Speeding Up Multi-Scalar Multiplications for Pairing-Based zkSNARKs

Xinxin Fan[1], Veronika Kuchta[2], Francesco Sica[2], and Lei Xu[3]

[1]IoTeX, Menlo Park, CA 94025
[2]Department of Mathematics and Statistics
Florida Atlantic University, Boca Raton, FL 33431
[3]Department of Computer Science
Kent State University, Kent, OH 44242

Abstract. Multi-scalar multiplication (MSM) is a fundamental component in many zero-knowledge succinct non-interactive argument systems (ZK-SNARKs), and a major performance bottleneck in proof generation for these schemes. One key strategy to accelerate MSM is through precomputation. Several algorithms, such as Pippenger's and BGMW, along with their variations, have been proposed to address this.
In this paper, we revisit the recent precomputation-based MSM method introduced by Luo, Fu, and Gong at CHES 2023 [11] and extend their approach. Specifically, we present a generalized construction for optimal buckets. Given a set of multipliers M, we propose an algorithm that identifies the optimal bucket set B to minimize the computation time.
This new construction yields performance improvements over the original Pippenger's MSM method, as demonstrated through both theoretical analysis and experimental results. Additionally, we correct Property 1 in the MSM method from [11], providing the corresponding experimental validations. To further enhance run-time efficiency and reduce storage requirements, we propose the use of an efficient endomorphism, supported by theoretical and experimental analysis.

1 Introduction

The concept of zero-knowledge proof (ZKP) was introduced by Goldwasser, Micali, and Rackoff in 1985 [6]. Terms like ZKP or zero-knowledge arguments (ZKA) satisfy the three security properties, such as *correctness* - meaning that an honest prover will always convince a verifier of knowing a secret to a public statement, *soundness*—ensuring that a dishonest prover cannot prove a false statement—and *zero-knowledge*—indicating that the proof reveals no extra information beyond the truthfulness of the statement the prover aims to prove. While in ZKP systems the soundness property holds for provers with unbounded (statistical) capabilities, it is assumed that the prover is computationally bounded in ZKA systems.

Author list in alphabetical order; see `https://www.ams.org/about-us/governance/committees/Statement_JointResearchanditsPublication.pdf`

There has been a surge of interest in putting ZKPs and ZKAs into practice in the past few years, which was first triggered by the demands for privacy protection in the blockchain environment (e.g., Zerocash (zcash) [1]), and then more general applications such as verifiable computation.

An advanced version of ZKAs with short proofs and efficient verification is known as zkSNARKs (**z**ero-**k**nowledge **S**uccinct **N**on-interactive **AR**guments of **K**nowledge). They can be seen as a composition of the **N**on-**I**nteractive **Z**ero-**K**nowledge proofs (NIZKs) and succinct Arguments of Knowledge. NIZKs were introduced by Blum, Feldman and Micali [3] while Kilian [9] provided a definition on efficient zero-knowledge arguments. In a proof system, the prover's computational power may be unbounded, but in an argument system, it is assumed that the prover is computationally bounded.

While many zkSNARK schemes have been proposed since then, pairing-based zkSNARK is still one of the most attractive options in practice. While the verification in zkSNARKs is fast, the construction of such argument systems is usually time-consuming and hinders their wide adoption. In pairing-based zkSNARK constructions (e.g., [5, 7, 8, 12]), the proof consists of several points in an elliptic curve group which operate with each other within of this group.

One of the main computational bottlenecks of such zkSNARK constructions lies in the used multiscalar multiplication (MSM) method. Let $S_{n,r}$ be the following n-scalar multiplication over fixed points $P_1, \ldots, P_n$,

$$S_{n,r} = \sum_{i=1}^{n} a_i P_i, \tag{1}$$

where $a_i \in [0, r), i = 1, \ldots, n$ are integers. In the following, r will denote the order of the (elliptic curve) group where all these computations take place.

In this paper, we conduct a comprehensive analysis of existing multi-scalar multiplication (MSM) methods, focusing on their application in ZK-SNARK verification. Our objective is to enhance these methods by optimizing performance specifically for ZK-SNARKs, without the need for constant-time countermeasures, thereby improving efficiency and security in cryptographic operations.

