A Different Base Approach for Better Efficiency on Range Proofs

Esra Günsay[a,*], Cansu Betin Onur[a], Murat Cenk[a]

[a]*Institute of Applied Mathematics,*
Middle East Technical University, Ankara, Turkey

Abstract

Zero-knowledge range proofs (ZKRPs) are commonly used to prove the validation of a secret integer lies in an interval to some other party in a secret way. In many ZKRPs, the secret is represented in binary and then committed via a suitable commitment scheme or represented as an appropriate encryption scheme. This paper is an extended version of the conference paper presented in 14th IEEE International Conference on Security of Information and Networks. To this end, we first analyze the proof proposed by Mao in 1998 in both discrete logarithm-setting and elliptic-curve settings. Mao's proof contains a bit commitment scheme with an OR construction as a sub-protocol. We have extended Mao's range proof to base-u with a modified OR-proof. We investigate and compare the efficiency of different base approaches on Mao's range proof. Later, we analyze the range poof proposed by Bootle et al. in both finite fields and elliptic-curve settings. This proof contains polynomial commitment with matrix row operations. We take the number of computations in modulo exponentiation and the cost of the number of exchanged integers between parties. Then, we generalize these costs for u-based construction. We show that compared with the base-2 representation, different base approach provides efficiency in communication cost or computation cost, or both.

Keywords: Zero knowledge range proof, OR proof, Pedersen commitment.

1. Introduction

The zero-knowledge proofs are used to prove the accuracy of specific information in a secret way and are important building blocks of privacy-preserving systems. A particular and notable example of ZKPs is the zero-knowledge range proofs, which are commonly used to prove a secret integer lies in a given interval. There are various usage areas of ZKRP in the real world, such as e-cash [1, 2], e-voting [3], age validation, risk assessments/credit score systems [4] for banking and financial institutions, investment grading, e-auction [5], group signature schemes [6] and verifiable secret sharing. With the rapid development of blockchain-based distributed ledger technology, ZKRPs have become even more popular since they are used to validate cryptocurrency transactions. Monero [7], Zcash [8], and Zerocoin [2] are just a few examples of cryptocurrencies using ZKRPs for the validation of the transaction process.

The first study to construct a range proof was proposed by Brickell, Chaum, Damgrd & van de Graaf [9], in 1987. They managed to secretly send the disclosed bits to other participants. Their scheme was based on the discrete logarithm together with a *bit commitment*. Their construction has considerable negative features, especially in ranging. In 1995, Damgard [10] proposed a ZKRP scheme. Soon later, in 1997, Fujisaki & Okamoto [11] proposed another construction. Although these proposed constructions work correctly, their use in real-world systems was inefficient. Bellare and Goldwasser [12] presented *the binary decomposition range proof* in 1997. In this construction, the secret s is represented on a modulo-2 basis. In 1998, Chan et al. [1] proposed the CFT proof using the algorithm by Brickell et al. [9]. Nevertheless, their construction only succeeded in the non-negative ranges, and the order of the group has to be unknown. In 2000, Boudot [13] presented a scheme depending on the strong RSA problem. To prove that the secret x lies in the interval $[a,b]$, it is sufficient to show both $x-a$ and $b-x$ are positive.

Using inner product argument techniques presented by Bootle et. al [14], Bünz et. al proposed a novel range proof, namely Bulletproof [15]. Soon later , us-

*Corresponding author.
Email addresses: gunsay@metu.edu.tr (Esra Günsay), cbetin@metu.edu.tr (Cansu Betin Onur), mcenk@metu.edu.tr (Murat Cenk)

ing the zero-knowledge weighted inner product argument technique, Chung et. al improved the Bulletproof and presented Bulletproof+ [16]. Though these proofs are highly efficient and novel, due to their inner product structure, these are not applicable to reconstructing for different base approaches. Therefore, for the scope of our study, we will not try to represent them on different bases.

In almost all ZKRP schemes, the secret is committed at the beginning of the scheme. In the literature, there are three main approaches to commit a secret, which are *integer*, *binary*, or *the u−ary method*. Hence, these methods are listed as follows:

1. *integer method* In this method, the interval is of the form $\mathscr{I} = [a,b]$. Generally, $\mathscr{I}$ is chosen as a large interval space. It is enough to check whether a committed number belongs to $\mathscr{I}$ or not.
2. *binary method* This method allows checking if the committed value of binary represented secret is in the interval $[0,2^k-1]$.
3. *u−ary method* This method represents the message in u-ary base, then checks if the committed value is in the interval $[0,u^k-1]$. In the literature [17], it is stated that this method itself does not reduce the proof size.

Camenisch et al. [17] proposed improvements on the bilinear-group assumption-based set-membership proofs, which also can be used for range proofs. They recommended representing the secret in base-u instead of the folklore base-2 approach. Nevertheless, it is stated that this idea alone does not reduce the proof size; hence, it does not bring any efficiency alone. Consequently, they constructed a scheme that allows reusing the list of u signatures sent by the verifier. The verifier sends the list of u signatures, and the list is used by the prover to check the accuracy. In their system, the range of integers is denoted by $I = [1,u^n]$, where u is the representation base. Elements of I are signed using a digital signature by the verifier. These signatures are considered as common inputs. The prover proves that she knows a signature under the verification key for the element committed to C, while C is a commitment. Moreover, they showed that their approach could also be applicable to the strong RSA-like assumption with a significant level of improvement.

Though different base approaches were studied on bilinear groups and RSA-like schemes, up to our knowledge, only Günsay et. al [18] studied the DLP-based classical proof. However, they did not examine the version of the schemes that uses an elliptic curve in their work. Due to its wide range of usage areas, we investigate the efficiency of different base approaches in different settings. Our aim is to find the most efficient base approach for both computation and communication costs.

1.1. Our Contribution

In 1998, Mao [19] proposed a single verifier correctable binary multi-party secret-sharing construction, which is used to encrypt corresponding primes, using a protocol, namely *a proof of bit length*. We call this protocol as classical range proof. Overall, the secret x is represented in binary, and the binary Pedersen commitment scheme is used thereafter. In this protocol, an OR-Proof proposed in [20] is used as a sub-protocol. This study focuses on the efficiency of different base representations of range proof constructions and generalizes the idea presented by Günsay et al. [18]. We investigate the classical range proof with different base approaches in both discrete logarithm and elliptic curve settings. To this end, we decompose the secret in the $u-$ary method with an adapted sub-protocol. We also analyze the corresponding construction using El-Gamal encryption instead of the Pedersen commitment scheme in both discrete-logarithm and elliptic-curve-based settings. The results show that the base-3 method is more efficient for the classical proof than other base choices in both computation and communication costs. Later, we analyze the range proof proposed by Bootle et al. [21]. This scheme also has a sub-protocol for polynomial commitments. We investigate the computation and communication costs of the corresponding scheme in both finite fields and elliptic-curve-based constructions. The results show that in the finite field setting, while we prove a single instance, the scheme has its lowest computation cost in base-4 with 13.4% efficiency and the lowest communication cost in base-4 with 24% efficiency. In the elliptic-curve-setting, we have the highest computation efficiency with 24.2% improvement while working on base-4.

1.2. Outline of the Paper

Section 2 presents the details of the underlying cryptographic primitives such as commitment schemes, zero-knowledge proofs, and Σ-protocols. The classical range proof proposed by Mao and the base-3 approach to the existing construction, together with a generalization to base-u is described in Section 3. In Section 4, we represent the elliptic curve variant of the scheme with generalization to different bases and comparisons. In Section 5, we described the batched range proof for

low-degree polynomial relations, where the generalizations to different bases are analyzed and compared both for computation and communication.

2. Preliminaries

We introduce the basic primitives, notations, and definitions in this section. Almost in all range proofs, the secret is expressed as a committed value which makes them essential in the context of range proofs. Therefore, after giving a short notations part, we explain the Pedersen commitment scheme. Later, we define the $\Sigma-$protocols and their OR-composition due to their significant role in the classical range proof.

