n m_1}, \mathbf{A}_2 \leftarrow \mathbb{Z}_{q_2}^{n \times m_2}$. Our proof will prove knowledge of an $\bar{\mathbf{S}}$ with somewhat larger coefficients than $\mathbf{S}$, and also an $\bar{\mathbf{R}} \in \mathbb{Z}^{n \cdot m_1 \times m_3}$ satisfying

$$\mathbf{A}_1 \cdot \left((\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \bar{\mathbf{S}} \bmod q_2 + \bar{\mathbf{R}} \cdot q_2\right) \bmod q_1 = \mathbf{T}, \tag{3}$$

Let us first show that the above extracted commitment of $(\bar{\mathbf{S}}, \bar{\mathbf{R}})$ is binding based on the hardness of SIS when $\|\bar{\mathbf{S}}\| \ll q_2, \|\bar{\mathbf{R}}\| \ll q_1/q_2$ and $q_2 \ll q_1$. Suppose, for contradiction, there are two $(\bar{\mathbf{S}}, \bar{\mathbf{R}}) \neq (\bar{\mathbf{S}}', \bar{\mathbf{R}}')$ satisfying (3). In the first case, suppose that $\bar{\mathbf{R}} \neq \bar{\mathbf{R}}'$. By definition, the coefficients of $(\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \bar{\mathbf{S}} \bmod q_2$ are smaller than q_2, and thus $\bar{\mathbf{R}} \neq \bar{\mathbf{R}}'$ implies that

$$(\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \bar{\mathbf{S}} \bmod q_2 + \bar{\mathbf{R}} \cdot q_2 \neq (\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \bar{\mathbf{S}}' \bmod q_2 + \bar{\mathbf{R}}' \cdot q_2. \tag{4}$$

[5] We provide additional background in Section 2.3 for readers not familiar with previous work.

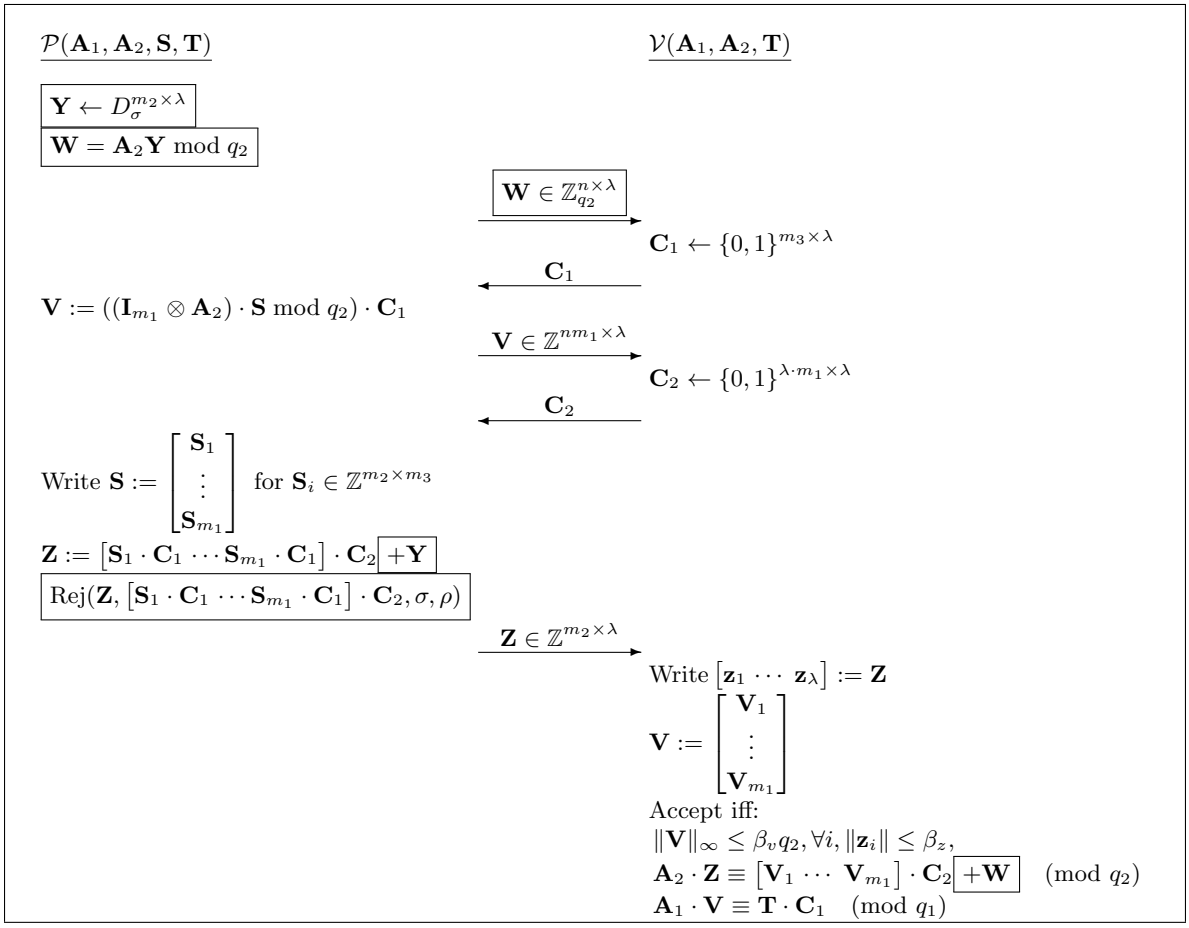

Fig. 1. Levelled commitment with two levels. Here, β_v and β_z are parameters which satisfy $\beta_z \ll q_2, \beta_v \ll q_1/q_2$.

If the parameters are set such that the coefficients of both sides of the above equation are less than q_1, then this gives a solution to SIS for $\mathbf{A}_1$. Now assume that $\bar{\mathbf{R}} = \bar{\mathbf{R}}'$, and so $\bar{\mathbf{S}} \neq \bar{\mathbf{S}}'$. If $(\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \bar{\mathbf{S}} \equiv (\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \bar{\mathbf{S}}' \pmod{q_2}$, then there must be some $\bar{\mathbf{S}}_i \neq \bar{\mathbf{S}}'_i \in \mathbb{Z}^{m_2 \times m_3}$ such that $\mathbf{A}_2 \cdot \bar{\mathbf{S}}_i \equiv \mathbf{A}_2 \cdot \bar{\mathbf{S}}'_i \pmod{q_2}$, and so we have a SIS solution for $\mathbf{A}_2$. If $(\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \bar{\mathbf{S}} \not\equiv (\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \bar{\mathbf{S}}' \pmod{q_2}$, then the inequality in (4) holds and we have a SIS solution for $\mathbf{A}_1$.

We present the basic protocol in Fig. 1. The boxed text contains the parts necessary to make the protocol zero-knowledge. In this overview, we will ignore these and only show that the protocol is a proof of knowledge. First, let us show the correctness of the protocol. Because the coefficients of $\mathbf{S}, \mathbf{C}_1$, and $\mathbf{C}_2$ are small, the coefficients of $\mathbf{Z}$ are also small with respect to q_2. Similarly, because the coefficients of $\mathbf{V}$ consist of a product of a matrix with coefficients less than q_2 with a 0/1 matrix $\mathbf{C}_1$, parameters can be set such that the coefficients of the product are less than q_1. Thus $\|\mathbf{z}_i\| \leq \beta_z$ and $\|\mathbf{V}\| \leq \beta_v q_2$ can be satisfied with an appropriate choice of parameters. We now move on to showing that the verification equations hold. Note that

$$\mathbf{V} = \begin{bmatrix} \mathbf{V}_1 \\ \vdots \\ \mathbf{V}_{m_1} \end{bmatrix} = \left(\begin{bmatrix} \mathbf{A}_2\mathbf{S}_1 \\ \vdots \\ \mathbf{A}_2\mathbf{S}_{m_1} \end{bmatrix} \bmod q_2 \right) \cdot \mathbf{C}_1 \equiv \begin{bmatrix} \mathbf{A}_2\mathbf{S}_1\mathbf{C}_1 \\ \vdots \\ \mathbf{A}_2\mathbf{S}_{m_1}\mathbf{C}_1 \end{bmatrix} \pmod{q_2}, \tag{5}$$

and so one can write

$$\mathbf{A}_2 \cdot \left[\mathbf{S}_1\mathbf{C}_1 \cdots \mathbf{S}_{m_1}\mathbf{C}_1\right] \equiv \left[\mathbf{V}_1 \cdots \mathbf{V}_{m_1}\right] \pmod{q_2}, \tag{6}$$

and therefore $\mathbf{A}_2 \cdot \mathbf{Z} \equiv \mathbf{A}_2 \cdot \left[\mathbf{S}_1\mathbf{C}_1 \cdots \mathbf{S}_{m_1}\mathbf{C}_1\right] \cdot \mathbf{C}_2 \pmod{q_2}$, which is the first verification equation. For the second verification equation, observe from (5) and (3) that

$$\mathbf{A}_1 \cdot \mathbf{V} = \mathbf{A}_1 \cdot \left(\begin{bmatrix} \mathbf{A}_2\mathbf{S}_1 \\ \vdots \\ \mathbf{A}_2\mathbf{S}_{m_1} \end{bmatrix} \bmod q_2 \right) \cdot \mathbf{C}_1 \equiv \mathbf{T} \cdot \mathbf{C}_1 \pmod{q_1}.$$

Finally, ignoring constant terms, the total communication cost (including the statement) can be bounded above by

$$m_3 \cdot (n \log q_1 + \lambda) + m_2 \cdot \lambda \log \beta_z + m_1 \cdot (n\lambda \log(\beta_v q_2) + \lambda^2) + n\lambda \log q_2.$$

Note that the last term does not depend on m_1, m_2, m_3 hence we ignore it for now. Therefore, in order to minimise the expression above, we want to set m_1, m_2, m_3 such that all three corresponding terms are (almost) equal. We select appropriate n, q_1, q_2 such that both $\mathbf{A}_1$ and $\mathbf{A}_2$ are binding (9) and we get $\log q_2 < \log q_1 = \mathcal{O}(\log N)$ and $n = \mathcal{O}(\lambda)$. Similarly, $\log \beta_v = \mathcal{O}(\log N)$ and $\log \beta_z = \mathcal{O}(\log N)$. Therefore, the total communication cost is approximately $\tilde{\mathcal{O}}(\sqrt[3]{N})$.

In Section 3, we extend this approach to more than two levels. Generally, we propose a proof of knowledge for $d \geq 1$ levels with total communication size equal to

$$\mathcal{O}\left(N^{\frac{1}{d+1}} \cdot (d^3\lambda \log^2 N + d\lambda^2)\right).$$

In Section 3.3, we also show how to apply techniques similar to previous work [14,12,5] in order to extract a relatively short solution to the relaxed equation (e.g. (3) for $d = 2$). Due to space limitations, we skip the details in this overview.

Bulletproofs folding. Our starting point is the lattice equation:

$$\mathbf{A}\mathbf{s} = \mathbf{t} \tag{7}$$

where $\mathbf{A} \in R^{1\times k}, \mathbf{s} \in R^k$ and $R = \mathbb{Z}_q[X]/(X^n + 1)$. Thus, the number of secrets is $N = kn$. In the same vein as [14,12], we are interested in constructing a protocol where proving knowledge of pre-image $\mathbf{s}$ of $\mathbf{t}$ comes down to proving knowledge of some other pre-image, say $\mathbf{s}'$, whose length $k/2$ is half that of $\mathbf{s}$. By recursively applying this argument $\log k$ times, we obtain poly-logarithmic proof size. Concretely, we fold the initial statement (7) as follows. Let us write $\mathbf{A}$ and $\mathbf{s}$ as

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{A}_2 \end{bmatrix} \text{ and } \mathbf{s} = \begin{bmatrix} \mathbf{s}_1 \\ \mathbf{s}_2 \end{bmatrix} \text{ where } \mathbf{s}_1, \mathbf{s}_2 \in R^{k/2}.$$

Let $\mathbf{l} := \mathbf{A}_1\mathbf{s}_2 \in R$ and $\mathbf{r} := \mathbf{A}_2\mathbf{s}_1 \in R$. Then, for all $c \in R$,

$$(c\mathbf{A}_1 + \mathbf{A}_2)(\mathbf{s}_1 + c\mathbf{s}_2) = c^2\mathbf{l} + c\mathbf{t} + \mathbf{r} \ .$$

We observe that $\mathbf{s}_1 + c\mathbf{s}_2$ has length $k/2$, suggesting the following protocol for proving knowledge of $\mathbf{s}$. First, the prover $\mathcal{P}$ sends $\mathbf{l}, \mathbf{r}$ to the verifier $\mathcal{V}$. Then, $\mathcal{V}$ samples a challenge c uniformly at random from a challenge space $\mathcal{C} \subseteq R$ and sends it to $\mathcal{P}$. Finally, $\mathcal{P}$ sends $\mathbf{z} = \mathbf{s}_1 + c\mathbf{s}_2$ to $\mathcal{V}$. Note that if $\mathcal{P}$ is honest then $\mathbf{z}$ satisfies the new lattice equation

$$\mathbf{B}\mathbf{z} = \mathbf{t}'$$

where $\mathbf{B} = c\mathbf{A}_1 + \mathbf{A}_2$ and $\mathbf{t}' = c^2\mathbf{l} + c\mathbf{t} + \mathbf{r}$. Therefore, instead of sending $\mathbf{z}$ directly, the prover might repeat the protocol and treat $\mathbf{z}$ as a new secret vector. By folding all the way down to vectors of length 1, we get communication costs of order $\log k$.

Using similar techniques to [14,12], one can extract a solution $\bar{\mathbf{z}}$ to the original equation $\mathbf{A}\bar{\mathbf{z}} = \mathbf{t}$. The problem is that unless we define the challenge space properly, we do not have any information on the size of $\|\bar{\mathbf{z}}\|$. Hence, we let $\mathcal{C}$ be the set of monomials of R, i.e. $\mathcal{C} = \{X^i : i \in \mathbb{Z}\}$. Then, using the fact that polynomials

of the form $2/(X^i - X^j) \in R$ have coefficients in $\{-1, 0, 1\}$ [9], we bound the size of an extracted solution. The only drawback of using this approach is that we only obtain a solution for a relaxed equation. Concretely, if we apply the folding technique d times, then we only manage to find a small solution $\bar{\mathbf{z}}$ for the equation $\mathbf{A}\bar{\mathbf{z}} = 8^d\mathbf{t}$ such that $\|\bar{\mathbf{z}}\| = \mathcal{O}\left(n^{3d} \cdot 12^d \cdot p\right)$ where $p = \|\mathbf{s}\|_\infty$. For $d = \log k$, the relaxation factor becomes k^3. The communication cost for the $(2d+1)$-round version of the protocol is equal to

$$N \log(2^d p)/2^d + 2dn \log q.$$

Then, we would just pick q which is a little bit larger than the slack. It is worth mentioning that the protocol in its current state gives us soundness error of order $1/n$, hence we would need to repeat it $\lambda/\log n$ times in order to achieve soundness error $2^{-\lambda}$. Therefore, the total proof size can be bounded by

$$\mathcal{O}\left(\frac{\lambda N \log(2^d p)}{2^d \log n} + \lambda d^2 n\right).$$

Comparison. We investigate in which applications one technique offers asymptotically smaller proof size than the other (see Section 4 for more details). First of all, consider the case when we do not require the extracted solution to be "very small". Then, levelled commitments for $d = \log N - 1$ levels provide proof size of order

$$\mathcal{O}\left(\lambda \log^5 N + \lambda^2 \log N\right).$$

On the other hand, by applying Bulletproofs folding $d = \log k$ times, we obtain proof size[6]

$$\mathcal{O}\left(\lambda n \log N \cdot (\log N + 1)\right).$$

Consequently, the Bulletproofs approach achieves smaller proof size.

Next, consider the case when one could only afford limited slack, i.e. the extracted solution is smaller than some set value $B = N^\alpha > N^2$. First, suppose that $N = \lambda^r$ for some $r \geq 3$ (we expect N to be much bigger than λ). Then, we show that levelled commitments and Bulletproofs provide $\tilde{\mathcal{O}}(N^u)$ and $\tilde{\mathcal{O}}(N^v)$ proof sizes respectively, where [7]

$$u \approx \frac{1}{(\alpha-2)r} \text{ and } v \approx 1 - \frac{\alpha - 1/2}{3\log n + 4}.$$

Assume the allowed slack is small enough that both u and v are larger than $1/\log N$. Then, we just check which one of u, v is bigger[8]. Since $\log n \geq 1$ and for all $r \geq 3$, the function

$$f_r(x) := (15 - 2x)(x - 2)r - 14$$

is positive for $3 \leq x \leq 7$, we deduce that u is smaller than v when $\alpha \leq 7$. This suggests that one should use the levelled commitments protocol when one can only tolerate a limited amount of slack.

1.3 Related Work

In this paper, we investigate techniques from [14,12] and [27] in the lattice world. These papers are the most closely related prior works, along with [14], which forms a key component of the argument in Section 3.

We review proof systems which can prove knowledge of a secret with N elements, or prove knowledge of a satisfying assignment to an arithmetic circuit with N gates.

[6] We note that more concrete bounds could be computed. However, this non-tight bound already shows that Bulletproofs folding offers smaller proof size.

[7] Here, n denotes the degree of the underlying cyclotomic polynomial $X^n + 1$.

[8] This would only *asymptotically* tell us which method offers smaller proof size.

Lattice-Based Arguments. The zero-knowledge argument given in [5] is based on the SIS assumption, and is capable of proving knowledge of commitment openings with proof size $\mathcal{O}(\sqrt{N})$. It was the first and only standard zero-knowledge protocol based on lattice assumptions to achieve a sublinear communication complexity. Previously, the only other lattice-based arguments of knowledge with better asymptotic proof size were lattice-based SNARKs [11,24,39]. Although they offer highly succinct, $\mathcal{O}(1)$-sized proofs, the proofs are only checkable by a designated verifier, and soundness is based on strong, non-falsifiable assumptions.

Hash-Based Arguments STARKs [6] and Aurora [8] are non-interactive argument systems. Both achieve $\mathcal{O}(\log^2 N)$-sized proofs, with Aurora more efficient by an order of magnitude due to better constants. Ligero [2] achieves $\mathcal{O}(\sqrt{N})$-sized proofs, but is highly efficient in practice.

Classically-Secure Arguments. In the discrete-logarithm setting, Bulletproofs [14] and a related argument [12] give $\mathcal{O}(\log N)$ communication complexity, using Pedersen commitments and the same recursive folding technique that inspired the argument described in Section 4. The protocol of [27] gives $\mathcal{O}(N^{1/3})$ proof sizes, and uses a two-tiered commitment scheme, based on Pedersen commitments and a related commitment scheme based on pairings. We extend the same idea to a multi-levelled lattice-based commitment scheme in Section 3.

There is also a long line of works on succinct non-interactive arguments based on pairings, culminating in protocols including [28] and [29] which have $\mathcal{O}(1)$ proof size, but rely on strong, non-falsifiable assumptions like the Knowledge-of-Exponent assumptions, or have security proofs in idealised models like the Generic Group Model [41] or Algebraic Group Model [22].

2 Preliminaries

Algorithms in our schemes receive a security parameter λ as input (sometimes implicitly) written in unary. Unless stated otherwise, we assume all our algorithms to be probabilistic. We denote by $\mathcal{A}(x)$ the probabilistic computation of the algorithm $\mathcal{A}$ on input x. If $\mathcal{A}$ is deterministic, we write $y := \mathcal{A}(x)$. We write PPT (resp. DPT) for probabilistic (resp. deterministic) polynomial time algorithms. The notation $y \leftarrow \mathcal{A}(x)$ denotes the event that $\mathcal{A}$ on input x returns y. Given two functions $f, g : \mathbb{N} \to [0,1]$ we write $f(\lambda) \approx g(\lambda)$ when $|f(\lambda) - g(\lambda)| = \lambda^{-\omega(1)}$. We say that f is *negligible* when $f(\lambda) \approx 0$ and that f is *overwhelming* when $f(\lambda) \approx 1$. For $n \in \mathbb{N}$, we write $[n] := \{1, \ldots, n\}$. Regular font letters denote elements in $\mathbb{Z}$ or $\mathbb{Z}_q$, for a prime q, and bold lower-case letters represent column vectors with coefficients in $\mathbb{Z}$ or $\mathbb{Z}_q$. Bold upper-case letters denote matrices. By default, all vectors are column vectors. Let $\mathbf{I}_n \in \mathbb{Z}_q^{n \times n}$ be the $n \times n$ identity matrix. We write a list of objects with square brackets, e.g. $[a_1, \ldots, a_k]$ is a list of k objects: $a_1, \ldots, a_k$. Also, we denote by $[]$ the empty list. For any statement st, we define $[\![st]\!]$ to be equal to 1 if st is true and 0 otherwise.