1.1 Existing MSM Computation Methods

In the last three decades, various methods have been proposed to accelerate MSM computation, and most of them utilize precomputation.

Straus Method To compute $S_{n,r}$, the Straus method precomputes 2^{nc} points

$$\Big\{ \sum_{i=1}^{n} b_i P_i | \forall b_i \in [0, 2^c - 1], i \in [1, n] \Big\}$$

where c is a small integer. Next, the algorithm divides each a_i from (1) into segments of length c, i.e.

$$a_i = a_{i,h-1} \| a_{i,h-2} \| \cdots \| a_{i,1} \| a_{i,0} = \sum_{j=0}^{h-1} a_{i,j} 2^{jc}, i \in [1, n]$$

where $h = \lceil \log_2(r)/c \rceil$ and $0 \leqslant a_{ij} < 2^c$ for $j \in [1, h-1]$ for $1 \leqslant j \leqslant h-1$. The algorithm retrieves the point

$$S_{n,2^c} = \sum_{i=1}^{n} a_{i,h-1} P_i$$

from the precomputation table, doubles it c times, adds the precomputed point $\sum_{i=1}^{n} a_{i,h-2} P_i$ to obtain

$$S_{n,2^{2c}} = \sum_{i=1}^{n} (a_{i,h-1} \| a_{i,h-2}) P_i.$$

After $h-1$ repetitions, we obtain

$$S_{n,2^{hc}} = \sum_{i=1}^{n} (a_{i,h-1} \| a_{i,h-2} \| \ldots \| a_{i,0}) P_0 \tag{2}$$

Pippenger's Bucket Method This method proceeds in the same way as in the Straus method except for the computation of

$$S_{n,2^c} = \sum_{i=1}^{n} a_{i,j} P_i,$$

where $j \in [0, h-1]$, $h = \lceil \log_2(r)/c \rceil$. First, the method sorts all points into (2^c-1) buckets with respect to their scalars. Let S_i denote the intermediate subsum of points corresponding to scalar i. The algorithm computes all S_i, for $i \in [1, 2^c-1]$ and finally it computes $S_{n,2^c} = \sum_{i=1}^{2^c-1} i \cdot S_i$ using at most $2(2^c-2)$ additions.

Luo-Fu-Gong (LFG) MSM Method [11] Let M be a set of integers and B be a set of non-negative integers containing zero. Given scalars $a_i, 0 \leqslant a_i < r$, the LFG method first computes [11, Algorithm 6] a radix q representation

$$a_i = \sum_{j=0}^{h-1} a_{ij} q^j$$

where $h = \lceil \log_q r \rceil$, and for every $i \in [1, n], j \in [0, h-1]$,

$$a_{ij} = \epsilon_{ij} m_{ij} b_{ij}, \text{ where } \epsilon_{ij} \in \{\pm 1\}, m_{ij} \in M, b_{ij} \in B \ .$$

Then, $S_{n,r}$ can be computed as:

$$\begin{aligned} S_{n,r} &= \sum_{i=0}^{n} a_i P_i = \sum_{i=1}^{n} \big(\sum_{j=0}^{h-1} a_{ij} q^j \big) P_i \\ &= \sum_{i=1}^{n} \big(\sum_{j=0}^{h-1} \epsilon_{ij} m_{ij} b_{ij} q^j \big) P_i = \sum_{i=1}^{n} \sum_{j=0}^{h-1} b_{ij} \epsilon_{ij} m_{ij} q^j P_i \end{aligned} \tag{3}$$

Let $P_{ij} = \epsilon_{ij} m_{ij} q^j P_i$. Then $S_{n,r}$ can be computed as follows.