2.1. Notations

Over the ring of integers $\mathbb{Z}$, let s, p, q be large primes such that $q = 2s+1$ and $p = kq+1$ hold where k is an even positive integer. Let $g \in \mathbb{Z}_p^*$ be an element of order q, and G be a group generated by g. We know that $g^q = 1 \bmod p$. Let $f \in G$ be a fixed element generated by a pseudo-random generator which is seeded by g, and its discrete logarithm in base g, $\log_g f$ is unknown. Throughout the paper, $r \in_R \mathbb{Z}_p^*$ denotes that r is randomly chosen in $\mathbb{Z}_p^*$. For randomly chosen r, we say that $E = \mathrm{Com}(x,r) = g^x f^r$ is a commitment to hide x. We denote the set of integers $(1,\ldots,n)$ as $[n]$.

2.2. Pedersen Commitment Scheme

The idea of the Pedersen commitment with perfectly hiding and computationally binding properties is presented for the first time in [22, 23]. The security of the scheme is based on the hardness of the DLP.

Overall, the scheme has three phases. In the setup phase, the receiver picks uniformly random primes q and p with $q|(p-1)$. Suppose G is the cyclic subgroup of $\mathbb{Z}_p^*$ of order q, and $G =< g >$. The receiver picks an element f in G randomly where $\log_g f$ is unknown. In the committing phase, for a randomly chosen $t \in_R \mathbb{Z}_q^*$ to commit a secret $x \in \mathbb{Z}_q^*$, the committer computes $Com(x,t) = g^x f^t$. In the opening phase, the committer reveals x and the corresponding t for the opener to compute $Com(x,t) = g^x f^t$ to check its correctness. A Pedersen commitment scheme should satisfy the perfectly hiding and binding properties.

2.3. Polynomial Pedersen Commitment Scheme

Suppose G is the cyclic subgroup of $\mathbb{Z}_p^*$, and $g_1,\ldots,g_n$ are random group elements, and our message space is $\mathbb{Z}_p^n$, so we want to commit a vector $\mathbf{m}$ with n instances. We commit this vector as, $Com(\mathbf{m},r) = f^r \prod_{i=1}^n g_i^{m_i}$. Note that this scheme is homomorphic, perfectly hiding, and computationally binding.

2.4. Zero Knowledge Σ-protocol and OR-composition

Definition 1 (Σ-Protocol)**.** *Let P and V refer to the probabilistic polynomial time machines. For the protocol system pair (P,V), where $\mathscr{R}$ is a binary relation, a Σ-Protocol for the relation $\mathscr{R}$, is of the three-movement form, namely message, challenge, and response.*

Let both P and V have x as a common input. P has a private input w, where $(x,w) \in \mathscr{R}$. A typical Σ-protocol needs to achieve three security parameters. These are (perfect) completeness, special soundness, and special honest-verifier zero-knowledgeness (sHVZK).

Combining the existing protocol for achieving different aims results in compositions. One of these compositions is the OR-composition, which establishes the correctness of one of the given two statements. In Figure 1, we show the workflow of the OR-composition of $\Sigma-$protocol for the Schnorr protocol. It is a 3-movement protocol that is used as a sub-protocol in the context of range proofs. The details of the workflow will be explained in detail in the next section.

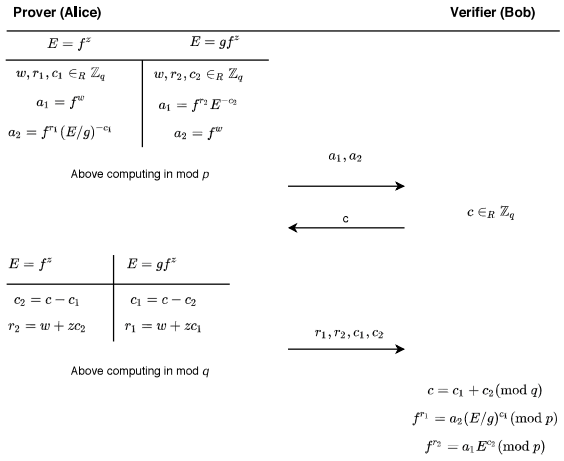


Figure 1: OR-composition of $\Sigma-$protocol [18].

3. Classical Range Proof in Discrete Logarithm Setting

The main idea is to prove a secret integer x belongs to some interval of the form $[0, 2^{k+1}-1]$. For this aim, we decompose the secret in base-2, then prove that this decomposition indeed occurs by 0's and 1's.

The length of the secret x equals to $\lfloor \log_2 x \rfloor + 1$ in bitwise representation, and since $x \in [0, 2^{k+1}-1]$, x can be written as:

$$x = x_0 2^0 + x_1 2^1 + \cdots + x_k 2^k \tag{1}$$

for $x_i \in \{0,1\}$ and $i = 0,1,\ldots,k$. The prover randomly chooses $t_0, t_1, \ldots, t_k \in_R \mathbb{Z}_q$, and computes t as:

$$t = \sum_{i=0}^{k} t_i 2^i \bmod q. \tag{2}$$

Then, she computes the following bit commitment scheme:

$$E_i = E(x_i, t_i) = g^{x_i} f^{t_i} \bmod p \text{ for } i = 0,1,\ldots,k. \tag{3}$$

After that, in each step, the prover proves that the value committed by $E(x_i,t_i)$ is whether 0 or 1. To this end, using a zero-knowledge sub-protocol, namely the OR-composition of Σ-protocol, one can show that she knows whether E_i is in base f or E_i/g is in base f as shown in the Figure 1.

Finally, after the verifier receives E_i and t values, he requires to check (14) using the homomorphic property.

$$g^x f^t \stackrel{?}{=} \prod_{i=0}^{k} E_i^{2^i} \bmod p. \tag{4}$$

In each iteration, both the prover and the verifier compute four exponentiation and seven integers, each length $k^{'}$ for communication. In the end, the cost of the exponentiations equals $4k$. Similarly, the cost of communication equals $7k^{'}k$.

3.1. Classical Range Proof with Base-3 OR-Construction

In this section, we give the scheme of the secret integer x belonging to some interval of the form $[0, 3^{\tilde{k}+1} - 1]$. To do so, this time, we decompose the secret in ternary and prove that decomposition truly occurs in 0's, 1's, and 2's. The length of the secret x equals to $\lfloor \log_3 x \rfloor + 1$ in ternary representation and $x \in [0, 3^{\tilde{k}+1} - 1]$. Hence, we denote the secret as follows:

$$x = x_0 3^0 + x_1 3^1 + \cdots + x_{\tilde{k}} 3^{\tilde{k}} \text{ for } x_i \in \{0,1,2\} \tag{5}$$

and $i = 0,1,\ldots,\tilde{k}$. The prover randomly chooses $t_0, t_1, \ldots, t_{\tilde{k}} \in_R \mathbb{Z}_q$, and computes t as:

$$t = \sum_{i=0}^{\tilde{k}} t_i 3^i \bmod q. \tag{6}$$

Later, computes the commitments as:

$$E_i = E(x_i, t_i) = g^{x_i} f^{t_i} \bmod p \text{ for } i = 0,1,\ldots,\tilde{k}. \tag{7}$$

Then, we need to use the OR-proof. Since the base-2 OR-proof is not usable for our case, we provide the base-3 OR-proof to prove the committed value is equal E_i, or E_i/g, or E_i/g^2 as seen in Figure 2.

After getting E_i and t values, the verifier needs to check the equality of the following property:

$$g^x f^t \stackrel{?}{=} \prod_{i=0}^{\tilde{k}} E_i^{3^i} \bmod p. \tag{8}$$

In each step, six exponentiation need to be computed by both the prover and the verifier. There are 10 integers to exchange, which costs $10k^{'}$. At the end of the protocol, the overall cost for exponentiation is $6\tilde{k}$, and the cost of exchanging integers is $10k^{'}\tilde{k}$. To show the characteristics of the proof, for $|p| = 1024$, $|q| = 1024$, and the security parameter $t = 90$, the completeness holds because the proof always succeeds if the secret $x \in [0,b]$, where b is of the form $3^{\tilde{k}+1} - 1$. For the soundness property, a cheating prover can succeed with a probability less than $1 - (1 - 3^{89})^b$. Finally, the proof is perfectly zero-knowledge in the random-oracle model described in [20].