Sizes of elements. For an even (resp. odd) positive integer α, we define $r' = r \bmod \alpha$ to be the unique element r' in the range $-\frac{\alpha}{2} < r' \leq \frac{\alpha}{2}$ (resp. $-\frac{\alpha-1}{2} \leq r' \leq \frac{\alpha-1}{2}$) such that $r' = r \bmod \alpha$. For an element $w \in \mathbb{Z}_q$, we write $\|w\|_\infty$ to mean $|w \bmod q|$. Define the ℓ_∞ and ℓ_2 norms for $\mathbf{w} = (w_1, \ldots, w_k) \in \mathbb{Z}_q^k$ as follows:

$$\|\mathbf{w}\|_\infty = \max_i \|w_i\|_\infty, \quad \|\mathbf{w}\| = \sqrt{\|w_1\|_\infty^2 + \ldots + \|w_k\|_\infty^2}.$$

However, if we do not state explicitly that $\mathbf{w} \in \mathbb{Z}_q^k$ but rather treat $\mathbf{w}$ as a vector of integers then the standard notions of L_2 and L_∞ norms apply. We will also consider the operator norm of matrices over $\mathbb{Z}$ defined by $s_1(\mathbf{A}) = \max_{\|\mathbf{x}\| \neq 0} \left(\frac{\|\mathbf{A}\mathbf{x}\|}{\|\mathbf{x}\|} \right)$.

Probability Distributions. Let $\mathcal{D}$ denote a distribution over some set S. Then, $d \leftarrow \mathcal{D}$ means that d was sampled from the distribution $\mathcal{D}$. If we write $d \xleftarrow{\$} S$ for some finite set S without a specified distribution this means that d was sampled uniformly random from S. We let $\Delta(X, Y)$ indicate the statistical distance between

two distributions X, Y. Define the function $\rho_\sigma(x) = \exp\left(\frac{-x^2}{2\sigma^2}\right)$ and the discrete Gaussian distribution over the integers, D_σ, as

$$D_\sigma(x) = \frac{\rho(x)}{\rho(\mathbb{Z})} \text{ where } \rho(\mathbb{Z}) = \sum_{v \in \mathbb{Z}} \rho(v).$$

We will write $\mathbf{A} \leftarrow D_\sigma^{k \times \ell}$ to mean that every coefficient of the matrix $\mathbf{A}$ is distributed according to D_σ.

Using the tail bounds for the 0-centered discrete Gaussian distribution (cf. [4]), we can show that for any $\sigma > 0$ the norm of $x \leftarrow D_\sigma$ can be upper-bounded using σ. Namely, for any $t > 0$,

$$\Pr_{x \leftarrow D_\sigma}[|x| > t\sigma] \leq 2e^{-t^2/2},$$

and when $\mathbf{x}$ is drawn from D_σ^m, we have

$$\Pr_{\mathbf{x} \leftarrow D_\sigma^m}[\|\mathbf{x}\| > \sqrt{2m} \cdot \sigma] < 2^{-m/4}. \tag{8}$$

2.1 Lattice-based Commitment Schemes

A non-interactive commitment scheme is a pair of PPT algorithms (Gen, Com). The setup algorithm $ck \leftarrow \text{Gen}(1^\lambda)$ generates a commitment key ck, which specifies message, randomness and commitment spaces $\mathsf{M}_{ck}, \mathsf{R}_{ck}, \mathsf{C}_{ck}$. It also specifies an efficiently sampleable probability distribution $D_{\mathsf{R}_{ck}}$ over R_{ck} and a binding set $\mathsf{B}_{ck} \subset \mathsf{M}_{ck} \times \mathsf{R}_{ck}$. The commitment key also specifies a deterministic polynomial-time commmitment function $\text{Com}_{ck} : \mathsf{M}_{ck} \times \mathsf{R}_{ck} \to \mathsf{C}_{ck}$. We define $\text{Com}_{ck}(\mathbf{m})$ to be the probabilistic algorithm that given $\mathbf{m} \in \mathsf{M}_{ck}$ samples $\mathbf{r} \leftarrow D_{\mathsf{R}_{ck}}$ and returns $\mathbf{c} = \text{Com}_{ck}(\mathbf{m}; \mathbf{r})$.

The commitment scheme is homomorphic, if the message, randomness and commitment spaces are abelian groups (written additively) and we have for all $\lambda \in \mathbb{N}$, and for all $ck \leftarrow \text{Gen}(1^\lambda)$, for all $\mathbf{m}_0, \mathbf{m}_1 \in \mathsf{M}_{ck}$ and for all $\mathbf{r}_0, \mathbf{r}_1 \in \mathsf{R}_{ck}$:

$$\text{Com}_{ck}(\mathbf{m}_0; \mathbf{r}_0) + \text{Com}_{ck}(\mathbf{m}_1; \mathbf{r}_1) = \text{Com}_{ck}(\mathbf{m}_0 + \mathbf{m}_1; \mathbf{r}_0 + \mathbf{r}_1).$$

Definition 2.1 (Hiding). *The commitment scheme is hiding if for all PPT stateful interactive adversaries* $\mathcal{A}$

$$\Pr\left[\begin{matrix} ck \leftarrow \text{Gen}(1^\lambda); (\mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathcal{A}(ck); b \leftarrow \{0,1\}; \\ \mathbf{r} \leftarrow D_{\mathsf{R}_{ck}}; \mathbf{c} \leftarrow \text{Com}_{ck}(\mathbf{m}_b; \mathbf{r}) : \mathcal{A}(\mathbf{c}) = b \end{matrix}\right] \approx \frac{1}{2},$$

where $\mathcal{A}$ *outputs* $\mathbf{m}_0, \mathbf{m}_1 \in \mathsf{M}_{ck}$.

Definition 2.2 (Binding). *The commitment scheme is computationally binding if a commitment can only be opened to one value within the binding set* B_{ck}. *For all PPT adversaries* $\mathcal{A}$

$$\Pr\left[\begin{matrix} ck \leftarrow \text{Gen}(1^\lambda); (\mathbf{m}_0, \mathbf{r}_0, \mathbf{m}_1, \mathbf{r}_1) \leftarrow \mathcal{A}(ck) : \\ \mathbf{m}_0 \neq \mathbf{m}_1 \text{ and } \text{Com}_{ck}(\mathbf{m}_0; \mathbf{r}_0) = \text{Com}_{ck}(\mathbf{m}_1; \mathbf{r}_1) \end{matrix}\right] \approx 0,$$

where $\mathcal{A}$ *outputs* $(\mathbf{m}_0, \mathbf{r}_0), (\mathbf{m}_1, \mathbf{r}_1) \in \mathsf{B}_{ck}$.

The commitment scheme is compressing if the sizes of commitments are smaller than the sizes of the committed values.

Compressing Commitments Based on SIS. We work with the standard SIS (shortest integer solution) commitment scheme [5,30], which was already implicit in the aforementioned work of Ajtai [1] and uses uniformly random matrices $\mathbf{A}_1 \in \mathbb{Z}_q^{r \times 2r \log_p q}$ and $\mathbf{A}_2 \in \mathbb{Z}_q^{r \times n}$ as a commitment key, where n is the number of elements that one wishes to commit to and $p < q$. A commitment to a vector $\mathbf{m} \in \mathbb{Z}_p^n$ involves choosing a random vector $\mathbf{r} \in \mathbb{Z}_p^{2r \log_p q}$ and outputting the commitment vector $\mathbf{v} = \mathbf{A}_1\mathbf{r} + \mathbf{A}_2\mathbf{m} \bmod q$. By the leftover

hash lemma, $(\mathbf{A}_1, \mathbf{A}_1\mathbf{r} \bmod q)$ is statistically close to uniform, and so the commitment scheme is statistically hiding.[9] To prove binding, note that if there are two different $(\mathbf{r}, \mathbf{m}) \neq (\mathbf{r}', \mathbf{m}')$ such that

$$\mathbf{v} = \mathbf{A}_1\mathbf{r} + \mathbf{A}_2\mathbf{m} \equiv \mathbf{A}_1\mathbf{r}' + \mathbf{A}_2\mathbf{m}' \pmod{q},$$

then

$$\mathbf{A}_1(\mathbf{r} - \mathbf{r}') + \mathbf{A}_2(\mathbf{m} - \mathbf{m}') \equiv \mathbf{0} \pmod{q}$$

and the non-zero vector $\mathbf{s} = \begin{bmatrix} \mathbf{r} - \mathbf{r}' \\ \mathbf{m} - \mathbf{m}' \end{bmatrix}$ is a solution to the SIS problem for the matrix $\mathbf{A} = [\mathbf{A}_1 \ \mathbf{A}_2]$, i.e. $\mathbf{A}\mathbf{s} \equiv \mathbf{0} \pmod{q}$. As long as the parameters are set such that $\|\mathbf{s}\|$ is smaller than

$$\min\{q, 2^{2\sqrt{r \log q \log \delta}}\}^{10}, \tag{9}$$

the binding property of the commitment is based on an intractable version of the SIS problem [23].

In this paper, we will use the following lattice commitment scheme.

$\mathrm{Gen}(1^\lambda) \to ck$**:** Select parameters p, q, r, v, N, B, σ.

Pick uniformly random matrices $\mathbf{A}_1 \xleftarrow{\$} \mathbb{Z}_q^{r \times r \log_p q}$ and $\mathbf{A}_2 \xleftarrow{\$} \mathbb{Z}_q^{r \times n}$.

Return $ck = (p, q, r, v, \ell, N, \beta, \mathbb{Z}_q, \mathbf{A}_1, \mathbf{A}_2)$.

The commitment key defines the following message, randomness, commitment and binding spaces and randomness distribution

$$\mathsf{M}_{ck} = \mathbb{Z}_q^n, \quad \mathsf{R}_{ck} = \mathbb{Z}_q^{2r \log_p q}, \quad \mathsf{C}_{ck} = \mathbb{Z}_q^r$$

$$\mathsf{B}_{ck} = \left\{ \mathbf{s} = \begin{bmatrix} \mathbf{m} \\ \mathbf{r} \end{bmatrix} \in \mathbb{Z}_q^{n+2r \log_p q} \;\middle|\; ||\mathbf{s}|| < B \right\}, \quad D_{\mathsf{R}_{ck}} = D_\sigma^r.$$

$\mathrm{Com}_{ck}(\mathbf{m}; \mathbf{r})$**:** Given $\mathbf{m} \in \mathbb{Z}_q^n$ and $\mathbf{r} \in \mathbb{Z}_q^{2r \log_p q}$ return $\mathbf{c} = \mathbf{A}_1\mathbf{r} + \mathbf{A}_2\mathbf{m}$.

In the following, when we make multiple commitments to vectors $\mathbf{m}_1, \ldots, \mathbf{m}_\ell \in \mathsf{M}_{ck}$ we write $\mathbf{C} = \mathrm{Com}_{ck}(\mathbf{M}; \mathbf{R})$ when concatenating the commitment vectors as $\mathbf{C} = [\mathbf{c}_1, \cdots, \mathbf{c}_\ell]$. This corresponds to computing $\mathbf{C} = \mathbf{A}_1\mathbf{R} + \mathbf{A}_2\mathbf{M}$ with $\mathbf{M} = [\mathbf{m}_1, \cdots, \mathbf{m}_\ell]$ and randomness $\mathbf{R} = [\mathbf{r}_1, \cdots, \mathbf{r}_\ell]$.

2.2 Arguments of Knowledge

We will now formally define arguments of knowledge. Let R be a polynomial-time-decidable ternary relation. The first input will contain public parameters (a.k.a. common reference string) pp. We define the corresponding language L_{pp} indexed by pp that consists of statement u with a witness w such that $(pp, u, w) \in R$. This is a natural generalisation of standard NP languages, which can be cast as the special case of relations that ignore the first input.

A proof system consists of a PPT parameter generator $\mathcal{K}$, and interactive and stateful PPT algorithms $\mathcal{P}$ and $\mathcal{V}$ used by the prover and verifier. We write $(tr, b) \leftarrow \langle \mathcal{P}(pp, s), \mathcal{V}(pp, t) \rangle$ for running $\mathcal{P}$ and $\mathcal{V}$ on inputs pp, s, and t and getting communication transcript tr and the verifier's decision bit b. We use the convention that $b = 0$ means reject and $b = 1$ means accept.

Definition 2.3. *Proof system $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ is called an* argument of knowledge *for the relation R if it is complete and knowledge sound as defined below.*

Definition 2.4. *$(\mathcal{K}, \mathcal{P}, \mathcal{V})$ has statistical completeness with completeness error $\rho : \mathbb{N} \to [0; 1]$ if for all adversaries $\mathcal{A}$*

$$\Pr\left[\begin{array}{c} pp \leftarrow \mathcal{K}(1^\lambda); (u, w) \leftarrow \mathcal{A}(pp); (tr, b) \leftarrow \langle \mathcal{P}(pp, u, w), \mathcal{V}(pp, u) \rangle : \\ (pp, u, w) \in R \text{ and } b = 0 \end{array} \right] \leq \rho(\lambda).$$

[9] For improved efficiency, one could reduce the number of columns in $\mathbf{A}_1$ and make the commitment scheme computationally-hiding based on the hardness of the LWE problem.

[10] This constant δ is related to the optimal block-size in BKZ reduction [23], which is the currently best way of solving the SIS problem. Presently, the optimal lattice reductions set $\delta \approx 1.005$.

Definition 2.5. $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ *is knowledge sound with knowledge error* $\epsilon : \mathbb{N} \to [0;1]$ *if for all PPT* $\mathcal{P}^*$ *there exists an expected polynomial time extractor* $\mathcal{E}$ *such that for all PPT adversaries* $\mathcal{A}$

$$\Pr\left[\begin{matrix} pp \leftarrow \mathcal{K}(1^\lambda); (u,s) \leftarrow \mathcal{A}(pp); (tr,b) \leftarrow \langle \mathcal{P}^*(pp,u,s), \mathcal{V}(pp,u)\rangle; \\ w \leftarrow \mathcal{E}^{\mathcal{P}^*(pp,u,s)}(pp,u,tr,b) : (pp,u,w) \notin R \text{ and } b = 1 \end{matrix}\right] \leq \epsilon(\lambda).$$

It is sometimes useful to relax the definition of knowledge soundness by replacing R with a relation $\bar{R}$ such that $R \subset \bar{R}$. For instance, in this work, our zero-knowledge proofs of pre-images will have "slack". Thus, even though $\mathbf{v}$ is constructed using $\mathbf{r}, \mathbf{m}$ with coefficients in $\mathbb{Z}_p$, we will only be able to prove knowledge of vectors $\bar{\mathbf{r}}, \bar{\mathbf{m}}$ with larger norms. This extracted commitment is still binding as long as the parameters are set so that the norm of the vector $\begin{bmatrix} \bar{\mathbf{r}} - \bar{\mathbf{r}}' \\ \bar{\mathbf{m}} - \bar{\mathbf{m}}' \end{bmatrix}$ is smaller than the bound in (9).

We say the proof system is *public coin* if the verifier's challenges are chosen uniformly at random independently of the prover's messages. A proof system is special honest-verifier zero-knowledge if it is possible to simulate the proof without knowing the witness whenever the verifier's challenges are known in advance.

Definition 2.6. *A public-coin argument of knowledge* $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ *is said to be* statistical special honest-verifier zero-knowledge (SHVZK) *if there exists a PPT simulator* $\mathcal{S}$ *such that for all interactive and stateful adversaries* $\mathcal{A}$

$$\Pr\left[\begin{matrix} pp \leftarrow \mathcal{K}(1^\lambda); (u,w,\varrho) \leftarrow \mathcal{A}(pp); (tr,b) \leftarrow \langle \mathcal{P}(pp,u,w), \mathcal{V}(pp,u;\varrho)\rangle : \\ (pp,u,w) \in R \text{ and } \mathcal{A}(tr) = 1 \end{matrix}\right]$$

$$\approx \Pr\left[\begin{matrix} pp \leftarrow \mathcal{K}(1^\lambda); (u,w,\varrho) \leftarrow \mathcal{A}(pp); (tr,b) \leftarrow \mathcal{S}(pp,u,\varrho) : \\ (pp,u,w) \in R \text{ and } \mathcal{A}(tr) = 1 \end{matrix}\right],$$

where ϱ *is the randomness used by the verifier.*

2.3 Amortized Proofs of Knowledge

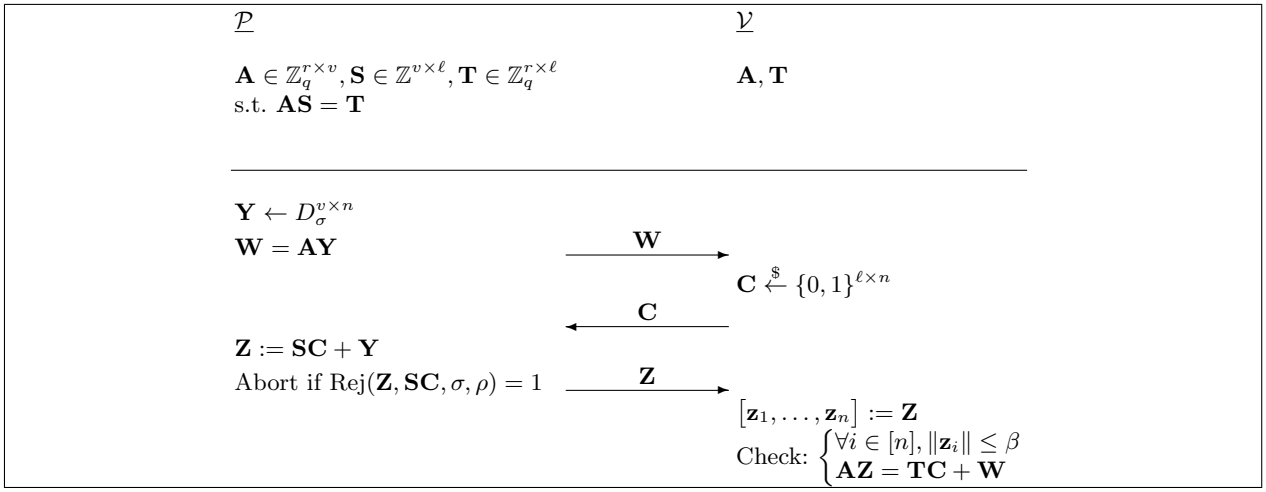

Fig. 2. Amortized proof for ℓ equations.

Baum et al. [5] give an amortized proof of knowledge for preimages of SIS commitments (see Fig. 2). The prover $\mathcal{P}$ wants to prove knowledge of the secret matrix $\mathbf{S}$ such that

$$\mathbf{AS} \equiv \mathbf{T} \pmod{q} \ ,$$

where $\mathbf{A}, \mathbf{T}$ are known to the verifier $\mathcal{V}$.

The protocol begins with $\mathcal{P}$ selecting a "masking" value $\mathbf{Y}$ with small coefficients and sending $\mathbf{W} = \mathbf{AY} \bmod q$. Then $\mathcal{V}$ picks a random challenge matrix $\mathbf{C} \in \{0,1\}^{\ell \times n}$, and sends it to $\mathcal{P}$. Then, $\mathcal{P}$ computes $\mathbf{Z} = \mathbf{SC} + \mathbf{Y}$ and performs a rejection-sampling step (Fig. 3) to make the distribution of $\mathbf{Z}$ independent of $\mathbf{S}$, and if it passes, sends $\mathbf{Z}$ to $\mathcal{V}$. Finally, $\mathcal{V}$ checks that all columns of $\mathbf{Z}$ have small norms and that $\mathbf{AZ} \equiv \mathbf{TC} + \mathbf{W} \pmod q$.

$\underline{\mathrm{Rej}(\mathbf{Z}, \mathbf{B}, \sigma, \rho)}$
01 $u \leftarrow [0,1)$
02 **if** $u > \frac{1}{\rho} \cdot \exp\left(\frac{-2\langle \mathbf{Z}, \mathbf{B} \rangle + \|\mathbf{B}\|^2}{2\sigma^2}\right)$
03 **then return** 0
04 **else**
05 **return** 1

Fig. 3. Rejection Sampling [34,35].

This protocol can be proved zero-knowledge using exactly the same techniques as in [34,35], i.e. Lemma 2.7. One proves knowledge-soundness using a standard heavy-row argument (Lemma A.1).

Lemma 2.7 ([35]). *Let $\mathbf{B} \in \mathbb{Z}^{r \times n}$ be any matrix. Consider a procedure that samples $\mathbf{Y} \leftarrow D_\sigma^{r \times n}$ and then returns the output of $\mathrm{Rej}(\mathbf{Z} := \mathbf{Y} + \mathbf{B}, \mathbf{B}, \sigma, \rho)$ where $\sigma \geq \frac{12}{\ln \rho} \cdot \|\mathbf{B}\|$. The probability that this procedure outputs 1 is within 2^{-100} of $1/\rho$. The distribution of $\mathbf{Z}$, conditioned on the output being 1, is within statistical distance of 2^{-100} of $D_\sigma^{r \times n}$.*

By choosing appropriate parameters (r, v, n, ℓ), Baum et al. obtain a $\tilde{\mathcal{O}}(\sqrt{N})$ proof size for the standard SIS commitment scheme where $N = v\ell$ is the number of entries in the matrix $\mathbf{S}$.