$$\begin{aligned} S_{n,r} &= \sum_{i=1}^{n} \sum_{j=0}^{h-1} b_{ij} P_{ij} = \sum_{i=1}^{n} \sum_{j=0}^{h-1} \big(\sum_{k \in B} k \cdot \sum_{i,j \text{ s.t. } b_{ij}=k} P_{ij} \big) \\ &= \sum_{k \in B} k \cdot \big(\sum_{i=1}^{n} \sum_{j=0}^{h-1} \sum_{i,j \text{ s.t. } b_{ij}=k} P_{ij} \big). \end{aligned} \tag{4}$$

Assume that there are $nh|M|$ such points which are defined as

$$\{ m q^j P_i | 1 \leqslant i \leqslant n, 0 \leqslant j \leqslant h-1, m \in M \}$$

These points are precomputed (we don't need to precompute their opposites, see below). Then define intermediate subsums

$$S_k = \sum_{i=1}^{n} \sum_{j=0}^{h-1} \sum_{i,j \text{ s.t. } b_{ij}=k} P_{ij}, k \in B \quad .$$

All S_k's can be computed with at most $nh - (|B| - 1)$ additions and the remainder is computed by Algorithm 1 with at most $2(|B| - 1) + d - 3$ additions, where d is the maximum difference between the two neighboring elements in B. The total cost of computing $S_{n,r}$ is therefore at most

$$nh + |B| + d - 4 \tag{5}$$

elliptic curve additions.

Let M denote a set of multipliers, which are used to generate the precomputed points. Furthermore, the set B is called a bucket set which contains sorted points. In Algorithm 2 we recall the MSM algorithm of [11]. Note that Luo, Fu and Gong merge the multiplier set with the units ± 1, so that their M is in fact what for us will be $M \cup -M$.

1.2 Our contribution

Our contribution in this paper is threefold:

(1) We point out a problem with Property 1 in the MSM method in [11], which proposes a new decomposition of scalars in base $q = 2^c$ and states the following: given a power of two integer $q = 2^c$ (for $10 \leqslant c \leqslant 31$), for all $0 \leqslant t \leqslant q$

Algorithm 1 Subsum accumulation algorithm [11]

```
**Input:** $1 \leqslant b_1 \leqslant b_2 \leqslant \ldots \leqslant b_{|B|}, S_1, S_2, \ldots, S_{|B|}$
**Output:** $S = b_1 S_1 + \cdots + b_{|B|} S_{|B|}$
Define a length-$(d+1)$ array $\mathsf{tmp} = [0] \times (d+1)$
**for** $i = |B|$ to 1 by $-1$ **do**
    $\mathsf{tmp}[0] = \mathsf{tmp}[0] + S_i$
    $k = b_i - b_{i-1}$
    **if** $k \geqslant 1$ **then**
        $\mathsf{tmp}[k] = \mathsf{tmp}[k] + \mathsf{tmp}[0]$
**return** $1 \cdot \mathsf{tmp}[1] + 2 \cdot \mathsf{tmp}[2] + \cdots + d \cdot \mathsf{tmp}[d]$
```

there exists a value $b \in B$ and a multiplier $m \in \{1, 2, 3\}$ such that $t = mb$ or $q - t = mb$. This decomposition is then used in a modified Pippenger bucket algorithm. The main idea behind this property is to remove redundant points from an initially defined set B_0 to obtain a new set B_1. This can be done for the purpose of saving computational costs, since in an elliptic curve group, $-P_i = (x_i, -y_i)$ can be determined on the fly from the computed points $P_i = (x_i, y_i)$. The algorithm provided by Luo, Fu, and Gong [11] for constructing B_1 from B_0 discards all elements of the form $q - 2i$ and $q - 3j$ for all values $i, j \in B_0$ and $q/4 \leqslant i < q/2$ and $q/6 \leqslant j < q/4$. We provide a counterexample that shows Property 1 of [11] (we will denote it the "LFG Property 1") does not always hold. Furthermore, we provide a general construction of counterexamples and prove that the LFG Property 1 is false for at least $q/(216) + O(1)$ integers. We also show that the LFG Property 1 is correct for all odd t, where $0 < t < q$.