3.2. Generalizations to Base-u and Comparisons

In this section, we investigate the performance of the base-u decomposition of the secret x. Recall that, for the complexity of the base-3 construction, both the prover and the verifier need to compute six exponentiations in each step. In total, we need $6\tilde{k}$ exponentiations. We also consider the number of exchanged integers, each of the length $k^{'}$, between the prover and the verifier. In this case, there exist 10 integers to exchange in each iteration. The cost of the exchange then is equal to $10\tilde{k}k^{'}$ for the base-3 case. Apparently, the base-3 scheme succeeds in our case.

Throughout this paper, we use $\mathscr{E}$, $\mathscr{I}$, and $\mathscr{M}$ to denote the cost of the operations of exponentiation, inversion, and multiplication in $\mathbb{Z}_p^*$, respectively. Before generalizing this scheme to the base-u, it is enough to compute g^{-1} only once. Since we work in $\mathbb{Z}_p^*$, the cost of the inversion can be considered as $\mathscr{I} \approx \mathscr{E}$ [24]. We may also assume that $\mathscr{E} > 1000\mathscr{M}$ using the square and multiply algorithm for cryptographic applications [24]. Therefore other operations can be considered negligible. Hence, we consider the exponentiations and inverses for comparison. Generalizing in base-u, both the prover and the verifier need to compute $2u$ exponentiations in each iteration. Additionally, $3u+1$ numbers are exchanged in each step of base-u. The required exponentiations are given in Table 1.

The above computations are valid only for one iteration of OR-proof. Since we call the OR-proof as many times as the secret length, these exponentiations and the

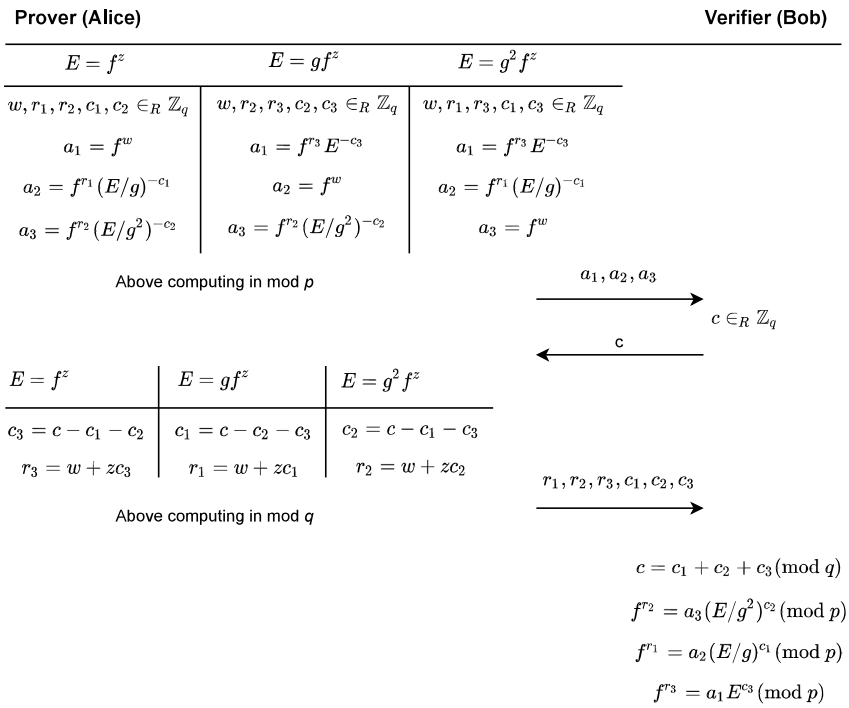


Figure 2: Base-3 OR-proof [18].

Table 1: Comparisons of the required operations for OR-proof in each step [18].

Cost type \ Basis	Base-2	Base-3	Base-4	Base-5	Base-6	...	Base-u
operations	$4\mathscr{E}+\mathscr{I}$	$6\mathscr{E}+\mathscr{I}$	$8\mathscr{E}+\mathscr{I}$	$10\mathscr{E}+\mathscr{I}$	$12\mathscr{E}+\mathscr{I}$	...	$2u\mathscr{E}+\mathscr{I}$
numbers to exchange	7	10	13	16	19	...	$3u+1$

number exchanges repeat as many times as the secret length in base u. Let $k = \lfloor \log_2 x \rfloor$ and $\tilde{k} = \lfloor \log_3 x \rfloor$. Since $\log 2 = 0.30102$ and $\log 3 = 0.47771$, in the base-3 representation $\tilde{k} = \frac{\log 2}{\log 3} k \approx 0.63k$. So, we can compare these two worst-case complexities of the operations executed by the prover as follows:

$$\frac{0.63k(7\mathscr{E})}{k(5\mathscr{E})} = \frac{4.41\mathscr{E}}{5\mathscr{E}} \approx 0.88. \tag{9}$$

Approximately 12% efficiency is achieved in the proof generation when we use the ternary representation instead of the binary representation.

We also analyze other basis complexities in the same way and generalize this analysis for base-u. In general, for the base-u representation, the number of required exponentiations can be formalized as follows:

$$\frac{\log 2}{\log u}(2u)k. \tag{10}$$

After achieving improvements on base-3, base-4, and base-5 representations, the computation cost increases starting with base-6. We also observe that, with respect to computation cost, the base-3 approach has the most efficient cost.

Similar computations can be done for total communication cost. While in the binary representation $7kk'$ bits are exchanged, In the ternary representation $10\tilde{k}k'$ bits are exchanged, and as we mentioned before:

$$\frac{0.63k(10k')}{k(7k')} = \frac{6.309\mathscr{E}}{7\mathscr{E}} \approx 0.90. \tag{11}$$

Therefore, approximately 10% efficiency is executed in the ternary approach compared with the binary approach. It can be formalized for a general base-u as $\frac{\log 2}{\log u}(3u+1)$, and the most efficient computations reached in base-3. In Table 2, the total required operations are tabulated.

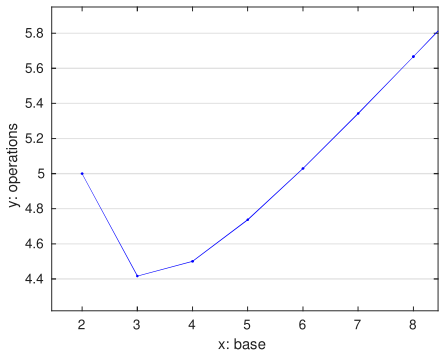
We also sketch the total required exponentiations in

Table 2: Table that compares the total computation cost [18].

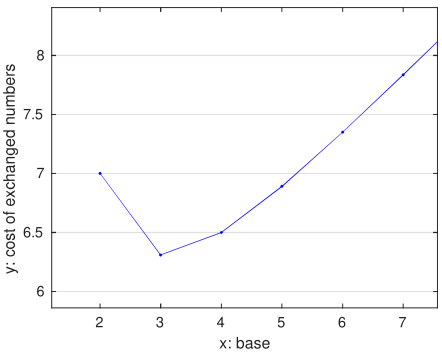
Cost type \ Basis	Base-2	Base-3	Base-4	Base-5	Base-6	...	Base-u
operations	$5k$	$4.41k$	$4.5k$	$4.7k$	$5.02k$	...	$\frac{\log 2}{\log u}(2u)k$
numbers to exchange	$7k'$	$6.309k'$	$6.5k'$	$6.88k'$	$7.22k'$	...	$\frac{\log 2}{\log u}(3u+1)\,k'$

the first of the following graphs of Figure 3a. The graph shows the maximum efficiency working in base-3, i.e., it has the minimum value in $(3, 4.41)$, which is our most efficient point.

The second graph 3b, on the other hand, illustrates the number of bits used in communication. Although in folklore bit-representation it equals 7, in base-3, base-4 and base-5, the scheme has better results. Still, the minimum value is achieved in $(3, 6.31)$, which is our most efficient point. As a result of both comparisons, the construction achieves the most efficiency while using the ternary approach.