3 Levelled Commitments

In this section, we define levelled lattice commitments and show how to obtain proofs of knowledge with proof size $\tilde{\mathcal{O}}(N^{1/c})$ where N is the number of secrets and c is a constant. Recall that Baum et al. [5] give an amortized proof of knowledge for statements of the form $\mathbf{T} = \mathbf{AS} \bmod q$. We call this a *level-one commitment*. Roughly speaking, the main idea is to apply lattice commitments $c-1$ times to the secret $\mathbf{S}$ in a structured way.

In Appendix B, we extend this result and sketch out the details of an arithmetic circuit satisfiability argument which uses the proof of knowledge based on levelled commitments as a key component.

From now on, we assume that the secret matrix $\mathbf{S}$ already includes the randomness. This not only significantly improves the readability of our protocol, but also ensures that the standard SIS commitment defined in Section 2.1 is both binding and hiding.

3.1 Overview

We define our levelled commitment scheme with d levels for constant d. Let $n, m_0, m_1, ..., m_d, m_{d+1} \in \mathbb{N}$ such that $m_0 = 1$ and $N = m_1 \cdot \ldots \cdot m_{d+1}$. We denote $M_{i,j} = m_i \cdot m_{i+1} \cdot \ldots \cdot m_j$ and for simplicity, we write $M_i = M_{0,i}$. Consider d distinct moduli $q_1 > q_2 > \ldots > q_d$. Let $\mathbf{A}_d, \ldots, \mathbf{A}_1$ be matrices such that $\mathbf{A}_d \in \mathbb{Z}_{q_d}^{n \times m_d}$ and $\mathbf{A}_i \in \mathbb{Z}_{q_i}^{n \times n \cdot m_i}$ for $i \in [d-1]$. Then, the levelled commitment is a function F defined as follows:

$$F_{i,j}(\mathbf{S}) := \begin{cases} \mathbf{A}_i \mathbf{S} \bmod q_i, & \text{if } i = j \\ F_{i,j-1}\left(\left(\mathbf{I}_{M_{i,j-1}} \otimes \mathbf{A}_j\right)\mathbf{S} \bmod q_j\right) & \text{if } i < j. \end{cases} \tag{10}$$

For example, when $d = 2$, the explicit formula for F is

$$F_{1,2}(\mathbf{S}) = \mathbf{A}_1 \cdot ((\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \mathbf{S} \bmod q_2) \bmod q_1. \tag{11}$$

When $d = 3$, the explicit formula for F is

$$F_{1,3}(\mathbf{S}) = \mathbf{A}_1 \cdot ((\mathbf{I}_{m_1} \otimes \mathbf{A}_2)((\mathbf{I}_{m_1 \cdot m_2} \otimes \mathbf{A}_3) \cdot \mathbf{S} \bmod q_3) \bmod q_2) \bmod q_1. \tag{12}$$

Observe that explicit formulae for F written without tensor notation bear some similarity to Merkle trees of SIS commitments. For instance, if $d = 3$ then

$$\mathbf{T} = F_{1,3}\left(\begin{bmatrix} \mathbf{S}_1 \\ \vdots \\ \mathbf{S}_{m_1 m_2} \end{bmatrix}\right) = \mathbf{A}_1 \cdot \begin{bmatrix} \mathbf{A}_2 \cdot \begin{bmatrix} \mathbf{A}_3\mathbf{S}_1 \\ \vdots \\ \mathbf{A}_3\mathbf{S}_{m_2} \end{bmatrix} \\ \vdots \\ \mathbf{A}_2 \cdot \begin{bmatrix} \mathbf{A}_3\mathbf{S}_{m_1(m_2-1)+1} \\ \vdots \\ \mathbf{A}_3\mathbf{S}_{m_1 m_2} \end{bmatrix} \end{bmatrix}.$$

Here, $\mathbf{T}$ represents a commitment to the whole tree. In our protocol, the statement will be

$$F_{1,d}(\mathbf{S}) \equiv \mathbf{T} \pmod{q_1}, \tag{13}$$

where $\mathbf{S}$ is a matrix consisting of small elements.

For readability, let us introduce commitments for intermediate vertices in this tree. We start from the leaves and denote them as $\mathbf{S}_{[i_1,\ldots,i_{d-1}]}$ where each $i_k \in [m_k]$. More concretely, write

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_{[1,\ldots,1,1]} \\ \mathbf{S}_{[1,\ldots,1,2]} \\ \vdots \\ \mathbf{S}_{[1,\ldots,1,m_{d-1}]} \\ \mathbf{S}_{[1,\ldots,2,1]} \\ \vdots \\ \mathbf{S}_{[m_1,\ldots,m_{d-1}]} \end{bmatrix}, \text{ where } \mathbf{S}_{[i_1,\ldots,i_{d-1}]} \in \mathbb{Z}^{m_d \times m_{d+1}}. \tag{14}$$

Now we can define commitments for the intermediate vertices in the commitment tree. Fix $k \in [d-2]$ and recursively define

$$\mathbf{S}_{[i_1,\ldots,i_k]} := (\mathbf{I}_{m_{k+1}} \otimes \mathbf{A}_{k+2}) \begin{bmatrix} \mathbf{S}_{[i_1,\ldots,i_k,1]} \\ \mathbf{S}_{[i_1,\ldots,i_k,2]} \\ \vdots \\ \mathbf{S}_{[i_1,\ldots,i_k,m_{k+1}]} \end{bmatrix} \bmod q_{k+2} \in \mathbb{Z}^{nm_{k+1} \times m_{d+1}}. \tag{15}$$

Let us also set $\mathbf{S}_{[]} := (\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \begin{bmatrix} \mathbf{S}_{[1]} \\ \vdots \\ \mathbf{S}_{[m_1]} \end{bmatrix} \bmod q_2 \in \mathbb{Z}^{nm_1 \times m_{d+1}}$. Then, we have $\mathbf{A}_1\mathbf{S}_{[]} \equiv \mathbf{T} \pmod{q_1}$.

Relaxed Opening. Recall that our protocol aims to prove knowledge of a small matrix $\mathbf{S}$ such that $F_{1,d}(\mathbf{S}) \equiv \mathbf{T} \pmod{q_1}$. However, our extraction algorithm finds a slightly larger (but still small) matrix $\mathbf{S}'$ and additional matrices $\mathbf{R}_1, \ldots, \mathbf{R}_{d-1}$ such that

$$\tilde{F}_{1,d}(\mathbf{S}'; \mathbf{R}_1, \ldots, \mathbf{R}_{d-1}) \equiv \mathbf{T} \pmod{q_1}$$

where $\tilde{F}$ is defined by

$$\tilde{F}_{i,j}\left(\mathbf{S}';\mathbf{R}_i,\ldots,\mathbf{R}_{j-1}\right) := \tilde{F}_{i,j-1}(\mathbf{X} \bmod q_j + q_j\mathbf{R}_{j-1};\mathbf{R}_i,\ldots,\mathbf{R}_{j-2}) \tag{16}$$

and

$$\mathbf{X} := (\mathbf{I}_{M_{i-1,j-1}} \otimes \mathbf{A}_j)\mathbf{S}'$$

for $i < j$ and $\tilde{F}_{i,i}(\mathbf{S}') := (\mathbf{I}_{m_{i-1}} \otimes \mathbf{A}_i)\mathbf{S}' \bmod q_i$. For example, if $d = 2$ then $\tilde{F}_{1,d}$ is defined to be

$$\tilde{F}_{1,d}(\mathbf{S}';\mathbf{R}_1) = \mathbf{A}_1 \cdot ((\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \mathbf{S}' \bmod q_2 + \mathbf{R}_1 \cdot q_2) \bmod q_1 \tag{17}$$

similarly to (3). Clearly, if $\mathbf{R}_1,\ldots,\mathbf{R}_{d-1}$ are all zero matrices then $\tilde{F}_{1,d}\left(\mathbf{S}';\mathbf{R}_1,\ldots,\mathbf{R}_{d-1}\right) = F_{1,d}(\mathbf{S}')$.

We observe that this is enough for practical applications as long as $\mathbf{A}_1,\ldots,\mathbf{A}_d$ are binding. Indeed, one can show, using similar methods as in Section 1.2, that $\tilde{F}_{i,j}$ is binding based on the hardness of SIS for appropriate parameter choice $q_1,\ldots,q_d$ (see Section 3.4).

Formally, given matrices $\mathbf{A}_d,\ldots,\mathbf{A}_1$ such that $\mathbf{A}_d \in \mathbb{Z}_{q_d}^{n\times m_d}$ and $\mathbf{A}_i \in \mathbb{Z}_{q_i}^{n\times n\cdot m_i}$ for $i \in [d-1]$, the relation we give a zero-knowledge proof of knowledge for the relation

$$R = \left\{ \begin{array}{c} (pp,u,w) = \left((\mathbf{q},\mathbf{m},n,B,B_R,\mathbf{A}_1,\ldots,\mathbf{A}_d),\mathbf{T},(\mathbf{S}',\bar{\mathbf{R}})\right) \Big| \\ [\![\|\mathbf{s}_i\| \le 2^d B]\!]_{i\in[m_{d+1}]} \wedge [\![\|\mathbf{R}_i\|_\infty \le B_R]\!]_{i\in[d-1]} \wedge [\![\mathbf{R}_i \in \mathbb{Z}^{nM_i\times m_{d+1}}]\!]_{i\in[d-1]} \\ \wedge\ (\mathbf{S}',\mathbf{T}) \in \mathbb{Z}^{M_d\times m_{d+1}} \times \mathbb{Z}_{q_1}^{n\times m_{d+1}} \wedge\ \tilde{F}_{1,d}\left(\mathbf{S}';\bar{\mathbf{R}}\right) \equiv \mathbf{T} \pmod{q_1} \end{array} \right\}$$

where we denote $\bar{\mathbf{R}} := (\mathbf{R}_1,\ldots,\mathbf{R}_{d-1}), \mathbf{S}' := [\mathbf{s}_1 \ldots \mathbf{s}_{m_{d+1}}], \mathbf{q} := (q_1,\ldots,q_d)$ and $\mathbf{m} := (m_0,\ldots,m_{d+1})$.

3.2 The Main Protocol

We present our zero-knowledge proof of knowledge in Fig. 4. First, we describe supporting algorithms that we will use in the protocol. Firstly, BT_i takes a matrix $\mathbf{Z}$ which has a number of rows divisible by m_i and outputs its *block transpose*:

$$\mathsf{BT}_i(\mathbf{Z}) := \left[\mathbf{Z}_1 \cdots \mathbf{Z}_{m_i}\right], \text{ where } \mathbf{Z} = \begin{bmatrix} \mathbf{Z}_1 \\ \vdots \\ \mathbf{Z}_{m_i} \end{bmatrix}.$$

On the other hand, Fold_i is a recursive algorithm which takes as input M_i matrices $\mathbf{U}_1,\ldots,\mathbf{U}_{M_i}$ and $i+1$ challenge matrices $\mathbf{C}_1,\ldots,\mathbf{C}_{i+1}$. If $i = 0$ then it simply outputs $\mathbf{U}_1\mathbf{C}_1$. Otherwise, it splits the vector $(\mathbf{U}_1,\ldots,\mathbf{U}_{M_i})$ into m_i shorter ones, i.e. $\bar{\mathbf{U}}_j = (\mathbf{U}_{(j-1)M_{i-1}+1},\ldots,\mathbf{U}_{jM_i})$ and runs $\mathbf{U}'_j = \mathsf{Fold}_{i-1}(\bar{\mathbf{U}}_j;\mathbf{C}_1,\ldots,\mathbf{C}_i)$ for each $j \in [m_i]$. Eventually, it outputs $\left[\mathbf{U}'_1 \cdots \mathbf{U}'_{m_i}\right]\mathbf{C}_{i+1}$. We show more properties of this algorithm in the correctness section.

The statement is $F_{1,d}\left(\mathbf{S}\right) \equiv \mathbf{T} \pmod{q_1}$. The protocol begins with the prover $\mathcal{P}$ selecting a masking value for $\mathbf{Y}$ with small coefficients and sending $\mathbf{W} = \mathbf{A}_d\mathbf{Y} \bmod q_d$. In the i-th round, the verifier $\mathcal{V}$ picks a random challenge $\mathbf{C}_i$ and sends it to $\mathcal{P}$. The prover applies Fold to the intermediate commitments

$$\mathbf{V}_i = (\mathbf{S}_{[1,\ldots,1]}, \mathbf{S}_{[2,1,\ldots,1]}, \ldots, \mathbf{S}_{[m_1,1,\ldots,1]}, \mathbf{S}_{[1,2,\ldots,1]}, \ldots, \mathbf{S}_{[m_1,\ldots,m_{i-1}]})$$

as well as all the previous challenges $\mathbf{C}_1,\ldots,\mathbf{C}_i$ sent by $\mathcal{V}$. If $i = d$ then $\mathcal{P}$ also adds $\mathbf{Y}$ and runs rejection sampling. Next, it returns

$$\mathbf{Z}_i = \mathsf{Fold}_{i-1}(\mathbf{V}_i;\mathbf{C}_1,\ldots,\mathbf{C}_i) + [\![i = d]\!]\mathbf{Y}.$$

Finally, the verifier checks that all the $\mathbf{Z}_i$ are small and for all $i \in [d-1]$:

$$\mathbf{A}_{i+1}\mathbf{Z}_{i+1} \equiv \mathsf{BT}_i(\mathbf{Z}_i)\mathbf{C}_{i+1} + [\![i = d-1]\!]\mathbf{W} \pmod{q_{i+1}}.$$

The intermediate vertices and simulation. We provide $\mathbf{S}_{[i_1,\ldots,i_{d-2}]} = \left(\mathbf{I}_{M_{d-1}} \otimes \mathbf{A}_d\right)\mathbf{S} \bmod q_d$ to the simulator when proving zero-knowledge, although this information is not used by the honest verifier algorithm. Consequently, for all $0 \le k < d-1$ and any $(i_1, \ldots i_k) \in [m_1] \times \ldots \times [m_k]$, $\mathbf{S}_{[i_1,\ldots,i_k]}$ is also known to the simulator. We now argue that this is safe.

Consider the Merkle tree of commitments known to the honest prover, with the plaintexts $\mathbf{S}_{[i_1,\ldots,i_{d-1}]}$ at the leaves, and the commitment $\mathbf{T}$ at the root. Only the bottom level of the Merkle tree uses hiding commitments, and we assumed that the plaintexts consist of both messages and randomness for notational convenience. That is, the plaintexts $\mathbf{S}_{[i_1,\ldots,i_{d-1}]}$ already contain the randomness required in order to make $\mathbf{S}_{[i_1,\ldots,i_{d-2}]}$ indistinguishable from random, since $\mathbf{S}_{[i_1,\ldots,i_{d-2}]}$ is a collection of hiding commitments to the plaintexts using the SIS matrix $\mathbf{A}_d$. For $k < d-2$, the values $\mathbf{S}_{[i_1,\ldots,i_k]}$ are simply hashes of $\mathbf{S}_{[i_1,\ldots,i_{k+1}]}$ using SIS matrix $\mathbf{A}_{k+2}$, and do not incorporate any additional randomness.

As such, the root $\mathbf{T}$ does not hide the values of $\mathbf{S}_{[i_1,\ldots i_k]}$ for $k < d-1$, but does hide the plaintexts $\mathbf{S}_{[i_1,\ldots i_{d-1}]}$. Therefore, it is reasonable to give the simulator $\mathbf{S}_{[i_1,\ldots i_k]}$ for $k < d-1$; these are not hidden by $\mathbf{T}$, and could have been safely revealed. The simulator should *not* have access to the plaintexts $\mathbf{S}_{[i_1,\ldots i_{d-1}]}$.

On the other hand, neither the verifier nor the knowledge extractor requires access to $\mathbf{S}_{[i_1,\ldots i_k]}$ for $k < d-1$, and indeed, by the binding property of $\tilde{F}$, the extractor should not be able to produce different values $\mathbf{S}_{[i_1,\ldots i_k]}$ than those known to the simulator, even as part of a relaxed opening.

$\underline{\mathcal{P}_0(pp, \mathbf{S})}$
$\mathbf{Y} \leftarrow D_\sigma^{m_d \times \lambda}$
$\mathbf{W} := \mathbf{A}_d\mathbf{Y} \bmod q_d$
return $(\mathbf{W}, St = \mathbf{Y})$

$\underline{\mathcal{P}_i(pp, (\mathbf{C}_1, \ldots, \mathbf{C}_i), \mathbf{S}, St = \mathbf{Y}) \text{ for } i \in [d]}$
if $i = 1$ **then**
  $\mathbf{V}_i := \mathbf{S}_{[]}$
else $\mathbf{V}_i := (\mathbf{S}_{[1,\ldots,1]}, \mathbf{S}_{[2,1,\ldots,1]}, \ldots, \mathbf{S}_{[m_1,\ldots,m_{i-1}]})$
$\mathbf{V}'_i := \mathsf{Fold}_{i-1}(\mathbf{V}_i; \mathbf{C}_1, \ldots, \mathbf{C}_i)$
if $i = d$ **then**
  $\mathbf{Z}_d := \mathbf{V}'_d + \mathbf{Y}$
  Abort if $\mathrm{Rej}(\mathbf{Z}_d, \mathbf{V}'_d, \sigma, \rho) = 1$
else $\mathbf{Z}_i := \mathbf{V}'_i$
return $(\mathbf{Z}_i, St = \mathbf{Y})$

$\underline{\mathsf{Fold}_i(\mathbf{U}_1, \ldots, \mathbf{U}_{M_i}; \mathbf{C}_1, \ldots, \mathbf{C}_{i+1})}$
if $i = 0$
  then return $\mathbf{U}_1\mathbf{C}_1$
for $j \in [m_i]$
  $\mathbf{U}'_j := \mathsf{Fold}_{i-1}(\mathbf{U}_{(j-1)M_{i-1}+1}, \ldots, \mathbf{U}_{jM_{i-1}}; \mathbf{C}_1, \ldots, \mathbf{C}_i)$
return $\begin{bmatrix}\mathbf{U}'_1 \cdots \mathbf{U}'_{m_i}\end{bmatrix}\mathbf{C}_{i+1}$

$\underline{\mathsf{BT}_i(\mathbf{Z})}$
Write $\mathbf{Z} = \begin{bmatrix}\mathbf{Z}_1 \\ \vdots \\ \mathbf{Z}_{m_i}\end{bmatrix}$
return $\begin{bmatrix}\mathbf{Z}_1 \cdots \mathbf{Z}_{m_i}\end{bmatrix}$

$\underline{\mathcal{V}(pp, \mathbf{T}, (\mathbf{C}_i, \mathbf{Z}_i) \text{ for } i \in [d])}$
$\begin{bmatrix}\mathbf{z}_1, \ldots, \mathbf{z}_{m_{d+1}}\end{bmatrix} := \mathbf{Z}_d$
Check:
  1. $\forall j \in [m_{d+1}], \|\mathbf{z}_j\| \le B$
  2. $\forall i \in [d-1], \|\mathbf{Z}_i\|_\infty \le M_{i-1} m_{d+1} \lambda^{i-1} \frac{q_{i+1}-1}{2}$
  3. $\mathbf{A}_1\mathbf{Z}_1 \equiv \mathbf{T}\mathbf{C}_1 \pmod{q_1}$
  4. $\forall i \in [d-2], \mathbf{A}_{i+1}\mathbf{Z}_{i+1} \equiv \mathsf{BT}_i(\mathbf{Z}_i)\mathbf{C}_{i+1} \pmod{q_{i+1}}$
  5. $\mathbf{A}_d\mathbf{Z}_d \equiv \mathsf{BT}_{d-1}(\mathbf{Z}_{d-1})\mathbf{C}_d + \mathbf{W} \pmod{q_d}$

Fig. 4. Levelled Lattice Commitment Protocol.

3.3 Security Analysis

We start by proving certain properties of the Fold algorithm defined in Fig. 4. They will be crucial when proving correctness of our protocol.