(2) Our second contribution is to provide a method to fix the LFG Property 1. We present a new construction of the set B_1 by first removing elements of the form $q - 2i$ and $q - 3j$ from the initial set B_0 and then adding all elements of the form $q - 6k \in B_0$ with $k \notin B_0$. Next, we show how to modify the LFG Property 1 to obtain a MSM algorithm with optimal runtime. The modification involves an element $\epsilon_t \in \{\pm 1\}$ such that $t \equiv \epsilon_t m_t b_t \mod q$ for a multiplier $m_t \in M$ and a bucket element $b_t \in B$. While the achieved running time of the MSM algorithm in [11] is $(nh + 0.21q) \cdot \mathsf{Add}$, for $q = 2^c$ and $10 \leqslant c \leqslant 31$ and Add denoting the number of point additions on an elliptic curve, we stress out that the LFG Property 1 leaves out some values of t, therefore the time complexity of the LFG bucket method cannot be taken as a benchmark. In contrast to [11] the scalars decomposition property holds for all $0 < t < q$ and achieves the same space and running time complexity for $q = p^c$ and $4 \leqslant c \leqslant 11$ (see Table 1).

(3) In our third contribution we use efficient endomorphisms to achieve better running time complexity and to save storage space. We use an endomorphism ω such that $\omega^3(P) = P$ for all elliptic curve points P. Since it holds that $\omega(x, y) = (\zeta_3 x, y)$ for a complex cube root of unity $\zeta_3 \in \mathbb{F}_p$, i.e. $\zeta_3^3 = 1$, the computation of $\omega(P)$ can be done on the fly leading to significant savings of the storage cost of these points. The endomorphism ring of such elliptic

Algorithm 2 Multi-scalar multiplication over fixed points [11]

```
Input: Scalars $a_1, a_2, \dots, a_n$, fixed points $P_1, P_2, \dots, P_n$, radix $q$, scalar length $h$, multiplier set $M = \{m_0, m_1, \dots, m_{|M|-1}\}$, bucket set $B = \{b_0, b_1, \dots, b_{|B|-1}\}$.
Output: $S_{n,r} = \sum_{i=1}^{n} a_i P_i$
Precompute a length-$nh|M|$ point array precomputation, s.t. precomputation $[|M|((i-1)h + j) + k] = m_k q^j P_i$.
Precompute a hash table mindex to record the index of every multiplier, s.t. mindex$[m_k] = k$. Precompute a hash table bindex to record the index of every bucket, such that bindex$[b_k] = k$.
Convert every $a_i$ to its standard q-ary form, then convert it to $a_i = \sum_{j=0}^{h-1} m_{ij} b_{ij} q^j$.
Create a length-$nh$ scalar array scalars, s.t. scalars$[(i-1)h + j] = b_{ij}$. Create a length-$nh$ array points recording the index of points, such that points$[(i-1)h+j] = |M|((i-1)h + j) +$ mindex$[m_{ij}]$. $n$-scalar multiplication $S_{n,r}$ is equivalent to the following $nh$-scalar multiplication
    $$\sum_{i=0}^{nh-1} \mathsf{scalars}[i] \cdot \mathsf{precomputation}[\mathsf{points}[i]],$$
    where every scalar in scalars is from bucket set $B$.
Create a length-$|B|$ point array buckets to record the intermediate subsums, and initialize every point to infinity. For $0 \leqslant i \leqslant nh-1$, add point precomputation[points[$i$]] to bucket buckets[bindex[scalars[$i$]]].
Invoke Algorithm 1 to compute $\sum_{i=0}^{|B|-1} b_i \cdot \mathsf{buckets}[i]$, return the result.
```

curves is isomorphic to $\mathbb{Z}[\omega]$. With this in mind we update the scalar decomposition property to adapt it to the new setting, where $t \in \mathbb{Z}[w]$. We implement our idea with $p = 2 - \omega$, the multiplier set $M = \{1\}$ and $q = p^c$, where $p \mid 7$ in $\mathbb{Z}[\omega]$. With this approach the storage cost of $n(h+1)$ curve points is equivalent to Pippenger's original method and the time complexity is reduced from $(nh + 0.5q) \cdot \mathsf{Add}$ to $(n(h+1) + 0.2015\,|q|^2 + 20) \cdot \mathsf{Add}$[1].