(a) Computation cost



(b) Communication cost

Figure 3: Comparison of different base approaches [18].

Note that, instead of Pedersen commitment, this scheme can be constructed using any homomorphic encryption. For example, ElGamal encryption can be used due to its homomorphic property. Though changing the base gives the exact same communication efficiency as using the Pedersen commitment case, for the computation cost, the most efficient base is base-3 with 5.5% efficiency compared with the base-2 case.

4. Classical Range Proof in Elliptic Curve Setting

We can also represent Mao's range proof in the elliptic curve setting. This time we have point multiplications and additions. Suppose we have two points G and F on the elliptic curve. Where G is a publically pre-agreed point on the curve, F is a point derived from G. The process is quite similar to the discrete logarithm case. First, the prover represents the secret in its bits. For randomly chosen $t_0, t_1, \ldots, t_k \in_R \mathbb{Z}_n$, values, the prover computes $t = \sum_{i=0}^{k} t_i 2^i$. After that, for each bit of the secret, the prover computes the elliptic curve Pedersen Commitment as:

$$E_i = E(x_i, t_i) = x_i G + t_i F \text{ for } i = 0, 1, \ldots, k. \qquad (12)$$

For each committed bit, the elliptic curve OR-proof as seen in Figure 4 is called.

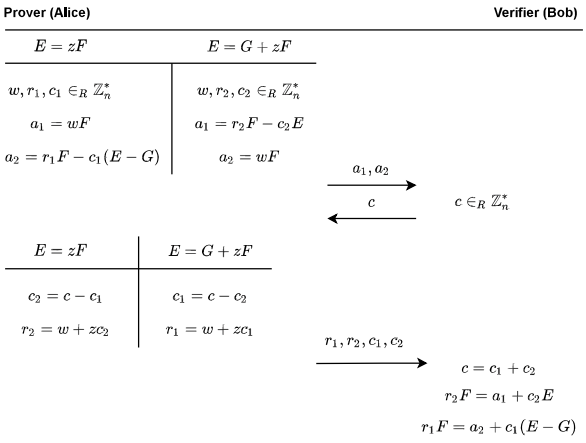


Figure 4: Elliptic Curve Base-2 OR-proof.

At the end of the protocol verifier needs to check the equality of the following property:

$$xG + tF \stackrel{?}{=} \sum_{i=0}^{k} 2^i E_i. \tag{13}$$

Note that both the prover and the verifier need to compute four elliptic curve multiplications and two additions in each OR-proof. Supposing $k = \lfloor \log_2 x \rfloor$ holds, at the end of the range proof, $4k$ point multiplication and $2k$ point additions would be computed. Integers exchanged between the prover, and the verifier will be the same as the discrete logarithm setting.

4.1. Elliptic Curve Classical Range Proof with Base-3 OR-Construction

Instead of using the classical bit-wise approach, suppose we use ternary representation. This time we check if each term is equal to 0, 1, or 2. To do so, after representing the secret in its trits, the prover computes $t = \sum_{i=0}^{k} t_i 3^i$ for randomly chosen $t_0, t_1, \ldots, t_{\tilde{k}} \in_R \mathbb{Z}_n$. For each trit of the secret, a Pedersen Commitment is computed as equation (12). Later the prover calls a modified OR-proof, as seen in Figure 5, for each committed secret trits.

$$xG + tF \stackrel{?}{=} \sum_{i=0}^{\tilde{k}} 3^i E_i. \tag{14}$$

4.2. Generalizations to Base-u and Comparisons

Consider that in this case, both the prover and the verifier need to compute six elliptic curve multiplications, four additions, and one doubling operation in each OR-proof. The required multiplications in each step are given in Table 3.

We call OR-proof till the term size of the decomposed secret. Regarding $\tilde{k} = \lfloor \log_3 x \rfloor$, in total $6\tilde{k}$ point multiplications and $4\tilde{k}$ point additions are computed. We may assume that $M \approx 8A$ using the double and add algorithm in elliptic curve cryptography. That is why we will only take account of multiplications to compare overall efficiency. We can compare these two worst-case complexities of the operations executed by the prover as follows:

$$\frac{0.63k(6\mathscr{M})}{k(4\mathscr{M})} = \frac{3.78\mathscr{M}}{4\mathscr{M}} \approx 0.945. \tag{15}$$

The required multiplications in each step are given in Table 3. This corresponds to approximately 5.5% efficiency in both proof generation and verification when we use the base-3 instead of the base-2 in the elliptic curve. For the characteristics of the proof, completeness trivially holds with probability 1, i.e., if x is in the given range, then the verifier always accepts. For the soundness property, the length of the challenge c is such that the number of possible c-values is super-polynomial in bit size k. Assuming any other prover, say P^* since $c \neq c'$ holds, a witness set holding our relation can be computed in polynomial time. This prof is also perfectly zero-knowledge in the random-oracle model described in [20].

In the first graph of Figure 6, we sketch the total required multiplications on different basis. Maximum efficiency can be observed when the base is selected as three. In the first graph 3a, one may find the number of elliptic curve point multiplications for the different basis. The graph has its minimum value in $(3, 3.78)$, which is our most efficient point. The second graph 3b is the same as the discrete logarithm setting, and we can see the number of bits exchanged. The construction achieves the most efficiency when we use the base-3 representation.

Similar to DLP-setting, instead of an elliptic curve Pedersen commitment, this scheme can be constructed using any homomorphic encryption. For example, elliptic-curve ElGamal encryption can be used due to its homomorphic property. Though changing the base gives the exact same communication efficiency as using the Pedersen commitment case, for the computation cost, the most efficient base is base-3 with 5.5% efficiency compared with the base-2 case.

5. A Batched Range Proof for Low Degree Polynomial Relations

Bootle et al.[21] proposed a zero-knowledge argument allowing to prove many instances of the same relation to be proved and verified in a single argument. Their protocol is based on discrete logarithm assumption in groups with prime order p. It can easily instantiate as polynomial evaluation proofs, membership proofs, and range proofs. They also give the most efficient parameters to use the protocol. In the scope of this study, we include their structure as range proofs to exemplify range proofs using polynomial commitments. In this protocol, t denotes the number of batched instances. We will consider the $t = 1$ case, where we have a single relation.

The secret a, represented in its bits as $a_0, a_1, \ldots, a_{m-1}$. We know that for each bit $a_i(1 - a_i) = 0$ should be satisfied. So that we define the vectors $\mathbf{a}$ and $\mathbf{b}$ as:

$$\mathbf{a} = (a_0, a_1, \ldots, a_{m-1}), \quad \mathbf{b} = (2^0, 2^1, \ldots, 2^{m-1}). \tag{16}$$

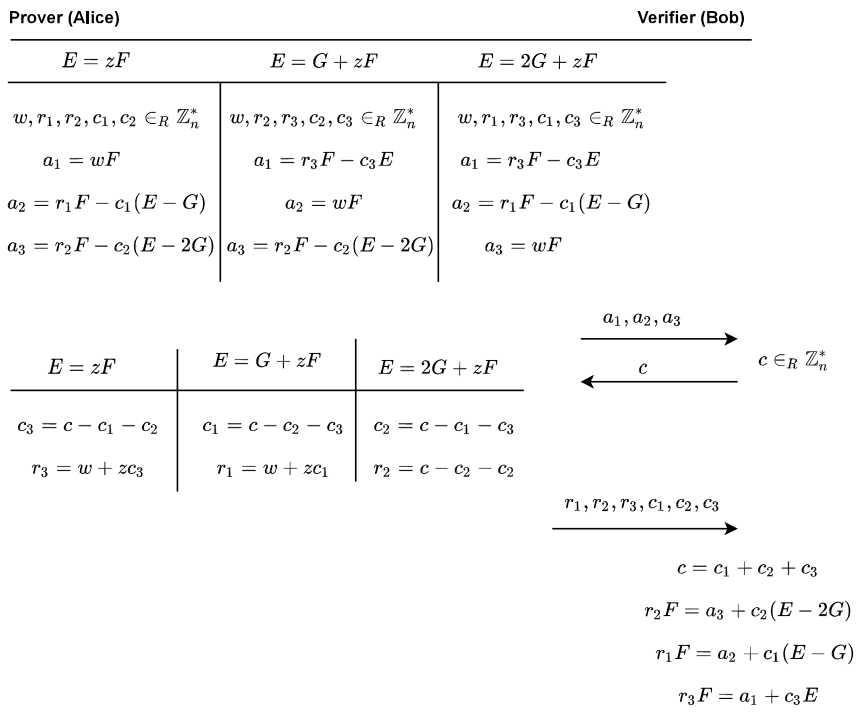


Figure 5: Elliptic Curve Base-3 OR-proof.