Lemma 3.1. *Let $i \in [d]$ and $k, \ell, m \in \mathbb{N}$. Take arbitrary $\mathbf{U}_1, \ldots, \mathbf{U}_{M_i} \in \mathbb{Z}^{k \times m_{d+1}}$ and $\mathbf{C}_1, \ldots, \mathbf{C}_{i+1}$ such that $\mathbf{C}_1 \in \{0,1\}^{m_{d+1} \times \lambda}$ and for $j > 1$, $\mathbf{C}_j \in \{0,1\}^{m_{j-1}\lambda \times \lambda}$. Then, the following hold.*

(i) There exist matrices $\mathbf{D}_1, \ldots, \mathbf{D}_{M_i} \in \mathbb{Z}^{m_{d+1} \times \lambda}$ *such that* $\|\mathbf{D}_i\|_\infty \leq \lambda^i$ *and*

$$\mathsf{Fold}_i(\mathbf{U}_1, \ldots, \mathbf{U}_{M_i}; \mathbf{C}_1, \ldots, \mathbf{C}_{i+1}) = \sum_{t=1}^{M_i} \mathbf{U}_t \mathbf{D}_t.$$

(ii) For all $\mathbf{A} \in \mathbb{Z}^{m \times k}$,

$$\mathbf{A} \cdot \mathsf{Fold}_i(\mathbf{U}_1, \ldots, \mathbf{U}_{M_i}; \mathbf{C}_1, \ldots, \mathbf{C}_{i+1}) = \mathsf{Fold}_i(\mathbf{A}\mathbf{U}_1, \ldots, \mathbf{A}\mathbf{U}_{M_i}; \mathbf{C}_1, \ldots, \mathbf{C}_{i+1}).$$

(iii) Suppose that each $\mathbf{U}_j$ *can be written as* $\mathbf{U}_j = \begin{bmatrix} \mathbf{U}_{j,1} \\ \vdots \\ \mathbf{U}_{j,\ell} \end{bmatrix}$, *where all matrices* $\mathbf{U}_{j,j'}$ *have the same dimensions.*

Then:

$$\mathsf{Fold}_i(\mathbf{U}_1, \ldots, \mathbf{U}_{M_i}; \mathbf{C}_1, \ldots, \mathbf{C}_{i+1}) = \begin{bmatrix} \mathsf{Fold}_i(\mathbf{U}_{1,1}, \ldots, \mathbf{U}_{M_i,1}; \mathbf{C}_1, \ldots, \mathbf{C}_{i+1}) \\ \vdots \\ \mathsf{Fold}_i(\mathbf{U}_{1,\ell}, \ldots, \mathbf{U}_{M_i,\ell}; \mathbf{C}_1, \ldots, \mathbf{C}_{i+1}) \end{bmatrix}.$$

Proof. See Section A.1.

We are now ready to prove security properties of our protocol.

Theorem 3.2. *Let* $s \geq \max_{i_1,\ldots,i_{d-1}} s_1(\mathbf{S}_{[i_1,\ldots,i_{d-1}]})$, $\rho > 1$ *be a constant,* $\sigma \in \mathbb{R}$ *be such that* $\sigma \geq \frac{12}{\ln \rho} M_{d-1} s \lambda^{d-1} \sqrt{m_{d+1}\lambda}$, *and* $B = \sqrt{2m_d}\sigma$. *Then the protocol described in Fig. 4 is a zero-knowledge proof of knowledge for* R.

Proof. We prove correctness and zero-knowledge, and prove knowledge soundness separately in Theorem 3.3.

Correctness. If $\mathcal{P}$ and $\mathcal{V}$ are honest then the probability of abort is exponentially close to $1 - 1/\rho$ (see Lemma 2.7). Indeed, note that by Lemma 3.1 (i) and the triangle inequality we know that $\|V_d'\|$ is bounded above by $M_{d-1} s \lambda^{d-1} \sqrt{m_{d+1}\lambda}$. In a similar manner, one can show that the second verification condition is satisfied. Now, we show that the equations verified by $\mathcal{V}$ are true.

Firstly, note that

$$\mathbf{A}_1 \mathbf{Z}_1 = \mathbf{A}_1 \mathsf{Fold}_0(\mathbf{S}_{[]}; \mathbf{C}_1) = \mathbf{A}_1 \mathbf{S}_{[]} \mathbf{C}_1 \equiv \mathbf{T}\mathbf{C}_1 \pmod{q_1}.$$

Now, fix $i \in [d-1]$. We know that $\mathbf{Z}_i = \mathsf{Fold}_{i-1}(\mathbf{V}_i; \mathbf{C}_1, \ldots, \mathbf{C}_i)$ (line 7) where

$$\mathbf{V}_i = (\mathbf{S}_{[1,\ldots,1]}, \mathbf{S}_{[2,1,\ldots,1]}, \ldots, \mathbf{S}_{[m_1,1,\ldots,1]}, \mathbf{S}_{[1,2,\ldots,1]}, \ldots, \mathbf{S}_{[m_1,\ldots,m_{i-1}]}).$$

By definition, each $\mathbf{S}_{[j_1,\ldots,j_{i-1}]}$ is equal to

$$\begin{bmatrix} \mathbf{A}_{i+1} \mathbf{S}_{[j_1,\ldots,j_{i-1},1]} \\ \vdots \\ \mathbf{A}_{i+1} \mathbf{S}_{[j_1,\ldots,j_{i-1},m_i]} \end{bmatrix}.$$

By Lemma 3.1 (ii) and (iii), we have

$$\mathbf{Z}_i = \begin{bmatrix} \mathbf{A}_{i+1} \mathsf{Fold}_{i-1}(\mathbf{V}_{i,1}; \mathbf{C}_1, \ldots, \mathbf{C}_i) \\ \vdots \\ \mathbf{A}_{i+1} \mathsf{Fold}_{i-1}(\mathbf{V}_{i,m_i}; \mathbf{C}_1, \ldots, \mathbf{C}_i) \end{bmatrix},$$

where

$$\mathbf{V}_{i,j} = (\mathbf{S}_{[1,\ldots,1,j]}, \mathbf{S}_{[2,1,\ldots,1,j]}, \ldots, \mathbf{S}_{[m_1,1,\ldots,1,j]}, \mathbf{S}_{[1,2,\ldots,1,j]}, \ldots, \mathbf{S}_{[m_1,\ldots,m_{i-1},j]}).$$

Observe that $\mathbf{V}_{i+1}$ is indeed equal to the concatenation of vectors $\mathbf{V}_{i,1}, \ldots \mathbf{V}_{i,m_i}$. Then, by applying the BT function to $\mathbf{Z}_i$ and by definition of Fold, we obtain:

$$\begin{aligned} \mathsf{BT}_i(\mathbf{Z}_i)\mathbf{C}_{i+1} &= \left[\mathbf{A}_{i+1}\bar{\mathbf{V}}_1 \cdots \mathbf{A}_{i+1}\bar{\mathbf{V}}_{m_i}\right] \mathbf{C}_{i+1} \\ &= \mathbf{A}_{i+1} \left[\bar{\mathbf{V}}_1 \cdots \bar{\mathbf{V}}_{m_i}\right] \mathbf{C}_{i+1} \\ &= \mathbf{A}_{i+1} \mathsf{Fold}_i(\mathbf{V}_{i+1}; \mathbf{C}_1, \ldots, \mathbf{C}_{i+1}) \\ &= \mathbf{A}_{i+1} \mathbf{Z}_{i+1}, \end{aligned} \tag{18}$$

where $\bar{\mathbf{V}}_j := \mathsf{Fold}_{i-1}(\mathbf{V}_{i,j}; \mathbf{C}_1, \ldots, \mathbf{C}_i)$. The last verification equation is also satisfied using the same argument as before and noting that $\mathbf{A}_d\mathbf{Y} = \mathbf{W}$.

Eventually, since each coefficient of $\mathbf{Z}$ is statistically close to D_σ, then according to (8) we have $\|\mathbf{z}_i\| \leq \sqrt{2m_d}\sigma$ with overwhelming probability.

Honest-Verifier Zero-Knowledge. We will now prove that our protocol is honest-verifier zero-knowledge. More concretely, we show that it is zero-knowledge when the prover does not abort prior to sending $\mathbf{Z}_d$. We recall that for all $0 \leq k < d-1$, $\mathbf{S}_{[i_1,\ldots,i_k]}$ is known to adversaries.

Define a simulator $\mathcal{S}$ as follows. Given uniformly random $\mathbf{C}_1 \xleftarrow{\$} \{0,1\}^{m_{d+1}\times\lambda}$ and $\mathbf{C}_j \xleftarrow{\$} \{0,1\}^{m_{j-1}\lambda\times\lambda}$ for $j = 2, \ldots, d$, $\mathcal{S}$ samples $\mathbf{Z}_d \leftarrow D_\sigma^{M_{d-1}\times\lambda}$. Then, for $i \in [d-1]$, the simulator sets $\mathbf{Z}_i := \mathsf{Fold}_{i-1}(\mathbf{V}_i; \mathbf{C}_1, \ldots, \mathbf{C}_i)$ where

$$\mathbf{V}_i = (\mathbf{S}_{[1,\ldots,1]}, \mathbf{S}_{[2,1,\ldots,1]}, \ldots, \mathbf{S}_{[m_1,1,\ldots,1]}, \mathbf{S}_{[1,2,\ldots,1]}, \ldots, \mathbf{S}_{[m_1,\ldots,m_{i-1}]}).$$

Finally, $\mathcal{S}$ sets $\mathbf{W} := \mathbf{A}_d\mathbf{Z}_d - \mathsf{BT}_{d-1}(\mathbf{Z}_{d-1})\mathbf{C}_d$ and outputs

$$(\mathbf{W}, \mathbf{C}_1, \mathbf{Z}_1, \ldots, \mathbf{C}_d, \mathbf{Z}_d).$$

It is clear that $\mathcal{V}$ verifies with overwhelming probability. We already argued in the section on correctness that in the real protocol when no abort occurs the distribution of $\mathbf{Z}_d$ is within statistical distance 2^{-100} of $D_\sigma^{M_{d-1}\times\lambda}$. Since $\mathbf{W}$ is completely determined by $\mathbf{A}_d, \mathbf{Z}_{d-1}, \mathbf{Z}_d, \mathbf{C}_d$ and additionally, the distribution of $\mathbf{Z}_i$ output by $\mathcal{S}$ is identical to the one in the real protocol for $i \in [d-1]$, the distribution of $(\mathbf{W}, \mathbf{C}_1, \mathbf{Z}_1, \ldots, \mathbf{C}_d, \mathbf{Z}_d)$ output by $\mathcal{S}$ is within 2^{-100} of the distribution of these variables in the actual non-aborting run of the protocol. □

Knowledge Soundness. We describe a knowledge extractor $\mathcal{E}$ which finds small matrices $\mathbf{S}'$ and $\mathbf{R}_1, \ldots, \mathbf{R}_{d-1}$ such that $\mathbf{T} = \tilde{F}_{1,d}(\mathbf{S}'; \mathbf{R}_1, \ldots, \mathbf{R}_{d-1})$.

Theorem 3.3. *For any prover $\mathcal{P}^*$ who succeeds with probability $\varepsilon > 2^{-\lambda+1} \cdot (4dN)^{2d}$ over its random tape $\chi \in \{0,1\}^x$ and the challenge choice $\mathbf{C}_1, \ldots, \mathbf{C}_d$, such that $\mathbf{C}_1 \xleftarrow{\$} \{0,1\}^{m_{d+1}\times\lambda}$ and $\mathbf{C}_j \xleftarrow{\$} \{0,1\}^{m_{j-1}\lambda\times\lambda}$ for $j > 1$, there exists a knowledge extractor $\mathcal{E}$ running in expected time $\mathsf{poly}(\lambda)/\varepsilon$ who can extract $\mathbf{S}'$ and $\mathbf{R}_1, \ldots, \mathbf{R}_{d-1}$ such that*

$$\tilde{F}_{1,d}(\mathbf{S}'; \mathbf{R}_1, \ldots, \mathbf{R}_{d-1}) = \mathbf{T}.$$

Moreover, each column of $\mathbf{S}'$ has norm at most $2^d B$ and $\forall k \in [d-1]$,

$$\|\mathbf{R}_k\|_\infty \leq 2^k \left(M_{k-1} m_{d+1} \lambda^{k-1} + 2\right) \ .$$

Proof. We provide a sketch of the proof here and include more detail in Section A.2. First, the extractor $\mathcal{E}$ constructs a tree $\mathcal{T}$ of partial transcripts similar to [14,12] where each vertex of $\mathcal{T}$ (apart from the root) is created using extraction techniques from [5] based on the heavy-rows argument. The tree-construction algorithm $\mathsf{TreeConstruct}$ is given in Fig. 7 in Section A.2. Next, $\mathcal{E}$ computes relaxed openings of the levelled commitments, using the algorithm in Fig. 8 in Section A.2.

We sketch some of the steps of the extraction algorithm. First, we can fix $\alpha \in [m_{d+1}]$ and define an extractor $\mathcal{E}$ which finds small vectors $\mathbf{s}', \mathbf{r}_1, \ldots, \mathbf{r}_{i-1}$ such that

$$F_{1,d}(\mathbf{s}'; \mathbf{r}_1, \ldots, \mathbf{r}_{d-1}) = \mathbf{t}_\alpha,$$

where $\mathbf{t}_\alpha$ is the α-th column vector of $\mathbf{T}$ [11]. Then, using the extraction strategy from [5], we can find $\mathbf{Z}'_1, \mathbf{Z}''_1$ such that

$$\mathbf{A}_1(\mathbf{z}'_{1,u} - \mathbf{z}''_{1,u}) \equiv \mathbf{t}_\alpha \pmod{q_1}$$

for some u, where $\mathbf{z}'_{1,u}$ (resp. $\mathbf{z}''_{1,u}$) is the u-th column of $\mathbf{Z}'_1$ (resp. $\mathbf{Z}''_1$). Hence, $\mathcal{E}$ must find a preimage of $\mathbf{z}'_{1,u}$ and $\mathbf{z}''_{1,u}$. We focus on the former. By symmetry, the latter can be obtained analogously.

[11] By collecting extracted solutions for all α, we can merge them and thus obtain the overall solution.

Suppose that we continue running the prover $\mathcal{P}^*$ given the first response $\mathbf{Z}'_1$. We want to get a preimage of the u-th column of $\mathbf{Z}'_1$. Note that when applying BT_1 to $\mathbf{Z}'_1$, the u-th column vector gets split into the u-th, $u+\lambda$-th,..., $u+(m_1-1)\lambda$-th columns. Take arbitrary $j \in \{u+i\lambda : 0 \leq i < m_1\}$. Then, again by rewinding $\mathcal{P}^*$, we can get $\mathbf{Z}'_2, \mathbf{Z}''_2$ such that

$$\mathbf{A}_2\hat{\mathbf{z}}_{2,j} = \mathbf{A}_2(\mathbf{z}'_{2,v} - \mathbf{z}''_{2,v}) \equiv \mathsf{BT}(\mathbf{Z}'_1)_j \pmod{q_2}$$

for some v, where $\hat{\mathbf{z}}_{2,j} := \mathbf{z}'_{2,v} - \mathbf{z}''_{2,v}$ and $\mathsf{BT}_1(\mathbf{Z}'_1)_j$ denotes the j-th column of $\mathsf{BT}_1(\mathbf{Z}'_1)$. By repeating this argument for all possible j, we obtain:

$$(\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \begin{bmatrix} \hat{\mathbf{z}}_{2,u} \\ \vdots \\ \hat{\mathbf{z}}_{2,u+(m_1-1)\lambda} \end{bmatrix} = \begin{bmatrix} \mathsf{BT}_1(\mathbf{Z}'_1)_u \\ \vdots \\ \mathsf{BT}_1(\mathbf{Z}'_1)_{u+(m_1-1)\lambda} \end{bmatrix} = \mathbf{z}'_{1,u} \pmod{q_2}.$$

Observe how the tree structure appears in the argument. We first find $\mathbf{Z}'_1$ and $\mathbf{Z}''_1$ which correspond to the two children of the root. Then, for each such vertex V, we repeat the same argument m_1 times and add new children $W_1, W'_1, \ldots, W_{m_1}, W'_{m_1}$ of V. In general, the tree $\mathcal{T}$ has exactly $2^i M_{i-1}$ vertices on each level $i > 0$.

Eventually, the extracted solution consists of responses which correspond to the leaves of $\mathcal{T}$. We also get additional terms $\mathbf{R}_i$ since each verification equation holds for different moduli. Hence, in order to make any implications from them, we need to first "lift" the previous verification equation and then we can apply it to the next one. The $\mathbf{R}_i$ terms are the result of such lifting. □

3.4 Asymptotic Parameter Choice

In this section, we set parameters for our protocol which minimise the total communication size (see Table 5). More concretely, we pick $q_1, \ldots, q_d$ and $m_1, \ldots, m_{d+1}$ (conditioned on the fact that $N = \prod_{i=1}^{d+1} m_i$ is fixed and $N = \mathcal{O}(\lambda^r)$ for some constant, integer r). For readability, we consider asymptotic parameter choice, neglecting constant terms and focussing on the leading terms using "big-$\mathcal{O}$" notation.

Parameter	Size	Description
λ		Security parameter
p	$\mathsf{poly}(\lambda)$	The largest value of the secrets, i.e. $\|\mathbf{S}\|_\infty$
s		Operator norm of $\mathbf{S}$
N	$m_1 \cdot \ldots \cdot m_{d+1} = \mathsf{poly}(\lambda)$	Number of secrets
n	$d \cdot \mathcal{O}(\log N)$	Number of rows in $\mathbf{A}_1, \ldots, \mathbf{A}_d$
q_i	$\mathcal{O}\left(N^{d-i+2}(2\lambda)^d p^2\right)$	Modulus corresponding to the commitment $\mathbf{A}_i$
m_i	$\left(\mathcal{O}\left(\frac{d\cdot\log N}{d^2\lambda\cdot\log^2 N+\lambda^2}\right)\cdot N\right)^{\frac{1}{d+1}}$	$i-$th dimension of $\mathbf{S}$ for $i \in [d-1]$
m_d	$\mathcal{O}\left(\frac{d^2\cdot\log^2 N+\lambda}{\lambda d\cdot\log N}\right)^{\frac{d}{d+1}} \cdot \left(\frac{N}{\lambda}\right)^{\frac{1}{d+1}}$	$d-$th dimension of $\mathbf{S}$
m_{d+1}	$\left(\mathcal{O}\left(\frac{\lambda^d d\cdot\log N}{d^2\cdot\log^2 N+\lambda}\right)\cdot N\right)^{\frac{1}{d+1}}$	$(d+1)-$th dimension of $\mathbf{S}$
σ	$\frac{12}{\ln\rho} M_{d-1} s \lambda^{d-1}\sqrt{m_{d+1}\lambda}$	Standard deviation for rejection sampling
B	$\sqrt{2m_d}\cdot\sigma$	Soundness slack from proof of knowledge
B_R	$(2\lambda)^d N\cdot\sigma$	Infinity norm of extracted matrices $\mathbf{R}_1, ..., \mathbf{R}_{d-1}$

Fig. 5. Parameter choice for our protocol.

To begin with, we compute simple upper bounds for the norms of the prover's responses. First, let us assume that secret elements in $\mathbf{S}$ have size at most $p < N$, i.e. $\|\mathbf{S}\|_\infty \leq p$. Using the Cauchy-Schwarz inequality and the definition of an operator norm, we get a bound $s \leq Np^2$. Now, we provide a simple bound on B which is defined in Theorem 3.2:

$$B = \sqrt{2m_d}\sigma = \sqrt{2m_d}\cdot\frac{12}{\ln\rho} M_{d-1} s \lambda^{d-1}\sqrt{m_{d+1}\lambda} = \mathcal{O}(\lambda^d N^2 p^2).$$

We note this bound can be substantially improved. Concretely,

$$s \leq m_d m_{d+1} p^2$$

since we only consider the operator norm of $m_d \times m_{d+1}$ matrices in $\mathbb{Z}_p$. By picking the parameters set below, we get $s = \mathcal{O}(\lambda^2 N^{2/d+1})$. However, for readability, we demonstrate a simpler bound.

We know from Theorem 3.3 that for $k \in [d-1]$ we have

$$\|\mathbf{R}_k\|_\infty \leq 2^k \left(M_{k-1} m_{d+1} \lambda^{k-1} + 2\right) \leq (2\lambda)^d N =: B_R.$$

We are ready to set q_d. In order to make $\mathbf{A}_d$ binding and satisfy (9), one needs to pick $q_d > 2\|\mathbf{s}'_i\|$ where $\mathbf{s}'_i$ is the i-th column of the extracted matrix $\mathbf{S}'$ in Theorem 3.2. We know that $\|\mathbf{s}'_i\| \leq 2^d B$ and therefore choose

$$q_d = \mathcal{O}\left((2\lambda)^d N^2 p^2\right).$$

Next, let us fix $i \in [d-1]$ and consider the explicit formula for $F_{i,j}$ in (16) without tensor notation. One observes that each copy of the matrix $\mathbf{A}_i$ is multiplied from the right-hand side by a matrix of the form $\mathbf{U} = (\mathbf{V} \bmod q_{i+1}) + q_{i+1}\mathbf{R}$ and we know that $\|\mathbf{R}\|_\infty \leq (2\lambda)^d N$. Thus, we just need to choose q_i which satisfies

$$q_i > 2N\|\mathbf{U}\|_\infty \geq N \cdot \left(q_{i+1} + 2 \cdot (2\lambda)^d N\right) = N q_{i+1} + 2 \cdot (2\lambda)^d N^2 \ .$$

We solve this recursive formula for q_i and obtain

$$\begin{aligned} q_i &= \mathcal{O}\left(N^{d-i}\left((2\lambda)^d N^2 p^2 + \frac{2 \cdot (2\lambda^4) N^2}{N-1}\right) - \frac{2 \cdot (2\lambda^4) N^2}{N-1}\right) \\ &= \mathcal{O}\left(N^{d+2-i}(2\lambda)^d p^2\right). \end{aligned} \tag{19}$$

Hence, we have $\log q_i \leq \log q_1 = d \cdot \mathcal{O}(\log N)$ for $i \in [d]$. Finally, in order to make all commitments $\mathbf{A}_1, \ldots, \mathbf{A}_d$ satisfy (9), we pick $n = d \cdot \mathcal{O}(\log N)$.