Finally we confirm our result for the fixed LFG Property 1 and the new bucket set constructions by implementing the MSM algorithm in C++. To enable a fair comparison with the LFG approach we use the same elliptic curve BLS12-381. We measure the space complexity in terms of the number of stored elliptic curve points P. Therefore, the expression $nh \cdot P$ indicates that a total of nh curve points are stored.

[1] In the endomorphism case, the norm $|q|^2$ plays the rôle of q in the original case. They are both of the same size, with equal corresponding values of h

[2] Notice that the time and space complexities in [11] are provided for their incomplete bucket set B. Since the repaired bucket set B contains more points, the complexities would be comparable to our results.

[3] This row normally has $h+1$ instead of h. However, in all cases except when $q = p^7$, we were able to prove the better figures. See the remark before Theorem 6.

Table 1. Comparison of Different MSM Algorithms, for $q = p^c,\ 4 \leqslant c \leqslant 11$.

Method	Space Complexity	Time Complexity (Worst Case)
Pippenger [2,13]	$n \cdot P$	$h(n + 0.5q) \cdot \mathsf{Add}$
Pippenger variant [4]	$nh \cdot P$	$(nh + 0.5q) \cdot \mathsf{Add}$
LFG [11], $p = 2^2$	$3nh \cdot P$	$(nh + 0.21q) \cdot \mathsf{Add}$
Repaired LFG Method, $p = 2$	$\boldsymbol{(3nh + n) \cdot P}$	$\boldsymbol{(n(h+1) + 0.21875q)} \cdot \mathsf{Add}$
Our Method for a prime $p > 2$	$\boldsymbol{n(h+1)\lvert M\rvert \cdot P}$	$\boldsymbol{(n(h+1) + q/(2\lvert M\rvert) + p - 4)} \cdot \mathsf{Add}$
Our Method with endomorphisms[3] for $\lvert M\rvert = \{1\}$	$\boldsymbol{nh \cdot P}$	$\boldsymbol{(nh + 0.2015\,\lvert q\rvert^2 + 20)} \cdot \mathsf{Add}$

2 Analysis of the LGF MSM Algorithm

In this section, we first point out an issue with the MSM method of [11], especially with their Property 1 on p. 369, and then propose a fix.

2.1 Background of Property 1 in [11]

For a prime p and a positive integer n, define $\mathrm{ord}_p(n)$ to be the integer $e \geqslant 0$ such that $n = p^e k$ with $p \nmid k$. Luo, Fu and Gong define a new decomposition of scalars in base $q = 2^c$ for $c \in \mathbb{N}$ to which they can apply a modified Pippenger bucket algorithm.

They start by first defining the set B_0 as follows:

$$B_0 = \{0\} \cup \{b \in \mathbb{N} \colon 1 \leqslant b \leqslant q/2, \mathrm{ord}_2(b) + \mathrm{ord}_3(b) \equiv 0 \pmod 2\} \quad .$$

Note that for any integer $1 \leqslant t \leqslant q/2$, there exists an $m \in \{1, 2, 3\}$ and $b \in B_0$ such that $t = mb$. Indeed, if $t \notin B_0$, then $2 \mid t$, in which case $b = t/2 \in B_0$ or $3 \mid t$, in which case $b = t/3 \in B_0$.

The authors want to take advantage of the fact that, in an elliptic curve group, opposites of points can be computed on the fly at virtually no cost, allowing for m to be chosen from $\{\pm 1, \pm 2, \pm 3\}$. This leads to the removal of redundant representations from B_0 as shown in Algorithm 3 by discarding from B_0 all elements of the form $q - 2i$ and $q - 3j$ for $i, j \in B_0$, with $q/4 \leqslant i < q/2$ and $q/6 \leqslant j < q/4$. The resulting set carved out of B_0 will be called B_1^{old} (B_1 in [11], but we will reserve this notation to our later fix), and the following property is claimed computationally for $B = B_1^{\mathrm{old}}$.