Table 3: Comparisons of the required operations for OR-proof in each step

Cost type \ Basis	Base-2	Base-3	Base-4	...	Base-u
operations	$4\mathscr{M}+2\mathscr{A}$	$6\mathscr{M}+4\mathscr{A}$	$8\mathscr{M}+4\mathscr{A}$	...	$2u\mathscr{M}+u\mathscr{A}$
numbers to exchange	7	10	13	...	$3u+1$

Table 4: Table that compares the required operations for range proof in total

Cost type \ Basis	Base-2	Base-3	Base-4	...	Base-u
operations	$4k$	$3.78k$	$4k$	...	$\frac{\log 2}{\log u}(2u)\ k^{'}$
numbers to exchange	$7k^{'}$	$6.309k^{'}$	$6.5k^{'}$	...	$\frac{\log 2}{\log u}(3u+1)\ k^{'}$

The polynomials we use in the protocols are defined as:

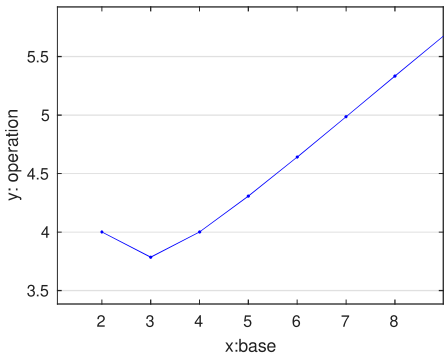
$$\mathbf{P}(\mathbf{a},\mathbf{b}) = \mathbf{a} \circ (\mathbf{1} - \mathbf{a}), \tag{17}$$

$$\mathbf{Q}(\mathbf{a},\mathbf{b}) = \sum_{i=0}^{m-1} a_i 2^i, \tag{18}$$

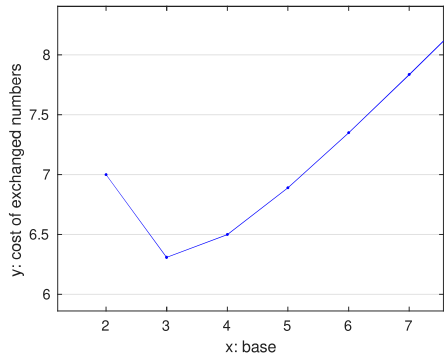
where we use the Hadamard product to show $a_i(1 - a_i) = 0$ for each bit. Note that we define l_a as the length of the secret $\mathbf{a}$, and $\mathbf{P}$, and $\mathbf{Q}$ are length l_P, l_Q vectors of polynomials of degree d_P, d_Q respectively.

Common reference strings are $crs = (ck, z_1, \ldots, z_m)$, where commitment key $ck \leftarrow \text{Gen}(1^\lambda)$, and $z_1, \ldots, z_m$ are points defining Lagrange polynomials $l_1(X), \ldots, l_m(X)$. We define $l_0(X) = \prod_{j=1}^{m}(X - z_j)$. The workflow of the general protocol can be summarized as follows:

$\mathbf{P} \to \mathbf{V}$ The prover picks $r_0, s_0, \ldots, r_m \leftarrow \mathbb{Z}_p$ and $\mathbf{a}_{0,1}, \ldots, \mathbf{a}_{0,n} \leftarrow \mathbb{Z}_p^{l_a}$ and $c_0, \ldots, c_n \leftarrow \mathbb{Z}_p^{l_Q}$.
Then computes: $C_0 = \text{Com}_{ck}(\mathbf{c}_1, \ldots, \mathbf{c}_n; r_0)$ and $A_i = \text{Com}_{ck}(\mathbf{a}_{0,1}, \ldots, \mathbf{a}_{0,n}; s_i)$ for $i \in 0 \cup [m]$
Defines:



(a) Required exponentiation



(b) Number of exchanged integers

Figure 6: Comparison of different base approaches in the elliptic curve setting.

$$\bar{\mathbf{a}}_j(X) = \sum_{i=0}^{m} \mathbf{a}_{i,j} l_i(X) \quad \bar{\mathbf{b}}_j(X) = \sum_{i=1}^{m} \mathbf{b}_{i,j} l_i(X), \tag{19}$$

$$\mathbf{P}^*_j(X) = \frac{\mathbf{P}\left(\bar{\mathbf{a}}_j(X), \bar{\mathbf{b}}_j(X)\right)}{l_0(X)}, \tag{20}$$

$$\mathbf{Q}^*_j(X) = \mathbf{c}_j + \frac{\sum_{i=1}^{m} \mathbf{Q}\left(\mathbf{a}_{i,j}, \mathbf{b}_{i,j}\right) l_i(X) - \mathbf{Q}\left(\bar{\mathbf{a}}_j(X), \bar{\mathbf{b}}_j(X)\right)}{l_0(X)} \tag{21}$$

As a sub-protocol polynomial commitment PolyCommit $(ck, \mathbf{P}^*_j(X)_{j\in[n]})$ is called. In this sub-protocol, for randomly chosen $b_1, \ldots, b_n \in \mathbb{Z}_p^{l_P}$, the prover arranges a matrix as described in [14], with the entries of $\mathbf{P}^*_j(X)$ as:

$$\begin{pmatrix} \mathbf{p}_0 & \mathbf{b}_1 & \cdots & \mathbf{b}_{n-1} & \mathbf{b}_n \\ \mathbf{p}_1 & \mathbf{p}_{d+1} & \cdots & \mathbf{p}_{(n-2)m+d+1} & \mathbf{p}_{(n-1)m+d+1} \\ & & & & \vdots \\ \mathbf{p}_d - \mathbf{b}_1 & \vdots & \ddots & & \mathbf{b}_n \\ 0 & & & & \mathbf{p}_{nm+1} \\ \vdots & & & & \vdots \\ 0 & \mathbf{p}_{m+d-1} & \cdots & \mathbf{p}_{(n-2)m+d-1} & \mathbf{p}_{N-1} \\ 0 & \mathbf{p}_{m+d} - \mathbf{b}_2 & \cdots & \mathbf{p}_{(n-2)m+d} - \mathbf{b}_n & \mathbf{p}_N \end{pmatrix} \tag{22}$$

For each row of the matrix, the prover randomly selects $r_i \in \mathbb{Z}_p$, and computes the commitment corresponding row. Output of this sub-protocol is $msg_{(P,1)} = (\{H_i\}_{i=0}^m)$ and a statement $st = (\mathbf{P}^*_j(X), \{b_j\}_{j=1}^n, \{r_i\}_{i=0}^m)$.
Similarly, we call polynomial commitment for $\mathbf{Q}^*_j(X)$, as PolyCommit$(ck, \mathbf{Q}^*_j(X)_{j\in[n]})$. This protocol outputs $(msg_{(Q,1)}, st_Q)$.

The prover sends the $\{A_i\}_{i\in[m]}$, $msg_{(Q,1)}$, and $msg_{(Q,1)}$ to the verifier.

$\mathbf{P} \leftarrow \mathbf{V}$ The verifier chooses a challenge $x \leftarrow \mathbb{Z}_p \setminus \{z_1, \ldots, z_m\}$, and sends it to the prover.