Now, let us set $m_1, \ldots, m_{d+1}$ which minimise the total communication cost of our protocol, including the statement $\mathbf{T}$. First, note that the verifier $\mathcal{V}$ sends

$$\lambda m_{d+1} + \lambda^2 \cdot (m_1 + \ldots + m_{d-1}) \text{bits}$$

as challenges. Next, consider the communication cost from the prover's side. At the beginning, $\mathcal{P}$ sends $\mathbf{W}$ which has $n\lambda \log q_d = \mathcal{O}(d^2 \lambda \log^2 N)$ bits. Since it does not contain any $m_1, \ldots, m_{d+1}$, we ignore this term for now. Next, we note that from the second verification equation, each $\mathbf{Z}_i$ sent by $\mathcal{P}$ satisfies:

$$\log\left(2\|\mathbf{Z}_i\|_\infty\right) \leq \log\left(M_{i-1} m_{d+1} \lambda^{i-1}(q_{i+1} - 1)\right) \leq \log(N\lambda^d q_{i+1}) = d \cdot \mathcal{O}(\log N)$$

for $i \in [d-1]$. On the other hand, with overwhelming probability we have $\|\mathbf{Z}_d\|_\infty \leq 6\sigma = \mathcal{O}(\lambda^d N^2 p^2)$ and thus

$$\log\left(2\|\mathbf{Z}_d\|_\infty\right) = \mathcal{O}(d \log \lambda + \log N) = d \cdot \mathcal{O}(\log N).$$

Therefore, $\mathcal{P}$ sends in total (excluding $\mathbf{W}$)

$$\begin{aligned} n m_{d+1} \log q_1 + \sum_{i=1}^{d-1} n m_i \lambda \log\left(2\|\mathbf{Z}_i\|\right) + m_d \lambda \log\left(2\|\mathbf{Z}_d\|_\infty\right) \\ \leq \left(n m_{d+1} + \sum_{i=1}^{d-1} n m_i \lambda + m_d \lambda\right) d \cdot \mathcal{O}(\log N) \end{aligned} \tag{20}$$

bits. Eventually, this can be upper-bounded by:

$$\sum_{i=1}^{d-1} (n\lambda d \cdot \mathcal{O}(\log N) + \lambda^2) \cdot m_i + \lambda d \cdot \mathcal{O}(\log N) \cdot m_d + (nd \cdot \mathcal{O}(\log N) + \lambda) \cdot m_{d+1}.$$

In order to minimise this expression, we want to set $m_1, \ldots, m_{d+1}$ in such a way that all these $d+1$ terms are (almost) equal. Fix m_{d+1}. Then,

$$m_d = \frac{nd \cdot \mathcal{O}(\log N) + \lambda}{\lambda d \cdot \mathcal{O}(\log N)} \cdot m_{d+1} \text{ and } m_i = \frac{m_{d+1}}{\lambda} \text{ for } i \in [d-1].$$

We compute an exact expression for m_{d+1} as follows:

$$N = \prod_{i=1}^{d+1} m_i = \frac{nd \cdot \mathcal{O}(\log N) + \lambda}{\lambda^d d \cdot \mathcal{O}(\log N)} (m_{d+1})^{d+1}$$

and hence we can set

$$m_{d+1} = \left(\frac{\lambda^d d \cdot \mathcal{O}(\log N)}{nd \cdot \mathcal{O}(\log N) + \lambda} \cdot N \right)^{\frac{1}{d+1}} < \left(\lambda^{d+1} N\right)^{\frac{1}{d+1}} = \lambda \cdot N^{\frac{1}{d+1}}.$$

Then, the total communication cost (now including $\mathbf{W}$) is bounded above by:

$$\begin{aligned} &\mathcal{O}(d^2 \lambda \log^2 N) + (d+1)(nd \cdot \mathcal{O}(\log N) + \lambda) \cdot m_{d+1} \\ &\qquad = \mathcal{O}\left(d^2 \lambda \log^2 N + (d+1)(d^2 \cdot \log^2 N + \lambda) \lambda N^{\frac{1}{d+1}} \right) \\ &\qquad = \mathcal{O}\left(N^{\frac{1}{d+1}} \cdot (d^3 \lambda \log^2 N + d\lambda^2) \right). \end{aligned} \tag{21}$$

To obtain logarithmic proof size, set $d + 1 = \log N$, giving communication cost

$$\lambda \cdot \mathcal{O}\left(\log^5 N + \lambda \log N \right).$$

4 Bulletproofs Folding Protocol

In the discrete logarithm setting, one can apply recursive arguments as in [14,12] and thus obtain logarithmic proof sizes. We show how these techniques can also be used in the lattice setting. Concretely, suppose the statement is as usual $\mathbf{As} = \mathbf{t}$ where $\mathbf{A} \in R^{1 \times k}$, $\mathbf{s} \in R^k$ with $\|\mathbf{s}\|_\infty \leq p$ and $R = \mathbb{Z}[X]/(X^n + 1)$. Then the number of secrets N is equal to kn. We highlight that the only variables which are defined the same in this section and the previous one are λ (security parameter), N (number of secrets) and p (the largest coefficient of the secrets).

We fold the initial statement as follows. Let us write $\mathbf{A}$ and $\mathbf{s}$ as

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{A}_2 \end{bmatrix} \text{ and } \mathbf{s} = \begin{bmatrix} \mathbf{s}_1 \\ \mathbf{s}_2 \end{bmatrix} \text{ where } \mathbf{s}_1, \mathbf{s}_2 \in R^{k/2}.$$

Hence, if we define $\mathbf{l} = \mathbf{A}_1 \mathbf{s}_2 \in R$ and $\mathbf{r} = \mathbf{A}_2 \mathbf{s}_1 \in R$ then for all $c \in R$,

$$(c\mathbf{A}_1 + \mathbf{A}_2)(\mathbf{s}_1 + c\mathbf{s}_2) = c^2\mathbf{l} + c\mathbf{t} + \mathbf{r}$$

. This gives the following proof of knowledge of $\mathbf{s}$.

$\mathcal{P}$		$\mathcal{V}$
$\mathbf{l} = \mathbf{A}_1\mathbf{s}_2, \mathbf{r} = \mathbf{A}_2\mathbf{s}_1$	$\xrightarrow{\mathbf{l}, \mathbf{r}}$	
	$\xleftarrow{c}$	$c \xleftarrow{\$} \{X^i : i \in \mathbb{Z}_{2n}\} \subset R$
$\mathbf{z} = \mathbf{s}_1 + c\mathbf{s}_2$	$\xrightarrow{\mathbf{z}}$	$(c\mathbf{A}_1 + \mathbf{A}_2)\mathbf{z} \stackrel{?}{=} c^2\mathbf{l} + c\mathbf{t} + \mathbf{r}$
		$\|\mathbf{z}\|_\infty \stackrel{?}{\leq} 2p$

The vector $\mathbf{z}$ has length $k/2$, so this protocol has half the communication cost of simply sending $\mathbf{s}$. We can repeat this protocol for the new statement $\mathbf{Bz} = \mathbf{t}'$ where

$$\mathbf{B} = c\mathbf{A}_1 + \mathbf{A}_2 \text{ and } \mathbf{t}' = c^2\mathbf{l} + c\mathbf{t} + \mathbf{r}.$$

Iterating the folding trick down to vectors of length 1 yields a protocol with communication cost $O(\log k)$. Extraction works in principle as follows. First, let us focus on extracting in the one-round protocol presented above. By rewinding we can get three equations

$$(c_i\mathbf{A}_1 + \mathbf{A}_2)\mathbf{z}_i = c_i^2\mathbf{l} + c_i\mathbf{t} + \mathbf{r}, \qquad i = 1, 2, 3$$

for three different challenges c_i and answers $\mathbf{z}_i$. Combine these to obtain

$$\mathbf{A}_1\left(\sum_{i=1}^{3}\lambda_i c_i\mathbf{z}_i\right) + \mathbf{A}_2\left(\sum_{i=1}^{3}\lambda_i\mathbf{z}_i\right) = \sum_{i=1}^{3}\lambda_i c_i^2\mathbf{l} + \sum_{i=1}^{3}\lambda_i c_i\mathbf{t} + \sum_{i=1}^{3}\lambda_i\mathbf{r}. \tag{22}$$

If $\lambda = (\lambda_1, \lambda_2, \lambda_3)^T$ is a solution of the system

$$\begin{pmatrix} c_1^2 & c_2^2 & c_3^2 \\ c_1 & c_2 & c_3 \\ 1 & 1 & 1 \end{pmatrix}\begin{pmatrix}\lambda_1 \\ \lambda_2 \\ \lambda_3\end{pmatrix} = \begin{pmatrix}0 \\ 1 \\ 0\end{pmatrix},$$

then Equation (22) implies

$$\mathbf{A}_1\left(\sum_{i=1}^{3}\lambda_i c_i\mathbf{z}_i\right) + \mathbf{A}_2\left(\sum_{i=1}^{3}\lambda_i\mathbf{z}_i\right) = \mathbf{A}\sum_{i=1}^{3}\lambda_i\begin{bmatrix}c_i\mathbf{z}_i \\ \mathbf{z}_i\end{bmatrix} = \mathbf{t}.$$

Hence, we get a preimage of $\mathbf{t}$ but the problem is that in general it will not be short since λ_i can be large. In order to estimate the size of λ_i, we use the fact that for $i \neq j$, polynomials of the form $2/(X^i - X^j) \in R$ have coefficients in $\{-1, 0, 1\}$ ([9]). Also, we know by the properties of Vandermonde matrices that λ_i are of the form

$$\pm\frac{f}{(X^u - X^v)(X^v - X^w)(X^w - X^u)}$$

for some pairwise distinct $u, v, w \in \mathbb{Z}_{2n}$ and $\|f\|_1 \leq 2$. Therefore, we have $\|8\lambda_i\|_\infty \leq 2n^2$. Hence, we have extracted a solution $\bar{\mathbf{z}}$ which satisfies $\mathbf{A}\bar{\mathbf{z}} = 8\mathbf{t}$ and

$$\|\bar{\mathbf{z}}\|_\infty = \left\|\sum_{i=1}^{3}8\lambda_i\begin{bmatrix}c_i\mathbf{z}_i \\ \mathbf{z}_i\end{bmatrix}\right\|_\infty \leq \sum_{i=1}^{3}\left\|8\lambda_i\begin{bmatrix}c_i\mathbf{z}_i \\ \mathbf{z}_i\end{bmatrix}\right\|_\infty \leq \sum_{i=1}^{3}2n^2 \cdot 2np = 12n^3p.$$

The extractor for the full protocol constructs a tree of partial transcripts similar to [14,12] and applies the strategy we described above at every level. Due to the small soundness error of order $1/n$, the protocol has to be repeated sufficiently many times to achieve negligible soundness error.

Proof size and slack. Let us consider the protocol with $d \leq \log k$ rounds. Then, using the same extraction strategy as above recursively, we obtain a relaxed opening $\bar{\mathbf{z}}$ to the modified equation: $\mathbf{A}\bar{\mathbf{z}} = 8^d\mathbf{t}$ such that

$$\|\bar{\mathbf{z}}\|_\infty = \left((6n^3)^d \cdot 2^d \cdot p\right) = \mathcal{O}\left(n^{3d} \cdot 12^d \cdot p\right) \quad .$$

Therefore, we set $q = \mathcal{O}\left(n^{3d} \cdot 12^d \cdot p\right)$. The proof size is then equal to

$$N\log(2^d p)/2^d + 2dn\log q$$

which is

$$\mathcal{O}(N\log(2^d p)/2^d + d^2 n\log n).$$

Since this gives a soundness error of $O(1/n)$, we repeat the protocol $\lambda/\log n$ times in order to get soundness error $2^{-\lambda}$. This gives a total proof size of

$$\mathcal{O}\left(\frac{\lambda N \log(2^d p)}{2^d \log n} + \lambda d^2 n\right) .$$

Suppose that we follow this protocol all the way down to vectors of length 1, i.e. $d = \log k$. Then, we have a "slack" [12] of

$$\|\bar{\mathbf{z}}\|_\infty = \mathcal{O}\left(n^{3\log N} N^4 p\right)$$

since $k = N/n < N$. The proof size is bounded by

$$\mathcal{O}\left(\lambda n \log N + \lambda n \log^2 N\right).$$

Comparison. We compare the Bulletproofs approach with levelled commitments introduced in Section 3 in terms of both proof sizes and slack. The latter one is not clearly defined in context of levelled commitments since one extracts some secret matrix $\mathbf{S}'$ along with additional terms $\mathbf{R}_1, \mathbf{R}_2, \ldots, \mathbf{R}_{d-1}$ (where d is a number of levels). Therefore, we only focus on the size of $\mathbf{S}'$ and ignore the other terms. We provide a comparison of sizes for both techniques in Fig. 6. Firstly, we observe that none of these methods provide a way to extract

	Bulletproofs	Levelled Commitments
Logarithmic proof size	$\mathcal{O}\left(\lambda n \log N + \lambda n \log^2 N\right)$	$\mathcal{O}\left(\lambda \log^5 N + \lambda^2 \log N\right)$
Corresponding slack	$\mathcal{O}\left(n^{3\log N} N^4 \sqrt{N} p\right)$	$\mathcal{O}(\lambda^{\log N} N^3 p^2)$
$\mathsf{poly}(\lambda, N^{1/c})$ proof size	$\mathcal{O}\left(\lambda N^{1/c} \log N + \lambda n \log^2 N\right)$	$\mathcal{O}\left(N^{1/c} \cdot (c^3 \lambda \log^2 N + c\lambda^2)\right)$
Corresponding slack	$\mathcal{O}\left(n^{3(c-1)\log N/c} \cdot N^{4(c-1)/c} \sqrt{N} \cdot p\right)$	$\mathcal{O}\left((2\lambda)^c N^2 p^2\right)$

Fig. 6. Comparison of lattice Bulletproofs and levelled commitments.

an exact solution to the original equation. Indeed, with lattice commitments we only manage to extract $\mathbf{S}'$ along with extra terms $\mathbf{R}_1, \ldots, \mathbf{R}_{d-1}$ which satisfy (16). On the other hand, with Bulletproofs we extract a relaxed opening $\bar{\mathbf{z}}$ such that $\mathbf{A}\bar{\mathbf{z}} = 8^d \mathbf{t}$. In practice, this implies that the slack we have for $\bar{\mathbf{z}}$ gets also multiplied by the relaxation factor 8^d in front of $\mathbf{t}$. For $d = \log k$, this factor becomes $k^3 = N^3/n^3$.

From Fig. 6 we deduce that Bulletproofs folding offers smaller proof size at the cost of larger slack. Indeed, if one is not limited with any particular amount of slack then one can achieve quadratic-logarithmic proof size as shown on the top-left part of the table. Now, suppose that we can only tolerate $B = N^\alpha$ of slack for some α. The question would be which method achieves smaller proof size given this condition. Note that if $\alpha = 7.5$ then by the argument above, one would simply use Bulletproofs (by setting $n = 2$). Hence, assume that $3 \leq \alpha \leq 7$. For readability, from now on we do not write the "big-$\mathcal{O}$" for each expression. Nevertheless, we still consider asymptotic parameters.

Let us first focus on levelled commitments – we find c such that $(2\lambda)^c N^2 p^2 = B$. Then

$$c = \frac{\log(B/N^2 p^2)}{\log(2\lambda)} \approx (\alpha - 2) \cdot \frac{\log N}{\log(2\lambda)} \approx (\alpha - 2) r$$

where $N = \lambda^r$ for some constant r [13]. Then, the levelled commitments achieve $\tilde{\mathcal{O}}(N^{1/(\alpha-2)r})$ proof size. Now consider the Bulletproofs solution. To begin with, we would like to find d such that $n^{3d} \cdot 12^d \cdot \sqrt{N} p = B$. By solving this equation we have

$$d \approx \frac{\log(B/\sqrt{N})}{3\log n + 4} = \frac{(\alpha - 1/2)\log N}{3\log n + 4} = \gamma \log N$$

[12] Slack here means the Euclidean norm of an extracted solution.
[13] We neglect the $\log p$ term.

where $\gamma = (\alpha - 1/2)/(3\log n + 4)$. Then, the Bulletproofs protocol has $\tilde{\mathcal{O}}(N^{1-\gamma})$ proof size. Therefore, we just need to compare $1 - \gamma$ with $1/(\alpha - 2)r$. The main observation is that for $r \geq 3$, the quadratic function

$$f_r(x) := (15 - 2x)(x - 2)r - 14$$

is positive when $3 \leq x \leq 7$. Hence

$$\frac{1}{(\alpha - 2)r} < \frac{15 - 2\alpha}{14} \leq 1 - \frac{\alpha - 1/2}{3\log n + 4} = 1 - \gamma.$$

This shows that if one is given only a limited (and relatively small) slack, one should consider using the levelled commitments approach to obtain small sub-linear proof sizes.

Acknowledgements

We would like to thank the CRYPTO 2020 reviewers and Keita Xagawa for the useful comments and pointing out the editorial mistakes. This work was supported by the SNSF ERC Transfer Grant CRETP2-166734 – FELICITY. Also, the work was done while Jonathan Bootle was at IBM Research – Zurich.

References

1. Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, STOC '96, pages 99–108, 1996.
2. Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017*, pages 2087–2104, 2017.
3. Thomas Attema, Vadim Lyubashevsky, and Gregor Seiler. Practical product proofs for lattice commitments. *IACR Cryptol. ePrint Arch.*, 2020:517, 2020.
4. Wojciech Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296(1):625–635, Dec 1993.
5. Carsten Baum, Jonathan Bootle, Andrea Cerulli, Rafaël del Pino, Jens Groth, and Vadim Lyubashevsky. Sub-linear lattice-based zero-knowledge arguments for arithmetic circuits. In *Proceedings of the 38th Annual International Cryptology Conference*, CRYPTO'18, pages 669–699, 2018.
6. Eli Ben-Sasson, Iddo Bentov, Alessandro Chiesa, Ariel Gabizon, Daniel Genkin, Matan Hamilis, Evgenya Pergament, Michael Riabzev, Mark Silberstein, Eran Tromer, and Madars Virza. Computational integrity with a public random string from quasi-linear pcps. In *Advances in Cryptology - EUROCRYPT 2017. Proceedings, Part III*, pages 551–579, 2017.
7. Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable zero knowledge with no trusted setup. In *Advances in Cryptology - CRYPTO 2019. Proceedings, Part III*, pages 701–732, 2019.
8. Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. Aurora: Transparent succinct arguments for R1CS. In *Advances in Cryptology - EUROCRYPT 2019. Proceedings, Part I*, pages 103–128, 2019.
9. Fabrice Benhamouda, Jan Camenisch, Stephan Krenn, Vadim Lyubashevsky, and Gregory Neven. Better zero-knowledge proofs for lattice encryption and their application to group signatures. In *Advances in Cryptology - ASIACRYPT 2014. Proceedings, Part I*, pages 551–572, 2014.
10. Daniel J. Bernstein, Andreas Hülsing, Stefan Kölbl, Ruben Niederhagen, Joost Rijneveld, and Peter Schwabe. The sphincs$^+$ signature framework. In *CCS*, pages 2129–2146. ACM, 2019.
11. Dan Boneh, Yuval Ishai, Amit Sahai, and David J. Wu. Quasi-optimal snargs via linear multi-prover interactive proofs. In *Proceedings of the 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, EUROCRYPT'18, pages 222–255, 2018.
12. Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In *Advances in Cryptology – EUROCRYPT 2016, Proceedings*, pages 327–357, 2016.
13. Jonathan Bootle, Vadim Lyubashevsky, and Gregor Seiler. Algebraic techniques for short(er) exact lattice-based zero-knowledge proofs. In *Advances in Cryptology - CRYPTO 2019. Proceedings, Part I*, pages 176–202, 2019.

14. Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *Proceedings of the 39th IEEE Symposium on Security and Privacy*, S&P '18, pages 315–334, 2018.
15. Ivan Damgård. On σ-protocols. *Lecture Notes, University of Aarhus, Department for Computer Science*, 2002.
16. Jelle Don, Serge Fehr, and Christian Majenz. The measure-and-reprogram technique 2.0: Multi-round Fiat-Shamir and more. *CoRR*, abs/2003.05207, 2020.
17. Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Security of the Fiat-Shamir transformation in the quantum random-oracle model. In *CRYPTO (2)*, volume 11693 of *Lecture Notes in Computer Science*, pages 356–383. Springer, 2019.
18. Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(1):238–268, 2018.
19. Léo Ducas, Vadim Lyubashevsky, and Thomas Prest. Efficient identity-based encryption over NTRU lattices. In *ASIACRYPT (2)*, volume 8874 of *Lecture Notes in Computer Science*, pages 22–41. Springer, 2014.
20. Muhammed F. Esgin, Ngoc Khanh Nguyen, and Gregor Seiler. Practical exact proofs from lattices: New techniques to exploit fully-splitting rings. *IACR Cryptol. ePrint Arch.*, 2020:518, 2020.
21. Muhammed F. Esgin, Ron Steinfeld, Joseph K. Liu, and Dongxi Liu. Lattice-based zero-knowledge proofs: New techniques for shorter and faster constructions and applications. In *Advances in Cryptology - CRYPTO 2019. Proceedings, Part I*, pages 115–146, 2019.
22. Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In *Advances in Cryptology - CRYPTO 2018. Proceedings, Part II*, pages 33–62, 2018.
23. Nicolas Gama and Phong Q. Nguyen. Predicting lattice reduction. In *Advances in Cryptology - EUROCRYPT 2008. Proceedings*, pages 31–51, 2008.
24. Rosario Gennaro, Michele Minelli, Anca Nitulescu, and Michele Orrù. Lattice-based zk-snarks from square span programs. In *Proceedings of the 25th ACM Conference on Computer and Communications Security*, CCS'18, pages 556–573, 2018.
25. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.
26. Jens Groth. Linear algebra with sub-linear zero-knowledge arguments. In *Advances in Cryptology - CRYPTO 2009. Proceedings*, pages 192–208, 2009.
27. Jens Groth. Efficient zero-knowledge arguments from two-tiered homomorphic commitments. In *Advances in Cryptology - ASIACRYPT 2011. Proceedings*, pages 431–448, 2011.
28. Jens Groth. On the size of pairing-based non-interactive arguments. In *Advances in Cryptology - EUROCRYPT 2016. Proceedings, Part II*, pages 305–326, 2016.
29. Jens Groth, Markulf Kohlweiss, Mary Maller, Sarah Meiklejohn, and Ian Miers. Updatable and universal common reference strings with applications to zk-snarks. In *Proceedings of the 38th Annual International Cryptology Conference*, CRYPTO'18, pages 698–728, 2018.
30. Akinori Kawachi, Keisuke Tanaka, and Keita Xagawa. Concurrently secure identification schemes based on the worst-case hardness of lattice problems. In *Advances in Cryptology - ASIACRYPT 2008, 14th International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, Australia, December 7-11, 2008. Proceedings*, pages 372–389, 2008.
31. Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. A concrete treatment of fiat-shamir signatures in the quantum random-oracle model. In *EUROCRYPT (3)*, volume 10822 of *Lecture Notes in Computer Science*, pages 552–586. Springer, 2018.
32. Leslie Lamport. Constructing digital signatures from a one-way function, 1979.
33. Qipeng Liu and Mark Zhandry. Revisiting post-quantum fiat-shamir. In *CRYPTO (2)*, volume 11693 of *Lecture Notes in Computer Science*, pages 326–355. Springer, 2019.
34. Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In *Advances in Cryptology - ASIACRYPT 2009. Proceedings*, pages 598–616, 2009.
35. Vadim Lyubashevsky. Lattice signatures without trapdoors. In *Advances in Cryptology - EUROCRYPT 2012. Proceedings*, pages 738–755, 2012.
36. Vadim Lyubashevsky and Daniele Micciancio. Asymptotically efficient lattice-based digital signatures. In *TCC*, volume 4948 of *Lecture Notes in Computer Science*, pages 37–54. Springer, 2008.
37. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 1–23. Springer, 2010.
38. Ralph C. Merkle. A certified digital signature. In *CRYPTO*, volume 435 of *Lecture Notes in Computer Science*, pages 218–238. Springer, 1989.

39. Anca Nitulescu. Lattice-based zero-knowledge snargs for arithmetic circuits. In *Proceedings of the 6th International Conference on Cryptology and Information Security in Latin America*, LATINCRYPT'19, pages 217–236, 2019.
40. Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. FALCON. Technical report, National Institute of Standards and Technology, 2017. https://csrc.nist.gov/projects/post-quantum-cryptography/ round-1-submissions.
41. Victor Shoup. Lower bounds for discrete logarithms and related problems. In *Advances in Cryptology - EUROCRYPT '97. Proceedings*, pages 256–266, 1997.

A Security Proofs

A.1 Proof of Lemma 3.1

First, we prove (i) by induction on i. When $i = 0$, we have $\mathsf{Fold}_0(\mathbf{U}_1; \mathbf{C}_1) = \mathbf{U}_1\mathbf{C}_1$ and $\|\mathbf{C}_1\|_\infty \leq \lambda^0 = 1$. Now, assume that the statement holds for some $i - 1 \geq 0$. By definition, we have

$$\mathsf{Fold}_i(\mathbf{U}_1, \ldots, \mathbf{U}_{M_i}; \mathbf{C}_1, \ldots, \mathbf{C}_{i+1}) = \left[\mathbf{U}'_1 \ldots \mathbf{U}'_{m_i}\right] \mathbf{C}_{i+1}$$

where $\mathbf{U}'_j := \mathsf{Fold}_{i-1}(\mathbf{U}_{(j-1)M_{i-1}+1}, \ldots, \mathbf{U}_{jM_{i-1}}; \mathbf{C}_1, \ldots, \mathbf{C}_i)$. Using the induction hypothesis, we can write

$$\mathbf{U}'_j = \sum_{k=1}^{M_{i-1}} \mathbf{U}_{(j-1)M_{i-1}+k}\mathbf{D}_{(j-1)M_{i-1}+k}$$

for some $\mathbf{D}_{(j-1)M_{i-1}+k}$ such that $\|\mathbf{D}_{(j-1)M_{i-1}+k}\|_\infty \leq \lambda^{i-1}$. Let us write

$$\mathbf{C}_{i+1} = \begin{bmatrix} \mathbf{C}_{i+1,1} \\ \vdots \\ \mathbf{C}_{i+1,m_i} \end{bmatrix} \text{ where } \mathbf{C}_{i+1,j} \in \{0,1\}^{\lambda \times \lambda}.$$

Then,

$$\begin{aligned} &\mathsf{Fold}_i(\mathbf{U}_1, \ldots, \mathbf{U}_{M_i}; \mathbf{C}_1, \ldots, \mathbf{C}_{i+1}) \\ &\qquad = \sum_{j=1}^{m_i} \mathbf{U}'_j \mathbf{C}_{i+1,j} \\ &\qquad = \sum_{j=1}^{m_i} \sum_{k=1}^{M_{i-1}} \mathbf{U}_{(j-1)M_{i-1}+k}\mathbf{D}_{(j-1)M_{i-1}+k}\mathbf{C}_{i+1,j}. \end{aligned} \tag{23}$$

To finish the proof, we observe that for any j, k we have

$$\|\mathbf{D}_{(j-1)M_{i-1}+k}\mathbf{C}_{i+1,j}\|_\infty \leq \lambda^{i-1}\lambda = \lambda^i.$$

Next, let us show (ii) using induction on i. Let $i = 0$. Then, we get

$$\mathbf{A} \cdot \mathsf{Fold}_0(\mathbf{U}_1; \mathbf{C}_1) = \mathbf{A}\mathbf{U}_1\mathbf{C}_1 = \mathsf{Fold}_0(\mathbf{A}\mathbf{U}_1; \mathbf{C}_1).$$

Suppose that the statement holds for some $i - 1 \geq 0$. By definition, we have

$$\begin{aligned} \mathbf{A} \cdot \mathsf{Fold}_i(\mathbf{U}_1, \ldots, \mathbf{U}_{M_i}; \mathbf{C}_1, \ldots, \mathbf{C}_{i+1}) &= \mathbf{A}\left[\mathbf{U}'_1 \ldots \mathbf{U}'_{m_i}\right] \mathbf{C}_{i+1} \\ &= \left[\mathbf{A}\mathbf{U}'_1 \ldots \mathbf{A}\mathbf{U}'_{m_i}\right] \mathbf{C}_{i+1}, \end{aligned} \tag{24}$$

where $\mathbf{U}'_j$ is defined above. By the induction hypothesis, we know that

$$\mathbf{AU}'_j = \mathsf{Fold}_{i-1}(\mathbf{AU}_{(j-1)M_{i-1}+1}, \ldots, \mathbf{AU}_{jM_{i-1}}; \mathbf{C}_1, \ldots, \mathbf{C}_i).$$

Thus, again by the definition of Fold:

$$\left[\mathbf{AU}'_1 \cdots \mathbf{AU}'_{m_i}\right] \mathbf{C}_{i+1} = \mathsf{Fold}_i(\mathbf{AU}_1, \ldots, \mathbf{AU}_{M_i}; \mathbf{C}_1, \ldots, \mathbf{C}_{i+1})$$

hence the result holds.

We now prove (iii), again by induction on i. If $i = 0$ then

$$\mathsf{Fold}_0(\mathbf{U}_1; \mathbf{C}_1) = \mathbf{U}_1\mathbf{C}_1 = \begin{bmatrix} \mathbf{U}_{1,1} \\ \vdots \\ \mathbf{U}_{1,\ell} \end{bmatrix} \mathbf{C}_1 = \begin{bmatrix} \mathbf{U}_{1,1}\mathbf{C}_1 \\ \vdots \\ \mathbf{U}_{1,\ell}\mathbf{C}_1 \end{bmatrix} = \begin{bmatrix} \mathsf{Fold}_0(\mathbf{U}_{1,1}; \mathbf{C}_1) \\ \vdots \\ \mathsf{Fold}_0(\mathbf{U}_{1,\ell}; \mathbf{C}_1) \end{bmatrix}.$$

Assume that the statement is true for some $i - 1 \geq 0$. Then, by definition of Fold we have

$$\mathsf{Fold}_i(\mathbf{U}_1, \ldots, \mathbf{U}_{M_i}; \mathbf{C}_1, \ldots, \mathbf{C}_{i+1}) = \left[\mathbf{U}'_1 \cdots \mathbf{U}'_{m_i}\right] \mathbf{C}_{i+1}$$

where $\mathbf{U}'_j$ is defined above. By the induction hypothesis, we get

$$\mathbf{U}'_j = \begin{bmatrix} \mathsf{Fold}_{i-1}(\mathbf{U}_{(j-1)M_{i-1}+1,1}, \ldots, \mathbf{U}_{jM_{i-1},1}; \mathbf{C}_1, \ldots, \mathbf{C}_i) \\ \vdots \\ \mathsf{Fold}_{i-1}(\mathbf{U}_{(j-1)M_{i-1}+1,\ell}, \ldots, \mathbf{U}_{jM_{i-1},\ell}; \mathbf{C}_1, \ldots, \mathbf{C}_i) \end{bmatrix}.$$

Let $U'_{j,t} = \mathsf{Fold}_{i-1}(\mathbf{U}_{(j-1)M_{i-1}+1,t}, \ldots, \mathbf{U}_{jM_{i-1},t}; \mathbf{C}_1, \ldots, \mathbf{C}_i)$. Note that

$$\left[\mathbf{U}'_{1,t} \cdots \mathbf{U}'_{m_i,t}\right] \mathbf{C}_{i+1} = \mathsf{Fold}_i(\mathbf{U}_{1,t}, \ldots, \mathbf{U}_{M_i,t}; \mathbf{C}_1, \ldots, \mathbf{C}_{i+1}). \tag{25}$$

Therefore, $\mathsf{Fold}_i(\mathbf{U}_1, \ldots, \mathbf{U}_{M_i}; \mathbf{C}_1, \ldots, \mathbf{C}_{i+1})$ is equal to

$$\left[\mathbf{U}'_1 \cdots \mathbf{U}'_{m_i}\right] \mathbf{C}_{i+1} = \begin{bmatrix} \mathbf{U}'_{1,1} & \cdots & \mathbf{U}'_{m_i,1} \\ \vdots & \ldots & \vdots \\ \mathbf{U}'_{1,\ell} & \cdots & \mathbf{U}'_{m_i,\ell} \end{bmatrix} \mathbf{C}_{i+1} = \begin{bmatrix} \left[\mathbf{U}'_{1,1} \cdots \mathbf{U}'_{m_i,1}\right] \mathbf{C}_{i+1} \\ \vdots \\ \left[\mathbf{U}'_{1,\ell} \cdots \mathbf{U}'_{m_i,\ell}\right] \mathbf{C}_{i+1} \end{bmatrix}$$

and thus, the result holds by applying (25). □

A.2 Proof of Theorem 3.3

Denote by $\mathbf{s}'_k$ the k-th column vector of $\mathbf{S}'$ and $\mathbf{r}_{j,k}$ the k-th column vector of $\mathbf{R}_j$. Then, using induction one can show that

$$\tilde{F}_{i,j}\left(\mathbf{S}'; \mathbf{R}_i, \ldots, \mathbf{R}_{j-1}\right) := \left[\mathbf{r}'_1 \ldots \mathbf{r}'_{m_{d+1}}\right],$$

where

$$\mathbf{r}'_k = \tilde{F}_{i,j}\left(\mathbf{s}'_k; \mathbf{r}_{i,k}, \ldots, \mathbf{r}_{j-1,k}\right).$$

Moreover, if we write

$$\mathbf{s}' = \begin{bmatrix} \mathbf{s}'_1 \\ \vdots \\ \mathbf{s}'_{m_{i-1}} \end{bmatrix} \text{ and } \hat{\mathbf{r}}_k = \begin{bmatrix} \hat{\mathbf{r}}_{1,k} \\ \vdots \\ \hat{\mathbf{r}}_{m_{i-1},k} \end{bmatrix} \text{ for } k = i+1, \ldots, j-1,$$

then we also have:

$$\tilde{F}_{i,j}\left(\mathbf{s}'; \hat{\mathbf{r}}_i, \ldots, \hat{\mathbf{r}}_{j-1}\right) = (\mathbf{I}_{m_{i-1}} \otimes \mathbf{A}_i)\bar{\mathbf{X}} \bmod q_i \tag{26}$$

where

$$\bar{\mathbf{X}} = \left(\begin{bmatrix} \tilde{F}_{i+1,j}(\mathbf{s}'_1; \hat{\mathbf{r}}_{1,i+1}, \ldots, \hat{\mathbf{r}}_{1,j-1}) \\ \vdots \\ \tilde{F}_{i+1,j}(\mathbf{s}'_{m_{i-1}}; \hat{\mathbf{r}}_{m_{i-1},i+1}, \ldots, \hat{\mathbf{r}}_{m_{i-1},j-1}) \end{bmatrix} + q_{i+1}\hat{\mathbf{r}}_i \right).$$

This result can be easily shown by induction using the fact that

$$\mathbf{I}_{M_{i-1,k-1}} \otimes \mathbf{A}_k = \mathbf{I}_{m_{i-1}} \otimes (\mathbf{I}_{M_{i,k-1}} \otimes \mathbf{A}_k).$$

Therefore, we can just fix $\alpha \in [m_{d+1}]$ and define an extractor $\mathcal{E}$ which finds small vectors $\mathbf{s}', \mathbf{r}_1, \ldots, \mathbf{r}_{d-1}$ such that

$$\tilde{F}_{1,d}(\mathbf{s}'; \mathbf{r}_1, \ldots, \mathbf{r}_{d-1}) = \mathbf{t}_\alpha,$$

where $\mathbf{t}_\alpha$ is the α-th column vector of $\mathbf{T}$.

Firstly, the extractor $\mathcal{E}$ constructs a tree $\mathcal{T}$ of partial transcripts using the algorithm $\mathsf{TreeConstruct}$ defined in Fig. 7. Informally, a vertex V on level i of $\mathcal{T}$ represents a situation when prover $\mathcal{P}^*$ did already receive certain i challenges and sent i responses. Concretely, each vertex has the following attributes: $\mathsf{chal}, \mathsf{resp}, \mathsf{index}$. The former two represent the last challenge and the last response from $\mathcal{P}^*$ respectively. For simplicity, we denote $\mathsf{resp}_{\mathsf{index}}(V)$ to be the $\mathsf{index}(V)$-th column of the matrix $\mathsf{resp}(V)$. Now, we define children of V by picking candidates for the $i+1$-th challenge using rewinding and techniques from [5], and get the corresponding responses. One notices that the partial transcript (or even a full one if we consider a leaf) can be obtained by starting at some vertex V and collecting values $\mathsf{chal}, \mathsf{resp}$ for each node on the path up to the root. On the other hand, index says for which column of $\mathsf{resp}(V)$ we are interested in finding a corresponding opening. Extractor $\mathcal{E}$ starts by obtaining the masking matrix $\mathbf{W}$ from $\mathcal{P}^*$ and running $\mathsf{TreeConstruct}_0([], \perp, \mathbf{T}, \alpha)$. Let root be the root of $\mathcal{T}$, also denoted as $\mathcal{T}[]$. Then, set $\mathsf{chal}(\mathsf{root}) = \perp, \mathsf{resp}(\mathsf{root}) = \mathbf{T}$ and $\mathsf{index}(\mathsf{root}) = \alpha$. For a vertex V located on level i [14] of the tree $\mathcal{T}$, we denote it by $\mathcal{T}[\mathbf{j}_1, \ldots, \mathbf{j}_i]$ where each $\mathbf{j}_k \in [m_{k-1}] \times \{0,1\}$ (or $\mathcal{T}[]$ if $V = \mathsf{root}$). Children of V are of form $\mathcal{T}[\mathbf{j}_1, \ldots, \mathbf{j}_i, \mathbf{j}]$ where $\mathbf{j} \in [m_i] \times \{0,1\}$. Left children (resp. right children) of V are the ones for which $\mathbf{j} \in [m_i] \times \{0\}$ (resp. $\mathbf{j} \in [m_i] \times \{1\}$). We define $\mathsf{LeftC}(V)$ to be the vector

$$\begin{bmatrix} \mathbf{z}_0 \\ \vdots \\ \mathbf{z}_{m_i-1} \end{bmatrix}, \text{ where } \mathbf{z}_k = \mathsf{resp}_{\mathsf{index}}(\mathcal{T}[\mathbf{j}_1, \ldots, \mathbf{j}_i, (k,0)]) \ ,$$

and define $\mathsf{RightC}(V)$ in a similar fashion.

Let us compute an upper bound on the probability of the event $\mathtt{abort}$ that $\mathsf{TreeConstruct}_0([], \perp, \mathbf{T}, \alpha)$ aborts. Let V be a vertex on level $d-1$ and denote $\mathtt{abort}(V)$ to be the event that the algorithm aborts in line 20 for V. Also, define $(V_0, \ldots, V_{d-2}, V)$ to be the sequence of vertices on the path from $V_0 := \mathsf{root}$ to V. We know that each V_i, where $i \geq 1$, is either a left child or a right child. Hence, denote $r_1, \ldots, r_\gamma$ to be all the indices such that V_{r_j} is a right child for all $j \in [\gamma]$. Now, in order to argue about $\Pr[\mathtt{abort}(V)]$ we use the generalised heavy-rows lemma which can be easily extended from [15].

Lemma A.1. *Let $K > 1$ and $\mathbf{H} \in \{0,1\}^{\ell \times n}$ for some $n, \ell > 1$, such that a fraction ε of the inputs of $\mathbf{H}$ are 1. We say that a row of $\mathbf{H}$ is "heavy" if it contains a fraction at least ε/K of ones. Then less than $1/K$ of the ones in $\mathbf{H}$ are located in heavy rows.*

We introduce the matrices $\mathbf{H}_i$ where we apply Lemma A.1. First, consider the binary matrix $\mathbf{H}_0$ whose rows are indexed by the value of $(\chi, \mathbf{C}'_1)$ and whose columns are indexed by the value of $(\mathbf{C}'_2, \ldots, \mathbf{C}'_d)$. An entry of $\mathbf{H}_0$ is 1 if $\mathcal{P}^*$ succeeds for the corresponding challenges $\mathbf{C}'_1, \ldots, \mathbf{C}'_d$ and the random tape χ. Here, we say that a row of $\mathbf{H}_0$ is ε'-*heavy* if it contains a fraction of at least ε' ones.

Define $\mathbf{C}_i = \mathsf{Chal}(V_i)$ for $i \in [d-1]$ and $\mathbf{c}^T_{i,j}$ to be the j-th row of $\mathbf{C}_i$ (similarly $\mathbf{c}'^T_{i,j}$ for a random matrix $\mathbf{C}'_i$). Now, we introduce $\mathbf{H}_i$ for $i \geq 1$. It is a binary matrix whose rows are indexed by the value of $(\chi, \mathbf{C}'_{i+1})$ and whose columns are indexed by the value of $(\mathbf{C}'_{i+2}, \ldots, \mathbf{C}'_d)$. The entries of $\mathbf{H}_i$ are defined by

$$\mathbf{H}_i[(\chi, \mathbf{C}'_{i+1})][(\mathbf{C}'_{i+2}, \ldots, \mathbf{C}'_d)] := \mathbf{H}_{i-1}[(\chi, \mathbf{C}_i)][\mathbf{C}'_{i+1}, \ldots, \mathbf{C}'_d].$$

[14] A vertex V is on level i if the distance from the root to V equals i.