Algorithm 3 Construction of auxiliary set B_1^{old} in [11]

```
Input: $B_0, q$
Output: $B_1^{\mathrm{old}}$
$B_1^{\mathrm{old}} = B_0$
for $i = \frac{q}{4}$ to $\frac{q}{2} - 1$ do
    if $i \in B_0$ and $q - 2i \in B_0$ then
        $B_1^{\mathrm{old}} = B_1^{\mathrm{old}} - \{q - 2i\}$
for $i = \lfloor \frac{q}{6} \rfloor$ to $\frac{q}{4} - 1$ do
    if $i \in B_0$ and $q - 3i \in B_0$ then
        $B_1^{\mathrm{old}} = B_1^{\mathrm{old}} - \{q - 3i\}$
return $B_1^{\mathrm{old}}$
```

Property 1 (Property 1 in [11])**.** *Given* $q = 2^c$ $(10 \leqslant c \leqslant 31)$, *for all* $0 \leqslant t \leqslant q$, *there exist* $b \in B$ *and* $m \in \{1, 2, 3\}$ *such that*

$$t = mb \quad \textit{or} \quad q - t = mb \ . \tag{6}$$

We now provide counterexamples to this property, when $B = B_1^{\mathrm{old}}$. In fact, large families of counterexamples can be constructed for all such q. For instance, let $q = 2^{10} = 1024$ and $t = 292$. Note that

$$t = 2^2 \cdot 73 \in B_0 \quad \text{and} \quad q - t = 732 = 3 \cdot 2^2 \cdot 61 = 3j \ ,$$

with $j = 244 = 2^2 \cdot 61 \in B_0, 170 = \lfloor q/6 \rfloor < j < q/4 = 256$. Hence $t \in B_0$, so that $m = 1$ in $t = mb$ in (6), but $t \notin B_1^{\mathrm{old}}$.

On the other hand,

$$j = 2^2 \cdot 61 = q - 2^2 \cdot 3 \cdot 5 \cdot 13 = q - 2i \ ,$$

with $i = 390 = 2 \cdot 3 \cdot 5 \cdot 13 \in B_0, 256 = q/4 \leqslant i < q/2 = 512$, hence $j \notin B_1^{\mathrm{old}}$. Similarly,

$$q - t = 2i' \ ,$$

where $i' = 366 = 2 \cdot 3 \cdot 61 \in B_0$, and

$$i' = q - 2 \cdot 7 \cdot 47 = q - 2 \cdot i'' \ ,$$

with $i'' = 329 = 7 \cdot 47 \in B_0, 256 = q/4 \leqslant i'' < q/2 = 512$, hence $i' \notin B_1^{\mathrm{old}}$. Since $q - t = mb$ in (6) implies that $m = 2$ (resp. 3), and $b = i'$ (resp. $b = j$) is not in B_1^{old}, we conclude that $t = 292$ doesn't satisfy Property 1 for $q = 1024$.

2.2 General Construction of Counterexamples

We show the general construction of counterexamples for Property 1 of [11].

Proposition 1. *Property 1 is false for at least* $\dfrac{q}{216} + O(1)$ *integers* $0 \leqslant t \leqslant q$.

Proof. As above, we let $q = 2^c$ for an integer $c \geqslant 10$. Pick any integer n that is coprime to 6, with $\frac{4q}{72} < n < \frac{5q}{72}$. Note that among six consecutive integers, at least two will be coprime to 6, i.e. the neighbors of a multiple of 6. Hence, there are at least

$$\frac{q}{216} + O(1)$$

such values of n. Define $j = 6n$ and $\theta = 3(q/2 - j)$. Note that

$$\frac{q}{3} < j < \frac{5q}{12} \quad \text{and} \quad \operatorname{ord}_2(j) \equiv \operatorname{ord}_3(j) \equiv 1 \pmod 2 \ ,$$

hence

$$\frac{q}{4} < \theta < \frac{q}{2} \quad \text{and} \quad \operatorname{ord}_2(\theta) \equiv \operatorname{ord}_3(\theta) \equiv 1 \pmod 2 \ ,$$

so that $j, \theta \in B_0$. Note that in particular, $3 \mid \theta$. Let $t = q - 2\theta$. We will show $0 < t < q/2$ does not satisfy Property 1.