$\mathbf{P} \rightarrow \mathbf{V}$ After getting the challenge x, the prover evaluates the polynomials in the point x using the statements st_P and st_Q. To this end, the prover calls PolyEval(st_P, x). In this evaluation step the prover computes:

$$\bar{\mathbf{p}}_j = \sum_{i=0}^{m} \mathbf{p}_{i,j} x^i, \tag{23}$$

together with $\tilde{r} = \sum_{i=0}^{m} r_i x^i$. Output of this step is $msg_{P,2} = (\{\bar{\mathbf{p}}_j\}, \tilde{r})$. Similarly, the prover calls the evaluation step for the polynomial Q as PolyEval(st_Q, x), and get the $msg_{Q,2}$.

The prover computes:

$$\bar{\mathbf{a}}_j = \bar{\mathbf{a}}_j(x), \quad \bar{r} = \sum_{i=0}^{m} r_i l_i(x), \quad \bar{s} = \sum_{i=0}^{m} s_i l_i(x). \tag{24}$$

Then sends $\{\bar{a}_j\}_{j\in[n]}$, $\bar{r}$, $\bar{s}$, $msg_{P,2}$, $msg_{Q,2}$ to the verifier.

The verifier runs the PolyVerify$(ck, msg_{P,1}, msg_{P,2}, x)$. In this step, the verifier checks :

$$Com(\bar{p}_0, \ldots, \bar{p}_n; \bar{r}) = \prod_{i=0}^{m} H_i^{x^i}. \tag{25}$$

This step outputs $\bar{p} = (\bar{p}_1, \ldots, \bar{p}_n)$. Similarly he runs the PolyVerify $(ck, msg_{Q,1}, msg_{Q,2}, x)$, which outputs $\bar{q} =$

$(\bar{q}_1,\ldots,\bar{q}_n)$. Then he checks:

$$Com(\bar{a}_1,\ldots,\bar{a}_n;\bar{s}) = \prod_{i=0}^{m} A_i^{l_i(x)}. \tag{26}$$

After computing $\bar{b}_j = \bar{b}_j(x)$, he checks:

$$P(\bar{a}_j,\bar{b}_j) = \bar{p}_j l_0(x). \tag{27}$$

Lastly checks:

$$Com_{ck}(\{\bar{q}_j l_0(x) + Q(\bar{a}_j,\bar{b}_j)\}_{j\in[n]};\bar{r}) = \prod_{i=0}^{m} C_i^{l_i(x)}. \tag{28}$$

To make the complexity analysis, we need to consider the polynomial multi-exponentiations in the prover's part. With the help of the techniques presented in [25], which we will call Pippenger's algorithm from the rest of the paper, n multi-exponentiation costs approximately $\frac{n}{\log n}$. Note that we have such multi-exponentiations while we were computing A_i, C_0, $msg_{P,1}$, and $msg_{Q,1}$.

First, we want to get the cost of computing $A_i = \text{Com}_{ck}\ (\mathbf{a}_{0,1},\ldots,\mathbf{a}_{0,n};s_i)$ for $i \in 0 \cup [m]$. For each A_i we have n instances; each has length l_a. So we have $l_a n$ multi-exponentiation, which will cost $\frac{l_a n}{\log l_a n}$ by using Pippenger's algorithm. Since we call these multi exponentiations $m+1$ times, in total the operation will cost $(m+1)\frac{l_a n}{\log l_a n}$. Considering $t = mn$, we will take this cost roughly $\frac{l_a t}{\log l_a n}$.

Second, while computing C_0 =$\text{Com}_{ck}(\mathbf{c}_1,\ldots,\mathbf{c}_n;r_0)$, we have n instances each of them have length l_Q. Therefore, we have $l_Q n$ multi-exponentiation, which will cost $\frac{l_Q n}{\log l_Q n}$.

Third, while generating $msg_{(P,1)} = (\{H_i\}_{i=0}^m)$ we have row-by-row polynomial commitments on the matrix generating with the entries of $\mathbf{P}_j^*(X)$. We have $n+1$ instances in each row, each length l_P, and the degree d_P. Therefore, we compute $l_p d_P(n+1)$ exponentiations for each of the $m+1$ rows. In total generating the $msg_{(P,1)}$ will cost roughly $\frac{l_P d_P t}{\log l_P d_P n}$.

Lastly, very similar to $msg_{(P,1)}$, the cost of generating the $msg_{(Q,1)}$ will cost roughly $\frac{l_Q d_Q t}{\log l_Q d_Q n}$.

We find that the prover's computational costs are dominated by the following:

$$O\left(\frac{l_a t}{\log l_a n} + \frac{l_Q n}{\log l_Q n} + \frac{l_P d_P t}{\log l_P d_P n} + \frac{l_Q d_Q t}{\log l_Q d_Q n}\right). \tag{29}$$

Note that work on $t = 1$. For base-2 we set the lengths $l_a = l_P = l$, and $l_Q = 1$, the degrees $d_P = 2$, $d_Q = 1$, so that the bit-wise computation costs are dominated by:

$$\frac{l}{\log l} + 2 + \frac{2l}{\log l} + 1. \tag{30}$$

In the communication complexity, when $t = 1$, we have a single relation. The prover sends $\{A_i\}_{i\in[m]}$, $msg_{(P,1)}$, and $msg_{(Q,1)}$ to the verifier, which corresponds to 6 group elements. The verifier, in return, sends the challenge x point, which costs 1 group element. As a response, the prover sends $\{\bar{a}_j\}_{j\in[n]}$, $\bar{r}$, $\bar{s}$, $msg_{P,2}$, $msg_{Q,2}$, costing $l_a + l_P d_P + l_Q d_Q + 4$ field elements to the verifier.

In total, when $t = 1$, 7 group elements and $l_a + l_P d_P + l_Q d_Q + 4$ field elements are used for communication. While working on base-2, 7 group elements and $l + 2l + 5$ field elements are used for communication.

5.1. Generalizations to Base-u and Comparisons

Consider that if we generalize this protocol to u-ary representation, we first represent the secret a as $a_0, a_1, \ldots, a_{\tilde{m}-1}$.

We set $l_a, l_b, l_P = \tilde{m}$, $d_P = u$, $l_Q = d_Q = 1$. We have vectors:

$$\mathbf{a} = (a_0, a_1, \ldots, a_{\tilde{m}-1}), \quad \mathbf{b} = (u^0, u^1, \ldots, u^{\tilde{m}-1}). \tag{31}$$

The polynomial we use in the protocols are defined as follows:

$$\mathbf{P}(\mathbf{a},\mathbf{b}) = \mathbf{a} \circ (\mathbf{1}-\mathbf{a}) \circ \cdots \circ (\mathbf{a}-u+1), \tag{32}$$

$$\mathbf{Q}(\mathbf{a},\mathbf{b}) = \sum_{i=0}^{\tilde{m}-1} a_i u^i. \tag{33}$$

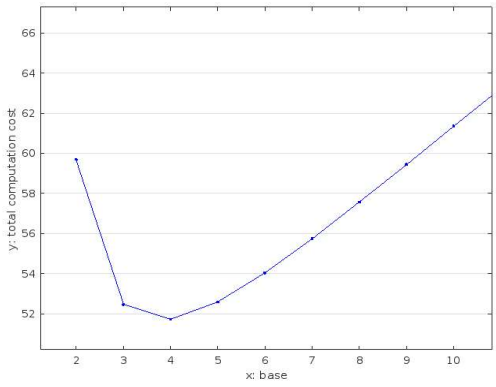
In this case, the ternary-base computation cost will be approximately:

$$\frac{0.63l}{\log 0,63l} + 2 + \frac{3(0.63l)}{\log(0.63l)} + 1. \tag{34}$$

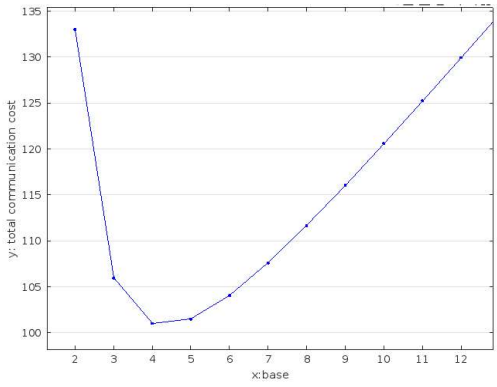
When we change it to other bases, though the length of the polynomials will decrease, the degree of them will increase. To compare the different bases, we set some values to length l; if we take the l as 32-bit, the change in the computation cost is sketched in Figure 7. The graph 7a has its minimum value when we work on base-4, which corresponds to approximately 13.4% efficiency compared with the base-2 approach.