$\mathsf{TreeConstruct}_i([\mathbf{j}_1,\ldots,\mathbf{j}_i],\mathbf{C}_i,\mathbf{Z}_i,t)$

$V = \mathcal{T}[\mathbf{j}_1,\ldots,\mathbf{j}_i]$
$\mathsf{chal}(V)=\mathbf{C}_i, \mathsf{resp}(V)=\mathbf{Z}_i, \mathsf{index}(V)=t$
$(\mathsf{root}, V_1,\ldots,V_i)$:=vertices on the path from root to V where $V=V_i$
if $i=d$ **then**
  return
$\mathbf{z}_t := t$-th column vector of $\mathbf{Z}_i$
Write $\mathbf{z}_t = \begin{bmatrix} \mathbf{z}_{t,0} \\ \vdots \\ \mathbf{z}_{t,m_i-1} \end{bmatrix}$
$t_j := t+(j-1)\lambda$ for $j\in[m_i]$
for $j\in[m_i]$:
  Select random $\mathbf{C}'_{i+1}$ and then $\mathbf{C}''_{i+1}$ such that $\forall u\neq t_j, \mathbf{c}'^T_{i+1,u}=\mathbf{c}''^T_{i+1,u}$ and $\mathbf{c}''^T_{i+1,t_j}$ is freshly sampled
  Run $\mathcal{P}^*$ on the $i+1$-th random challenge $\mathbf{C}'_{i+1}$ until it outputs $\mathbf{Z}'_{i+1}$
  Rewind $\mathcal{P}^*$ and re-run it on the $i+1$-th challenge $\mathbf{C}''_{i+1}$ until it outputs $\mathbf{Z}''_{i+1}$
  $T' := (\mathbf{W}, \mathsf{chal}(V_1), \mathsf{resp}(V_1),\ldots,\mathbf{C}'_{i+1},\mathbf{Z}'_{i+1})$
  $T'' := (\mathbf{W}, \mathsf{chal}(V_1), \mathsf{resp}(V_1),\ldots,\mathbf{C}''_{i+1},\mathbf{Z}''_{i+1})$
  $\mathsf{count}=0$
  while $i=d-1$ and $\mathsf{count}<\lambda(4dN)^{2d}/\varepsilon$ and T' is not a valid transcript:
    Rewind $\mathcal{P}^*$ and run $\mathcal{P}^*$ on the new $\mathbf{C}'_d$ until it outputs $\mathbf{Z}'_d$
    $T' = (\mathbf{W}, \mathsf{chal}(V_1), \mathsf{resp}(V_1),\ldots,\mathbf{C}'_d,\mathbf{Z}'_d)$
    $\mathsf{count}=\mathsf{count}+1$
  if $\mathsf{count}\geq\lambda(4dN)^{2d}/\varepsilon$ **then** `abort`
  $\mathsf{count}=0$
  while $i=d-1$ and $\mathsf{count}<2\lambda(4dN)^{2d}/\varepsilon$ and T'' is not a valid transcript:
    Rewind $\mathcal{P}^*$ and run $\mathcal{P}^*$ on $\mathbf{C}''_{i+1}$ such that $\forall u\neq t_j, \mathbf{c}'^T_{i+1,u}=\mathbf{c}''^T_{i+1,u}$ and $\mathbf{c}''^T_{i+1,t_j}$ is freshly sampled
    Get response $\mathbf{Z}''_d$
    $T'' = (\mathbf{W}, \mathsf{chal}(V_1), \mathsf{resp}(V_1),\ldots,\mathbf{C}''_d,\mathbf{Z}''_d)$
    $\mathsf{count}=\mathsf{count}+1$
  if $\mathsf{count}\geq 2\lambda(4dN)^{2d}/\varepsilon$ **then** `abort`
  Let ℓ be an index where $\mathbf{c}'^T_{i+1,t_j}[\ell]\neq\mathbf{c}''^T_{i+1,t_j}[\ell]$ (w.l.o.g. $\mathbf{c}'^T_{i+1,t_j}[\ell]-\mathbf{c}''^T_{i+1,t_j}[\ell]=1$, otherwise swap)
  $\mathsf{TreeConstruct}_{i+1}([\mathbf{j}_1,\ldots,\mathbf{j}_i,(j,0)],\mathbf{C}'_{i+1},\mathbf{Z}'_{i+1},\ell)$
  $\mathsf{TreeConstruct}_{i+1}([\mathbf{j}_1,\ldots,\mathbf{j}_i,(j,1)],\mathbf{C}''_{i+1},\mathbf{Z}''_{i+1},\ell)$

Fig. 7. Construction of a tree $\mathcal{T}$ of partial transcripts for $\mathcal{P}^*$. We denote $\mathbf{c}'^T_{i+1,j}$ (resp. $\mathbf{c}'^T_{i+1,j}$) to be the j-th row of $\mathbf{C}'_{i+1}$ (resp. $\mathbf{C}'_{i+1}$).

Informally, $\mathbf{H}_i$ analyses the success probability of $\mathcal{P}^*$ after $i-1$ challenges $\mathbf{C}_1,\ldots,\mathbf{C}_{i-1}$ have been fixed. Similarly as before, we say that a row in $\mathbf{H}_i$ is ε'-heavy if it contains a fraction of at least ε' ones. For $i\leq d-2$, we define an event $\mathtt{h}_i(\varepsilon')$ that $\mathbf{C}_{i+1}$ is in a ε'-heavy row of $\mathbf{H}_i$.

We also consider alternative matrices $\mathbf{H}'_i$ defined as follows. For $i\geq 0$, $\mathbf{H}'_i$ is a binary matrix whose rows are indexed by the value of

$$(\chi, \mathbf{c}'^T_{i+1,1},\ldots,\mathbf{c}'^T_{i+1,\mathsf{index}(V_i)-1},\mathbf{c}'^T_{i+1,\mathsf{index}(V_i)+1},\ldots,\mathbf{c}'^T_{i+1,\lambda m_i})$$

and whose columns are indexed by the value of $(\mathbf{c}'^T_{i+1,\mathsf{index}(V_i)},\mathbf{C}'_{i+2},\ldots,\mathbf{C}'_d)$. Entries of $\mathbf{H}'_i$ are defined as follows:

$$\begin{aligned}\mathbf{H}'_i[(\chi,\hat{\mathbf{C}}_{i+1,\mathsf{index}(V_i)}][(\mathbf{c}'^T_{i+1,\mathsf{index}(V_i)},\mathbf{C}'_{i+2},\ldots,\mathbf{C}'_d)] :=\\ \mathbf{H}_i[(\chi,\mathbf{C}'_{i+1})][\mathbf{C}'_{i+2},\ldots,\mathbf{C}'_d]\end{aligned} \tag{27}$$

where

$$\hat{\mathbf{C}}_{i+1,\mathsf{index}(V_i)} = (\mathbf{c}'^T_{i+1,1},\ldots,\mathbf{c}'^T_{i+1,\mathsf{index}(V_i)-1},\mathbf{c}'^T_{i+1,\mathsf{index}(V_i)+1},\ldots,\mathbf{c}'^T_{i+1,r_i}),$$

$r_i = \lambda m_i$ for $i \geq 1$, $r_0 = m_{d+1}$ and

$$\mathbf{C}'_{i+1} = \begin{bmatrix} \mathbf{c}^T_{i+1,1} \\ \vdots \\ \mathbf{c}^T_{i+1,r_i} \end{bmatrix} .$$

We need matrices $\mathbf{H}'_i$ for analysing the success probability of $\mathcal{P}^*$ when we send a second challenge which is similar to the one sent earlier (line 10). Let $\mathtt{h}'_i(\varepsilon')$ be the event that row corresponding to

$$(\chi, \mathbf{c}^T_{i+1,1}, \ldots, \mathbf{c}^T_{i+1,\mathsf{index}(V_i)-1}, \mathbf{c}^T_{i+1,\mathsf{index}(V_i)+1}, \ldots, \mathbf{c}^T_{i+1,r_i})$$

is ε'-heavy in $\mathbf{H}'_i$, i.e. it contains a fraction of at least ε' ones.

We are ready to give an upper bound on $\Pr[\mathtt{abort}(V)]$. For readability, let $K = 4dN$. Our strategy is as follows. First, we check if V_1 is a right child. If so, then using Lemma A.1 we have the following bound:

$$\begin{aligned} \Pr[\mathtt{abort}(V)] &\leq \Pr[\mathtt{abort}(V)|\mathtt{h}'_0(\varepsilon/K)] + \Pr[\neg\mathtt{h}'_0(\varepsilon/K)] \\ &< \Pr[\mathtt{abort}(V)|\mathtt{h}'_0(\varepsilon/K)] + 1/K. \end{aligned} \tag{28}$$

Next, we want to bound $\Pr[\mathtt{abort}(V)|\mathtt{h}'_0(\varepsilon/K)]$. Again, we apply Lemma A.1 and get

$$\begin{aligned} \Pr[\mathtt{abort}(V)|\mathtt{h}'_0(\varepsilon/K)] &\leq \Pr\left[\mathtt{abort}(V)|\mathtt{h}'_0(\varepsilon/K) \wedge \mathtt{h}_0\left(\varepsilon/K^2\right)\right] \\ &\quad + \Pr\left[\neg\mathtt{h}_0(\varepsilon/K^2)|\mathtt{h}'_0(\varepsilon/K)\right] \\ &< \Pr\left[\mathtt{abort}(V)|\mathtt{h}'_0(\varepsilon/K) \wedge \mathtt{h}_0\left(\varepsilon/K^2\right)\right] + 1/K. \end{aligned} \tag{29}$$

Thus, if V_1 is indeed a right child then we get

$$\Pr[\mathtt{abort}(V)] < \Pr\left[\mathtt{abort}(V)|\mathtt{h}'_0(\varepsilon/K) \wedge \mathtt{h}_0\left(\varepsilon/K^2\right)\right] + 2/K$$

and otherwise

$$\Pr[\mathtt{abort}(V)] < \Pr\left[\mathtt{abort}(V)|\mathtt{h}_0\left(\varepsilon/K\right)\right] + 1/K.$$

In a similar manner we apply Lemma A.1 for events in the following order: $\mathtt{h}_1, \ldots, \mathtt{h}_{d-2}$ and also $\mathtt{h}'_{r_j-1}$ in-between $\mathtt{h}_{r_j-2}$ and $\mathtt{h}_{r_j-1}$ for each $j \in [\gamma]$. At the end, we have:

$$\Pr[\mathtt{abort}(V)] < \Pr[\mathtt{abort}(V)|\mathtt{E}] + \frac{d-1+\gamma}{K} \leq \Pr[\mathtt{abort}(V)|\mathtt{E}] + \frac{2(d-1)}{K}$$

where

$$\begin{aligned} \mathtt{E} = \mathtt{h}_0\left(\varepsilon/K\right) \wedge \ldots \wedge \mathtt{h}_{r_1-2}\left(\varepsilon/K^{r_1-1}\right) \wedge \mathtt{h}'_{r_1-1}\left(\varepsilon/K^{r_1}\right) \\ \wedge \mathtt{h}_{r_1-1}\left(\varepsilon/K^{r_1+1}\right) \wedge \ldots \wedge \mathtt{h}_{d-2}\left(\varepsilon/K^{d-1+\gamma}\right). \end{aligned} \tag{30}$$

Hence, we just need to bound $\Pr[\mathtt{abort}(V)|\mathtt{E}]$. Since $\mathtt{E}$ holds (and also $\mathtt{h}_{d-2}$), the row of $\mathbf{H}_{d-2}$ indexed by $(\chi, \mathbf{C}_{d-1})$ is heavy. Therefore, the probability that we hit one of the ones in this row is at least $\varepsilon/K^{d-1+\gamma}$. Thus, the algorithm aborts in line 20 with probability at most

$$(1 - \varepsilon/K^{d-1+\gamma})^{\lambda K^{2d}/\varepsilon} < (1 - \varepsilon/K^{2d})^{\lambda K^{2d}/\varepsilon} < e^{-\lambda} < 2^{-\lambda}.$$

Suppose that the algorithm finds a suitable $\mathbf{C}_d := \mathbf{C}'_d$. Then, we have two cases – either $\mathtt{h}'_{d-1}(\epsilon/K^{d+\gamma})$ holds or not. Note that $\Pr[\neg\mathtt{h}'_{d-1}(\epsilon/K^{d+\gamma})|\mathtt{E}] < 1/K$. Let us assume that $\mathtt{h}'_{d-1}(\epsilon/K^{d+\gamma})$ is true, i.e. the row of $\mathbf{H}'_{d-1}$ indexed by

$$(\chi, \mathbf{c}^T_{d,1}, \ldots, \mathbf{c}^T_{d,\mathsf{index}(V)-1}, \mathbf{c}^T_{d,\mathsf{index}(V)+1}, \ldots, \mathbf{c}'^T_{d,\lambda m_{d-1}})$$

is heavy. Thus, the probability that we pick a correct $\mathbf{C}''_d$ is at least $\epsilon/K^{d+\gamma} - 2^{-\lambda}$ because we want a reply for a challenge $\mathbf{C}''_d$ different from $\mathbf{C}'_d$. By assumption on ε, we know that

$$\varepsilon/K^{d+\gamma} - 2^{-\lambda} > \varepsilon/2K^{d+\gamma} > \varepsilon/2K^{2d}.$$

Hence, we can bound the probability that $\mathcal{P}^*$ does not succeed on any of the $2\lambda K^{2d}/\varepsilon$ attempts by

$$\left(1-\varepsilon/2K^{2d}\right)^{2\lambda K^{2d}/\varepsilon} < e^{-\lambda} < 2^{-\lambda}.$$

Eventually, we get

$$\Pr[\mathtt{abort}(V)|\mathtt{E}] < 2\cdot 2^{-\lambda} + 1/K$$

and therefore

$$\Pr[\mathtt{abort}(V)] < 1/2^{\lambda-1} + 2d/K.$$

By the union bound, the probability that the algorithm aborts on any of the vertices of level $d-1$ is at most

$$N/2^{\lambda-1} + 2dN/K = N/2^{\lambda-1} + 1/2$$

since we set $K = 4dN$. Thus, by running $\mathcal{O}(\lambda)$ copies of $\mathsf{TreeConstruct}$ we manage to construct a tree of transcripts. Note that the algorithm itself runs in expected time $\mathsf{poly}(\lambda)/\epsilon$ since $N = \mathsf{poly}(\lambda)$ and we assumed d is a constant term.

We now show a certain relationship between $\mathsf{resp}_{\mathsf{index}}(V)$, $\mathsf{LeftC}(V)$ and $\mathsf{RightC}(V)$.

Lemma A.2. *Let $V = \mathcal{T}[\mathbf{j}_1, \ldots, \mathbf{j}_i]$ be a vertex of a tree $\mathcal{T}$ on level $i < d$. Then*

$$(\mathbf{I}_{m_i} \otimes \mathbf{A}_{i+1})(\mathsf{LeftC}(V) - \mathsf{RightC}(V)) \equiv \mathsf{resp}_{\mathsf{index}}(V) \pmod{q_{i+1}}.$$

Proof. Let $t = \mathsf{index}(V), \mathbf{Z} = \mathsf{resp}(V)$ and $\mathbf{z}_t = \begin{bmatrix} \mathbf{z}_{t,1} \\ \vdots \\ \mathbf{z}_{t,m_i} \end{bmatrix} = \mathsf{resp}_{\mathsf{index}}(V)$. Then, one observes that $\mathbf{z}_{t,j}$ is the $t + (j-1)\lambda$-th column of $\mathsf{BT}(\mathbf{Z})$ for $j \in [m_i]$. Let us first fix j and set $W_0 := \mathcal{T}[\mathbf{j}_1, \ldots, \mathbf{j}_i, (j,0)]$ and $W_1 := \mathcal{T}[\mathbf{j}_1, \ldots, \mathbf{j}_i, (j,1)]$. We will show that

$$\mathbf{A}_{i+1}(\mathsf{resp}_{\mathsf{index}}(W_0) - \mathsf{resp}_{\mathsf{index}}(W_1)) \equiv \mathbf{z}_{t,j} \pmod{q_{i+1}} \ . \tag{31}$$

By following the $\mathsf{TreeConstruct}$ algorithm we get that:

$$\begin{aligned} \mathbf{A}_{i+1}\mathsf{resp}(W_0) &\equiv \mathsf{BT}_i(\mathbf{Z})\mathsf{chal}(W_0) \pmod{q_{i+1}}, \\ \mathbf{A}_{i+1}\mathsf{resp}(W_1) &\equiv \mathsf{BT}_i(\mathbf{Z})\mathsf{chal}(W_1) \pmod{q_{i+1}} \end{aligned} \tag{32}$$

and also matrices $\mathsf{chal}(W_0), \mathsf{chal}(W_1)$ have the same rows apart from the $t + (j-1)\lambda$-th one. By definition of $\mathsf{TreeConstruct}$, $\ell = \mathsf{index}(W_0) = \mathsf{index}(W_1)$ and the ℓ-th coefficient of the $t + (j-1)\lambda$-th row of $\mathsf{chal}(W_0) - \mathsf{chal}(W_1)$ is equal to 1. By subtracting the two equations in (32) and looking at the ℓ-th column of $\mathbf{A}_{i+1}(\mathsf{resp}(W_0) - \mathsf{resp}(W_1))$, we obtain Equation 31.

The lemma follows from applying (31) for all $j \in [m_i]$ and the definitions of $\mathsf{LeftC}(V)$ and $\mathsf{RightC}(V)$. □

After initializing the tree $\mathcal{T}$, extractor $\mathcal{E}$ runs the $\mathsf{Extract}$ algorithm described in Fig. 8 on root. Roughly speaking, given a vertex V, $\mathsf{Extract}$ runs its algorithm recursively for all children of V to obtain their relaxed opening of the levelled commitments. Then it subtracts solutions for left children from the right ones and merges them into a long vector. Clearly, running $\mathsf{Extract}$ from root takes polynomial time in the security parameter λ assuming that $\prod_{i=1}^{d+1} m_i = N = \mathsf{poly}(\lambda)$.

The following lemma shows that the solution $\mathcal{E}$ obtained at the end is indeed a relaxed opening of the levelled commitment.

Lemma A.3. *Let V be a vertex in $\mathcal{T}$ of level $i \le d-1$. Then,*

$$\tilde{F}_{i+1,d}(\mathsf{Extract}_i(V)) \equiv \mathsf{resp}_{\mathsf{index}}(V) \pmod{q_{i+1}} \ .$$

$\mathsf{Extract}_i(V)$

$\mathcal{T}[\mathbf{j}_1, \ldots, \mathbf{j}_i] = V$

if $i = d$ **then return** $\mathsf{resp}_{\mathsf{index}}(V)$

for $j \in [m_i]$:

  $(\mathbf{s}'_{j,0}; \mathbf{r}_{j,0,i+2}, \ldots, \mathbf{r}_{j,0,d-1}) \leftarrow \mathsf{Extract}_{i+1}(\mathcal{T}[\mathbf{j}_1, \ldots, \mathbf{j}_i, (j,0)])$

  $(\mathbf{s}'_{j,1}; \mathbf{r}_{j,1,i+2}, \ldots, \mathbf{r}_{j,1,d-1}) \leftarrow \mathsf{Extract}_{i+1}(\mathcal{T}[\mathbf{j}_1, \ldots, \mathbf{j}_i, (j,1)])$

  if $i < d-1$ **then**

    for $b \in \{0,1\}$:

      $\mathbf{w}_b = \tilde{F}_{i+2,d}(\mathbf{s}'_{j,b}; \mathbf{r}_{j,b,i+2}, \ldots, \mathbf{r}_{j,b,d-1})$

      $\mathbf{r}_{j,b,i+1} := (\mathsf{resp}_{\mathsf{index}}(\mathcal{T}[\mathbf{j}_1, \ldots, \mathbf{j}_i, (j,b)]) - \mathbf{w}) / q_{i+2}$

      $\mathbf{v}_b := (\mathbf{I}_{M_{i+1,d-1}} \otimes \mathbf{A}_d)\mathbf{s}'_{j,b} \bmod q_d$

    for $k = d-1, d-2, \ldots, i+1$:

      $\mathbf{u} := (\mathbf{v}_0 - \mathbf{v}_1 - (\mathbf{v}_0 - \mathbf{v}_1 \bmod q_k)) / q_k$

      $\hat{\mathbf{r}}_{j,k} := \mathbf{r}_{j,0,k} - \mathbf{r}_{j,1,k} + \mathbf{u}$

      if $k > i+1$ **then** $\mathbf{v}_b = (\mathbf{I}_{M_{i+1,k-1}} \otimes \mathbf{A}_k)(\mathbf{v}_b + \mathbf{r}_{j,b,k}) \bmod q_k$ for $b \in \{0,1\}$

return $\hat{\mathbf{s}} = \begin{bmatrix} \mathbf{s}'_{1,0} - \mathbf{s}'_{1,1} \\ \vdots \\ \mathbf{s}'_{m_i,0} - \mathbf{s}'_{m_i,1} \end{bmatrix}$ and $\hat{\mathbf{r}}_k = \begin{bmatrix} \hat{\mathbf{r}}_{1,k} \\ \vdots \\ \hat{\mathbf{r}}_{m_i,k} \end{bmatrix}$ for $k = i+1, \ldots, d-1$

Fig. 8. Extracting relaxed openings of levelled commitments, or more concretely, preimages of $\tilde{F}_{i+1,d}$.