When it comes to communication, under the condition that $t = 1$, remember we have a single relation so that 7 group elements and $l_a + l_P d_P + l_Q d_Q + 4$ field elements are used. When we change it to other bases, though the length of the polynomials will decrease, the

degree of them will increase. As an example, though 7 group elements are the same for all bases, for bit-wise representation cost will be $l+2l+5$ field elements, for the ternary base representation $0.63l+3(0,63l)+5$ field elements. We sketch the total communication cost for field elements in the following graph of Figure 7b. The graph has its minimum value in 4, as stated in the original paper [21], while $t=1$, the construction achieves the most efficiency when using the base-4 representation. Compared with the base-2 scheme, the base-4 scheme has approximately 24% improvement.



(a) Computation cost



(b) Communication cost

Figure 7: Comparison of different base approaches in the finite field setting.

If we instantiate this protocol using an elliptic curve group, instead of multi-exponentiation we have elliptic curve multiplications while we were computing A_i, C_0, $msg_{P,1}$ and $msg_{Q,1}$. Using the simultaneous multiple-point multiplication technique, see Algorithm 3.48 in [26], also known as Shamir's trick, these multiplications would be computed more efficiently. Shamir's trick states that where k and l are t-bit numbers, P and Q are two elliptic curve points, given width w, computing $kP+lQ$ has an expected running time approximately:

$$\Big(\frac{2^{2(w-1)}}{2^{2w}}d-1\Big) \text{ additions } +(d-1)w \text{ doublings.} \quad (35)$$

Also, to compute the kP one can use the fixed-base windowing method for point multiplication, see Algorithm 3.41 in [26]. This operation has an expected running time of approximately:

$$(2^w+d-3) \quad (36)$$

additions. Nevertheless, we work $t=mn=1$, which means we have a single instance. In our case, while computing A_i, C_0 we do not need such multi-multiplications so we will not consider these efficiency methods in our study. Although we do not use the corresponding efficiency methods described above, we note that they can be used if deemed necessary.

First, to get the cost of computing A_i $=\text{Com}_{ck}(\mathbf{a}_{0,1},\ldots,\mathbf{a}_{0,n};s_i)$ for $i\in 0\cup[m]$. For each A_i we have n instances, each of them has length l_a. So we have l_an multi-multiplications. Since we call these multi exponentiations $m+1$ times, in total we have $(m+1)(l_an+1)$ multiplications.

Second, while computing C_0 $=\text{Com}_{ck}(\mathbf{c}_1,\ldots,\mathbf{c}_n;r_0)$, we have n instances each of them have length l_Q. Therefore, we have l_Qn elliptic curve multiplications.

Third, while generating $msg_{(P,1)}=(\{H_i\}_{i=0}^m)$ we have row-by-row polynomial commitments on the matrix generating with the entries of $\mathbf{P}_j^*(X)$. In each row, we have $n+1$ instances, each of the length l_P and the degree d_P. Therefore, we compute $l_pd_P(n+1)$ multiplications for each of the $m+1$ rows. So that in total to generate $msg_{(P,1)}$ we compute $(m+1)(l_Pd_Pn+1)$ multiplications.

Lastly, very similar to $msg_{(P,1)}$, the cost of the generating $msg_{(Q,1)}$ will cost roughly $(m+1)(l_Qd_Qn+1)$ multiplications.

Therefore, the computation cost is roughly:

$$(m+1)(l_an+1)+l_Qn+(m+1)(l_Pd_Pn+1) \quad (37)$$
$$+(m+1)(l_Qd_Qn+1) \quad (38)$$

multiplications. While $t=1$, in binary representation it costs $2(l_a+1)+2+2(l_Pd_P+1)+2(l_Qd_Q+1)$ multiplications. If we put the corresponding lengths for our case, the cost is:

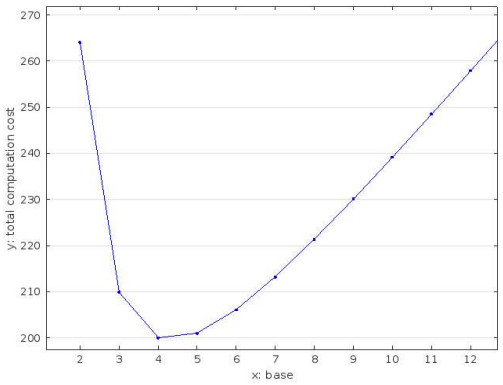
$$2(l+1)+2+2(2l+1)+4=(6l+10) \quad (39)$$

multiplications.

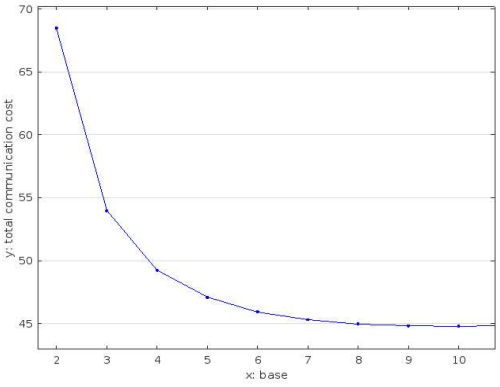
If we turn this protocol to base-3 total cost is $2(0.63l+1)+2+2(3(0,63l)+1)+4)=(5.04l+10)$

multiplications. This corresponds to approximately 20.4% efficiency in proof generation. Similarly for base-4, the total cost is $2(0.5l+1)+2+2(4(0,5l)+1)+4)=(5l+10)$ multiplications. The generalization of this protocol to base-u is sketched in Figure 8b. The graph 8a has its minimum value when we work on base-4 with approximately 24.2% efficiency in proof generation.

It is stated in the original paper that after some appropriate settings, the total communication cost will be minimized by roughly $\sqrt{l_P d_P}+\sqrt{l_Q d_Q}+5$ group elements, and $l_a+\sqrt{l_P d_P}+\sqrt{l_Q d_Q}+4$ field elements. We refer to the original paper [21], for the details of the corresponding settings. Change for the different bases is sketched in the graph 8b. We show that this protocol has always decreased communication costs in the elliptic curve case.



(a) Computation cost



(b) Number of exchanged integers

Figure 8: Comparison of different base approaches in the elliptic curve setting.

As a proof size, we may compare these novel alternatives, while l_r is the bit length of the range, and t is the number of instances to proved. Consider $t=1$ for the case where only a single instance is proved. Note that Bulletproof transmits $2(\log_2(l_r)+\log_2(t))+4$ group elements and 5 field elements. Bulletproof+ on the other hand, transmits $2(\log_2(l_r)+\log_2(t))+3$ group elements and 3 fields elements. In Table 5, transmitted group and field elements are tabulated.

Table 5: Table that compares the proof size

Cost type \ Basis	Group elements	Field elements
Base-u Classical	-	$t\frac{\log 2}{\log u}(3u+1)\,k'$
EC Base-u Classical	-	$t\frac{\log 2}{\log u}(3u+1)\,k'$
Base-u batched	7	$l_a+l_P d_P+l_Q d_Q+4$
EC Base-u batched	$4\sqrt{l_P d_P}+\sqrt{l_Q d_Q}+5$	$l_a+\sqrt{l_P d_P}+\sqrt{l_Q d_Q}+4$
Bulletproof	$2(\log_2(l_r)+\log_2(t))+4$	5
Bulletproof+	$2(\log_2(l_r)+\log_2(t))+3$	3

6. Conclusion

We have presented a modified OR-composition of the Shnorr protocol to base-u in both the discrete logarithm setting and the elliptic curve setting. We computed the overall complexity in the context of required exponentiations and the cost of the number of exchanged integers. After our derivations, we showed that the cost has a pattern, so we have generalized and formalized the proof for different bases. At the end of these comparisons, we have observed that the computational complexity of the base-3 representation in the discrete logarithm setting is 12% more efficient than the other base representations. Similarly, base-3 representation is 5.5% more efficient in the elliptic curve setting than bit-wise representation. Additionally, comparing in the context of the communication complexity, we showed that the base-3 approach is 10% more efficient with respect to the other base approaches. We have also observed that if we use El-Gamal encryption instead of Pedersen commitments, the base-3 representation is 5.5% more efficient in both discrete logarithm and elliptic curve-based constructions. Moreover, we have analyzed the range poof proposed by Bootle et al. in both finite fields and elliptic-curve settings. We obtained that when $t=1$, the protocol has the minimum cost for computation in base-4 representation with 13.4% improvement and for communication in base-4 representation with 24% improvement. Similarly, if we set this protocol using elliptic curve cryptography, base-4 representation gives the lowest computation cost with 24.2% improvement.

References

[1] A. H. Chan, Y. Frankel, Y. Tsiounis, Easy come - easy go divisible cash, in: Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998,

Proceeding, Vol. 1403 of LNCS, Springer, 1998, pp. 561–575. doi:10.1007/BFb0054154.

[2] I. Miers, C. Garman, M. Green, A. D. Rubin, Zerocoin: Anonymous distributed e-cash from bitcoin, in: 2013 IEEE Symposium on Security and Privacy, IEEE, 2013, pp. 397–411. doi:10.1109/SP.2013.34.

[3] P. McCorry, S. F. Shahandashti, F. Hao, A smart contract for boardroom voting with maximum voter privacy, in: A. Kiayias (Ed.), Financial Cryptography and Data Security, Vol. 10322 of LNCS, Springer International Publishing, 2017, pp. 357–375. doi:10.1007/978-3-319-70972-7_20.

[4] C. Lin, M. Luo, X. Huang, K.-K. R. Choo, D. He, An efficient privacy-preserving credit score system based on noninteractive zero-knowledge proof, IEEE Systems Journal (2021). doi:10.1109/JSYST.2020.3045076.

[5] A. Hahn, R. Singh, C.-C. Liu, S. Chen, Smart contract-based campus demonstration of decentralized transactive energy auctions, in: 2017 IEEE Power & energy society innovative smart grid technologies conference (ISGT), IEEE, 2017, pp. 1–5. doi:10.1109/ISGT.2017.8086092.

[6] J. Camenisch, M. Michels, Separability and efficiency for generic group signature schemes, in: M. Wiener (Ed.), Advances in Cryptology - CRYPTO' 99, Vol. 1666 of LNCS, Springer Berlin Heidelberg, Berlin, Heidelberg, 1999, pp. 413–430. doi:10.1007/3-540-48405-1_27.

[7] N. van Saberhagen, Cryptonote v 2.0, in: Tech. Rep. [Online], 2013.
URL Available:https://cryptonote.org/whitepaper.pdf

[8] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, M. Virza, Zerocash: Decentralized anonymous payments from bitcoin, in: 2014 IEEE Symposium on Security and Privacy, 2014, pp. 459–474. doi:10.1109/SP.2014.36.

[9] E. F. Brickell, D. Chaum, I. B. Damgrd, J. van de Graaf, Gradual and verifiable release of a secret (extended abstract), in: C. Pomerance (Ed.), Advances in Cryptology - CRYPTO '87, Vol. 293 of LNCS, Springer Berlin Heidelberg, Berlin, Heidelberg, 1988, pp. 156–166. doi:10.1007/3-540-48184-2_11.

[10] I. B. Damgrd, On the existence of bit commitment schemes and zero-knowledge proofs, in: G. Brassard (Ed.), Advances in Cryptology — CRYPTO' 89 Proceedings, Vol. 435 of LNCS, Springer New York, New York, NY, 1990, pp. 17–27. doi:10.1007/0-387-34805-0_3.

[11] E. Fujisaki, T. Okamoto, Statistical zero knowledge protocols to prove modular polynomial relations, in: B. S. Kaliski (Ed.), CRYPTO '97, Vol. 1431 of LNCS, Springer Berlin Heidelberg, Berlin, Heidelberg, 1997, pp. 16–30. doi:10.1007/BFb0052225.

[12] S. Canard, I. Coisel, A. Jambert, J. Traoré, New results for the practical use of range proofs, in: S. Katsikas, I. Agudo (Eds.), Public Key Infrastructures, Services and Applications, Vol. 8341 of LNCS, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014, pp. 47–64. doi:10.1007/978-3-642-53997-8_4.

[13] F. Boudot, Efficient proofs that a committed number lies in an interval, in: B. Preneel (Ed.), Advances in Cryptology - EUROCRYPT 2000, Vol. 1807 of LNCS, Springer Berlin Heidelberg, Berlin, Heidelberg, 2000, pp. 431–444. doi:10.1007/3-540-45539-6_31.

[14] J. Bootle, A. Cerulli, P. Chaidos, J. Groth, C. Petit, Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting, in: M. Fischlin, J.-S. Coron (Eds.), Advances in Cryptology – EUROCRYPT 2016, Springer Berlin Heidelberg, Berlin, Heidelberg, 2016, pp. 327–357.

[15] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, G. Maxwell, Bulletproofs: Short proofs for confidential transactions and more, in: 2018 IEEE symposium on security and privacy (SP), IEEE, 2018, pp. 315–334.

[16] H. Chung, K. Han, C. Ju, M. Kim, J. H. Seo, Bulletproofs+: Shorter proofs for a privacy-enhanced distributed ledger, IEEE Access 10 (2022) 42067–42082.

[17] J. Camenisch, R. Chaabouni, a. shelat, Efficient protocols for set membership and range proofs, in: J. Pieprzyk (Ed.), Advances in Cryptology - ASIACRYPT 2008, Vol. 5350 of LNCS, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 234–252. doi:10.1007/978-3-540-89255-7_15.

[18] E. Günsay, C. B. Onur, M. Cenk, An improved range proof with base-3 construction, in: 2021 14th International Conference on Security of Information and Networks (SIN), Vol. 1, 2021, pp. 1–6. doi:10.1109/SIN54109.2021.9699258.

[19] W. Mao, Guaranteed correct sharing of integer factorization with off-line shareholders, in: Public Key Cryptography, First International Workshop on Practice and Theory in Public Key Cryptography, PKC '98, Pacifico Yokohama, Japan, February 5-6, 1998, Proceedings, Vol. 1431 of LNCS, Springer, 1998, pp. 60–71. doi:10.1007/BFb0054015.

[20] R. Cramer, I. Damgrd, B. Schoenmakers, Proofs of partial knowledge and simplified design of witness hiding protocols, in: Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings, Vol. 839 of LNCS, Springer, 1994, pp. 174–187. doi:10.1007/3-540-48658-5_19.

[21] J. Bootle, J. Groth, Efficient batch zero-knowledge arguments for low degree polynomials, in: M. Abdalla, R. Dahab (Eds.), Public-Key Cryptography – PKC 2018, Springer International Publishing, Cham, 2018, pp. 561–588.

[22] R. Metere, C. Dong, Automated cryptographic analysis of the pedersen commitment scheme, in: J. Rak, J. Bay, I. Kotenko, L. Popyack, V. Skormin, K. Szczypiorski (Eds.), Computer Network Security, Vol. 10446 of LNCS, Springer International Publishing, 2017, pp. 275–287. doi:10.1007/978-3-319-65127-9_22.

[23] S. Micali, M. Rabin, J. Kilian, Zero-knowledge sets, in: 44th Annual IEEE Symposium on Foundations of Computer Science 2003 – FOCS 2003., Proceedings., 2003, pp. 80–91. doi:10.1109/SFCS.2003.1238183.

[24] A. Menezes, J. Katz, P. van Oorschot, S. Vanstone, Handbook of Applied Cryptography, Discrete Mathematics and Its Applications, CRC Press, 1996. doi:10.1201/9780429466335.

[25] N. Pippenger, On the evaluation of powers and monomials, SIAM J. Comput. 9 (1980) 230–250.

[26] D. Hankerson, A. J. Menezes, S. Vanstone, Guide to Elliptic Curve Cryptography, 1st Edition, Springer Publishing Company, Incorporated, 2010.