Proof. The proof is by induction on i. Let $i = d-1$ and consider $\mathsf{Extract}_{d-1}(V)$. Note that $\mathbf{s}'_0 = \mathsf{LeftC}(V)$ and $\mathbf{s}'_1 = \mathsf{RightC}(V)$ by definition. Then, by Lemma A.2 we get

$$\tilde{F}_{d,d}(\mathbf{s}'_0 - \mathbf{s}'_1) \equiv \mathsf{resp}_{\mathsf{index}}(V) \pmod{q_d}.$$

Assume the statement holds for $i+1 \leq d-1$ and suppose that

$$(\mathbf{s}'_{j,b}; \mathbf{r}_{j,b,i+2}, \ldots, \mathbf{r}_{j,b,d-1}) \leftarrow \mathsf{Extract}_{i+1}(\mathcal{T}[\mathbf{j}_1, \ldots, \mathbf{j}_i, (j,b)])$$

for each child $\mathcal{T}[\mathbf{j}_1, \ldots, \mathbf{j}_i, (j,b)]$ of V. Then, by induction hypothesis we have

$$\tilde{F}_{i+2,d}(\mathbf{s}'_{j,b}; \mathbf{r}_{j,b,i+2}, \ldots, \mathbf{r}_{j,b,d-1}) + q_{i+2}\mathbf{r}_{j,b,i+1} = \mathsf{resp}_{\mathsf{index}}(\mathcal{T}[\mathbf{j}_1, \ldots, \mathbf{j}_i, (j,b)])$$

for some vector $\mathbf{r}_{j,b,i+1}$. Observe that this equation holds over the integers. Then, by definition of LeftC we get:

$$\mathsf{LeftC}(V) = \begin{bmatrix} \tilde{F}_{i+2,d}(\mathbf{s}'_{1,0}; \mathbf{r}_{1,0,i+2}, \ldots, \mathbf{r}_{1,0,d-1}) + q_{i+2}\mathbf{r}_{1,0,i+1} \\ \vdots \\ \tilde{F}_{i+2,d}(\mathbf{s}'_{m_i,0}; \mathbf{r}_{m_i,0,i+2}, \ldots, \mathbf{r}_{m_i,0,d-1}) + q_{i+2}\mathbf{r}_{m_i,0,i+1} \end{bmatrix}$$

and similarly

$$\mathsf{RightC}(V) = \begin{bmatrix} \tilde{F}_{i+2,d}(\mathbf{s}'_{1,1}; \mathbf{r}_{1,1,i+2}, \ldots, \mathbf{r}_{1,1,d-1}) + q_{i+2}\mathbf{r}_{1,1,i+1} \\ \vdots \\ \tilde{F}_{i+2,d}(\mathbf{s}'_{m_i,1}; \mathbf{r}_{m_i,1,i+2}, \ldots, \mathbf{r}_{m_i,1,d-1}) + q_{i+2}\mathbf{r}_{m_i,1,i+1} \end{bmatrix}.$$

Then, we can write

$$\begin{aligned} &\mathsf{LeftC}(V) - \mathsf{RightC}(V) \\ &\qquad = \begin{bmatrix} \tilde{F}_{i+2,d}(\mathbf{s}'_{1,0} - \mathbf{s}'_{1,1}; \hat{\mathbf{r}}_{1,i+2}, \ldots, \hat{\mathbf{r}}_{1,d-1}) + q_{i+2}\hat{\mathbf{r}}_{1,i+1} \\ \vdots \\ \tilde{F}_{i+2,d}(\mathbf{s}'_{m_i,0} - \mathbf{s}'_{m_i,1}; \hat{\mathbf{r}}_{m_i,i+2}, \ldots, \hat{\mathbf{r}}_{m_i,d-1}) + q_{i+2}\hat{\mathbf{r}}_{m_i,1,i+1} \end{bmatrix} \end{aligned} \tag{33}$$

where $\|\hat{\mathbf{r}}_{j,k}\|_\infty \leq \|\mathbf{r}_{j,0,k}\|_\infty + \|\mathbf{r}_{j,1,k}\|_\infty + 1$ (lines 10-13). Here, we use the fact that for any matrix $\mathbf{A}$ and vectors $\mathbf{u}, \mathbf{v}$ we have: $(\mathbf{A}\mathbf{v} \bmod q) - (\mathbf{A}\mathbf{u} \bmod q) = \mathbf{A}(\mathbf{v} - \mathbf{u}) \bmod q + q\mathbf{r}$ where $\|\mathbf{r}\|_\infty \leq 1$.

Finally, by Lemma A.2, (33) and the property of $\tilde{F}_{i,j}$ described in (26), we get that

$$\begin{aligned} \mathsf{resp}_{\mathsf{index}}(V) &\equiv (\mathbf{I}_{m_i} \otimes \mathbf{A}_{i+1})(\mathsf{LeftC}(V) - \mathsf{RightC}(V)) \pmod{q_{i+1}} \\ &\equiv \tilde{F}_{i+1,d}(\hat{\mathbf{s}}; \hat{\mathbf{r}}_{i+1}, \ldots, \hat{\mathbf{r}}_{d-1}) \pmod{q_{i+1}}, \end{aligned} \tag{34}$$

where

$$\hat{\mathbf{s}} = \begin{bmatrix} \mathbf{s}'_{1,0} - \mathbf{s}'_{1,1} \\ \vdots \\ \mathbf{s}'_{m_i,0} - \mathbf{s}'_{m_i,1} \end{bmatrix} \text{ and } \hat{\mathbf{r}}_k = \begin{bmatrix} \hat{\mathbf{r}}_{1,k} \\ \vdots \\ \hat{\mathbf{r}}_{m_i,k} \end{bmatrix} \text{ for } k = i+1, \ldots, d-1.$$

□

In summary, Lemma A.3 implies that

$$\tilde{F}_{1,d}(\mathsf{Extract}_0(\mathsf{root})) \equiv \mathsf{resp}_{\mathsf{index}}(\mathsf{root}) \equiv \mathbf{t}_k \pmod{q_1}$$

and that $\mathsf{Extract}_0(\mathsf{root})$ is indeed a relaxed opening of $\mathbf{t}_k$.

Eventually, we investigate the sizes that $\mathsf{Extract}$ outputs.

Lemma A.4. *Let V be a vertex of $\mathcal{T}$ on level $i \leq d$ and $(\mathbf{s}'; \mathbf{r}_{i+1}, \ldots, \mathbf{r}_{d-1}) \leftarrow \mathsf{Extract}_i(V)$. Then, $\|\mathbf{s}'\| \leq 2^{d-i}B$ and also for $i \leq d-2$ and $k = i+1, \ldots, d-1$ we have*

$$\|\mathbf{r}_k\|_\infty \leq 2^{k-i}\left(M_{k-1}m_{d+1}\lambda^{k-1} + \sum_{j=0}^{k-i}\frac{1}{2^j}\right) .$$

Proof. The first part can be clearly proven by induction and this follows from the construction of $\mathsf{Extract}$ as well as the triangle inequality. Hence, we focus on the latter part. We prove it by induction on i.

Consider the base case $i = d-2$. Then, $\mathsf{Extract}_i(V)$ returns $(\mathbf{s}'; \mathbf{r}_{d-1})$. For simplicity, we use notation from Fig. 8. Line 08 tells us that

$$\|\mathbf{r}_{j,b,d-1}\|_\infty \leq M_{d-2}m_{d+1}\lambda^{d-2} + 1$$

for $j \in [m_{d-2}]$ and $b \in \{0,1\}$. Indeed, this follows from the second verification equation in Fig. 4 and the triangle inequality. Then, from line 12 we get that

$$\|\hat{\mathbf{r}}_{j,d-1}\|_\infty \leq 2(M_{d-2}m_{d+1}\lambda^{d-2} + 1) + 1 = 2\left(M_{d-2}m_{d+1}\lambda^{d-2} + 1 + \frac{1}{2}\right)$$

since $\|\mathbf{u}\|_\infty \leq 1$. This concludes the base case.

Suppose the statement holds for $i+1 \leq d-2$ and consider $\mathsf{Extract}_i(V)$. Firstly, fix $i+2 \leq k \leq d-1$. Then, using the induction hypothesis (in line 12) along with the triangle inequality we get that for $j \in [m_i]$,

$$\begin{aligned} \|\hat{\mathbf{r}}_{j,k}\|_\infty &\leq 2 \cdot 2^{k-(i+1)}\left(M_{k-1}m_{d+1}\lambda^{k-1} + \sum_{j=0}^{k-(i+1)}\frac{1}{2^j}\right) + 1 \\ &\leq 2^{k-i}\left(M_{k-1}m_{d+1}\lambda^{k-1} + \sum_{j=0}^{k-i}\frac{1}{2^j}\right) . \end{aligned} \tag{35}$$

The only case left is when $k = i+1$. As before, we observe that

$$\|\mathbf{r}_{j,b,i+1}\|_\infty \leq M_i m_{d+1}\lambda^i + 1,$$

and by the triangle inequality in line 12 we get

$$\|\hat{\mathbf{r}}_{j,k}\|_\infty \leq 2(M_i m_{d+1}\lambda^i + 1) + 1 = 2(M_i m_{d+1}\lambda^i + 1 + 1/2).$$

Since the inequalities hold for every $j \in [m_i]$, the result holds. □

The main conclusion of the lemma above is that for $i = 0$,

$$\|\mathbf{r}_k\|_\infty \leq 2^k \left(M_{k-1} m_{d+1} \lambda^{k-1} + 2\right)$$

since $\sum_{j=0}^{\infty} 1/2^j = 2$. Finally, the theorem holds by Lemmas A.2, A.3 and A.4. □

B Applications to Circuit Satisfiability Arguments

We sketch out the details of an arithmetic circuit satisfiability argument which uses the proof of knowledge based on levelled commitments as a key component. The approach is related to that of Baum et al [5] which gives arguments with a square-root communication complexity based on lattices, and that of Groth [27] which gives arguments with a cube-root communication complexity based on 2-level commitment schemes from pairings.

We focus on using the protocol described in Section 1.2, which has a $\tilde{\mathcal{O}}(N^{1/3})$ communication complexity for a secret with N elements. We use it to construct an argument for arithmetic circuit satisfiability which has a $\tilde{\mathcal{O}}(N^{1/3})$ communication complexity, for a circuit with N gates.

It is not too difficult to see that the same ideas can be extended to work in conjunction with the $\tilde{\mathcal{O}}(N^{1/c})$-protocol presented in Section 3 to produce an arithmetic circuit satisfiability argument with $\tilde{\mathcal{O}}(N^{1/c})$ communication complexity.

As in Section 1.2, the honest prover should commit to $\mathbf{S} \in \mathbb{Z}^{m_1 \cdot m_2 \times m_3}$ as

$$\mathbf{A}_1 \cdot ((\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \mathbf{S} \bmod q_2) \bmod q_1 = \mathbf{T}, \tag{36}$$

where $\mathbf{A}_1 \leftarrow \mathbb{Z}_{q_1}^{n \times n m_1}, \mathbf{A}_2 \leftarrow \mathbb{Z}_{q_2}^{n \times m_2}$.

Now, we explain how the verifier can interact with the prover to obtain useful linear combinations of the committed secret values. Then we explain how these linear combinations can be used to check arithmetic circuit satisfiability.

Verifying Linear Combinations of Secrets. Let $\mathbf{\Lambda} \in \mathbb{Z}^{m_3 \times 1}$. If the prover sends the verifier $\mathbf{T}' = \left((\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \bar{\mathbf{S}} \bmod q_2\right) \mathbf{\Lambda}$, then the verifier can check that $\mathbf{T}'$ is the correct linear combination of columns of $(\mathbf{I}_{m_1} \otimes \mathbf{A}_2)\, \mathbf{S} \bmod q_2$. The verifier can do this by checking the equation $\mathbf{A}_1 \mathbf{T}' \equiv \mathbf{T}\mathbf{\Lambda} \bmod q_1$.

Write $\mathbf{S} = \begin{bmatrix} \mathbf{S}_1 \\ \vdots \\ \mathbf{S}_{m_1} \end{bmatrix}$, where each $\mathbf{S}_i$ is an m_2 by m_3 matrix. Denote

$$\mathbf{S}\mathbf{\Lambda} = \begin{bmatrix} \mathbf{S}_1 \mathbf{\Lambda} \\ \vdots \\ \mathbf{S}_{m_1} \mathbf{\Lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{s}'_1 \\ \vdots \\ \mathbf{s}'_{m_1} \end{bmatrix}.$$

Next, note that

$$\mathbf{T}' = ((\mathbf{I}_{m_1} \otimes \mathbf{A}_2)\, \mathbf{S} \bmod q_2)\, \mathbf{\Lambda}$$

is congruent to

$$\begin{bmatrix} \mathbf{A}_2 & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_2 & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{A}_2 \end{bmatrix} \begin{bmatrix} \mathbf{s}'_1 \\ \vdots \\ \\ \vdots \\ \mathbf{s}'_{m_1} \end{bmatrix} \pmod{q_2}.$$

Set $\mathbf{S}' = \left[\mathbf{s}'_1 \dots \mathbf{s}'_{m_1}\right]$. Notice that $\mathbf{A}_2 \mathbf{S}' \equiv \mathsf{BT}_1(\mathbf{T}') \pmod{q_2}$. Let $\mathbf{w} \in \mathbb{Z}^{m_1 \times 1}$. Now, if the prover sends $\mathbf{s}'' = \mathbf{S}'\mathbf{w}$ to the verifier then the verifier can check that $\mathbf{s}''$ is the correct linear combination of columns of $\mathbf{S}'$. The verifier can do this by checking the equation $\mathbf{A}_2 \mathbf{s}'' \equiv \mathsf{BT}_1(\mathbf{T}')\, \mathbf{w} \bmod q_2$.

Consider the form of $\mathbf{s}''$. Write $\mathbf{w} = (w_i)_i$. Then $\mathbf{s}'' = \sum_i w_i \mathbf{s}'_i$. We also know that $\mathbf{s}'_i = \mathbf{S}_i \mathbf{\Lambda}$. Write $\mathbf{\Lambda} = (\Lambda_j)_j$. If we write $\mathbf{S}_i$ as the matrix with columns $\mathbf{s}_{i,j}$, then $\mathbf{s}'_i = \sum_j \mathbf{s}_{i,j} \Lambda_j$. So $\mathbf{s}'' = \sum_{i,j} w_i \mathbf{s}_{i,j} \Lambda_j$.

The prover has sent $\mathbf{T}$, $\mathbf{T}'$ and $\mathbf{s}''$ to the verifier, which have dimensions $n \times m_3$, $nm_1 \times 1$, and m_2 respectively. Since the total number of secret integers in $\mathbf{S}$ is $N = m_1 m_2 m_3$, we can obtain an argument which requires the prover to send $\tilde{\mathcal{O}}(\sqrt[3]{N})$ integers to the verifier by setting $m_1 \approx m_2 \approx m_3 \approx \tilde{\mathcal{O}}(\sqrt[3]{N})$. Figure 5 shows that when using levelled commitments, the modulus sizes of the commitment scheme can also depend polynomially on N, which translates into an extra logarithmic factor of N when the proof size is measured in bits.

Checking Useful Conditions. Previous works [5,26,12] use specially chosen linear combinations of committed secrets to help the verifier check that an arithmetic circuit is satisfiable using a sublinear communication complexity, and in zero-knowledge. Circuits consist of gates of two types. The first type is addition gates and multiplication gates with one public input. Such gates compute affine functions of their inputs. The second type is multiplication gates with neither input public. Thus, circuit satisfiability is often separated into the two sub-tasks of proving that inputs satisfy some linear relations, and proving multiplicative relations between inputs. We sketch out an approach for arithmetic circuits over prime fields.

For $i \in [m_1]$ and $j \in [m_3]$, let $\mathbf{a}_{i,j}$, $\mathbf{b}_{i,j}$ and $\mathbf{c}_{i,j}$ be vectors in $[p]^{m_2}$ for some prime p. Suppose that we want to prove multiplicative relations $\mathbf{a}_{i,j} \circ \mathbf{b}_{i,j} = \mathbf{c}_{i,j}$ for each i and j. Let x and y be uniformly random challenges chosen from $[p]$. For suitable $\mathbf{d}^+_{i,j}, \mathbf{d}^-_{i,j} \in \mathbb{Z}_p^{m_2}$, we can write

$$\begin{aligned}\left(\sum_{i,j} \mathbf{a}_{i,j} (xy)^{i+m_1 j}\right) \circ \left(\sum_{i,j} \mathbf{b}_{i,j} x^{-i-m_1 j}\right) \\ \equiv \left(\sum_{i,j} \mathbf{c}_{i,j} y^{i+m_1 j}\right) + x \left(\sum_{i,j} \mathbf{d}^+_{i,j} x^{i+m_1 j}\right) \\ + x^{-1} \left(\sum_{i,j} \mathbf{d}^-_{i,j} x^{-i-m_1 j}\right) \pmod p.\end{aligned} \tag{37}$$

Note that the $\mathbf{d}^+_{i,j}, \mathbf{d}^-_{i,j}$ will depend on y. Collapsing each bracketed term into a single value, we have

$$\mathbf{a} \circ \mathbf{b} \equiv \mathbf{c} + x\mathbf{d}^+ + x^{-1}\mathbf{d}^- \pmod p.$$

We have seen that the verifier can check that the prover has sent linear combinations of the form

$$\mathbf{s}'' = \sum_{i,j} \lambda_i \mathbf{s}_{i,j} \Lambda_j$$

correctly. If we set $w_i = (xy)^i \mod p$ and $\Lambda_j = (xy)^{m_1 j} \mod p$, then the verifier can obtain $\mathbf{a}$ in a verifiable manner. With other, similar choices of $\mathbf{w}$ and $\mathbf{\Lambda}$, the verifier can obtain $\mathbf{b}, \mathbf{c}, \mathbf{d}^+$ and $\mathbf{d}^-$, and check the equality.

Comparing the terms with x^0 on the left and right hand side, one sees that equality cannot hold unless

$$\sum_{i,j} (\mathbf{c}_{i,j} - \mathbf{a}_{i,j} \circ \mathbf{b}_{i,j})\, y^{i+m_1 j} \equiv 0 \pmod p. \tag{38}$$

If $\mathbf{c}_{i,j} \neq \mathbf{a}_{i,j} \circ \mathbf{b}_{i,j}$ for some (i,j), then the Schwarz-Zippel lemma bounds the probability that (37), and hence (38), still holds, over the random choices of x and y.

Linear relations between different witness values can be handled using similar types of polynomial expressions. Fuller explanations are given in [5,26,12].

A Real Argument. It still remains to see how to build a complete argument. In the first round of the argument, before having seen any random challenge values, the prover commits to the $\mathbf{a}_{i,j}$, $\mathbf{b}_{i,j}$ and $\mathbf{c}_{i,j}$, as three large arrays, $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$. After receiving random challenge y from the verifier, the prover computes and commits to the $\mathbf{d}^+_{i,j}$ and $\mathbf{d}^-_{i,j}$ as two large arrays $\mathbf{D}^+$ and $\mathbf{D}^-$. The prover and verifier use the proof-of-knowledge from Section 1.2 on all commitments. Then the prover and the verifier use the techniques given above to allow the verifier to receive the correct linear combinations of committed values.

With care, it should be possible to interleave the many-round sub-protocols described above. Also, we have not discussed how to achieve zero-knowledge. This could be done by committing to various randomly-sampled masking values, and adding them into the bracketed terms of expression (37) in a way that does not affect the conclusion about the committed values.

The Malicious Prover. In reality, we cannot be sure that the prover is honest, and cannot guarantee that $\mathbf{T}$ is actually a commitment. However, the protocol in Section 3 is knowledge-sound. The knowledge extractor produces $\bar{\mathbf{S}}$ and an $\bar{\mathbf{R}} \in \mathbb{Z}^{n \cdot m_1 \times m_3}$ satisfying the following equation.

$$\mathbf{A}_1 \cdot \left((\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \bar{\mathbf{S}} \bmod q_2 + \bar{\mathbf{R}} \cdot q_2 \right) \bmod q_1 = \mathbf{T}, \tag{39}$$

After receiving $\mathbf{T}'$ from the prover, the verifier checks whether $\mathbf{A}_1\mathbf{T}' = \mathbf{T}\mathbf{\Lambda} \mod q_1$. The collision resistance property associated with $\mathbf{A}_1$ guarantees that

$$\mathbf{T}' = \left((\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \bar{\mathbf{S}} \bmod q_2 \right) \mathbf{\Lambda} + q_2 \cdot \bar{\mathbf{R}}\mathbf{\Lambda}$$

Then we have

$$\mathbf{T}' \bmod q_2 \equiv (\mathbf{I}_{m_1} \otimes \mathbf{A}_2) \cdot \bar{\mathbf{S}}\mathbf{\Lambda} \bmod q_2$$

Therefore, continuing the opening procedure with $\mathbf{T}'$ modulo q_2, we see that the extra terms recovered by the knowledge extractor will have no impact on the verifier's checks.