First, $3 \nmid t$ and $\operatorname{ord}_2(t) = \operatorname{ord}_2(\theta) + 1 \equiv 0 \pmod 2$, therefore $t \in B_0$. However, since $\theta \in B_0$, $t \notin B_1^{\text{old}}$. Any expression $t = mb$ with $m \in \{1, 2, 3\}$ and $b \in B_1$ must therefore be excluded in Property 1. We now exclude that

$$q - t = mb, \quad \text{for some } b \in B_1^{\text{old}} \text{ and } m \in \{1, 2, 3\} \ .$$

Since $q - t > q/2$, $q - t \notin B_0$, hence $q - t \notin B_1^{\text{old}}$.

We have $q - t = 2\theta$. Write $\theta = q - 2i$ and note $3 \nmid i$, $\operatorname{ord}_2(i) \equiv 0 \pmod 2$. Also $0 < i < q/2$, therefore $i \in B_0$ and $\theta \notin B_1^{\text{old}}$.

Finally, define $\psi = \frac{q-t}{3} = 2\theta/3$. Then $\operatorname{ord}_2(\psi) \equiv \operatorname{ord}_3(\psi) \equiv 0 \pmod 2$, and since trivially $0 < \psi < q/3$, we get $\psi \in B_0$. On the other hand, by definition of θ,

$$\psi = q - 2j \ ,$$

where we saw $j \in B_0$. Therefore $\psi \notin B_1^{\text{old}}$ thus concluding our proof. □

2.3 Property 1 Holds for Odd $0 < t < q$

We can nevertheless show that the following is true.

Proposition 2. *Property 1 in [11] with $B = B_1^{\text{old}}$ holds whenever $0 \leqslant t \leqslant q$ is odd.*

Proof. Replacing t by $q - t$ if necessary, we can suppose that $t < q/2$. There are several cases to consider.

$3 \nmid$ t and $3 \nmid$ q − t: In this case, $t \in B_0$ and, since we can't write $t = q - 2n$ or $q - 3n$ with $n \in \mathbb{N}$, it follows that $t \in B_1^{\text{old}}$.

$3 \nmid$ t and $3 \mid$ q − t: Again, $t \in B_0$. If $t \notin B_1^{\text{old}}$, then we could write

$$t = q - 3i_3 \ , \quad i_3 < \frac{q}{4} \ , i_3 \in B_0 \ .$$

Similarly, if $q - t = 3i_3$ and $i_3 \notin B_1^{\mathrm{old}}$, then

$$i_3 = q - 3i_3' \quad , \quad i_3' < \frac{q}{4} \ , i_3' \in B_0 \ \ .$$

We reach a contradiction, since

$$q = i_3 + 3i_3' < \frac{q}{4} + \frac{3q}{4} = q \ \ .$$

$\mathbf{3 \mid t}$ and $\mathbf{3 \nmid q - t}$: This is the last possible case. Either $t \in B_0$ and then we can reason as in the first case to deduce that $t \in B_1^{\mathrm{old}}$; or

$$t = 3b \ \ , b \in B_0 \ \ .$$

If $b \notin B_1^{\mathrm{old}}$, then

$$b = q - 3j_3 \quad , \quad j_3 < \frac{q}{4} \ , j_3 \in B_0 \ \ .$$

This is again impossible, since $b = t/3 < q/6$, resulting in

$$q = b + 3j_3 < \frac{q}{6} + \frac{3q}{4} < q \ \ .$$

□

3 Repairing Property 1 of [11]

We show in this section how to change the definition of B_1^{old} so that Property 1 holds.

We construct the new set B_1 in Algorithm 4, by first removing from B_0 as before all elements of the form $q - 2i > 0$ and $q - 3j$, for $i, j \in B_0$, and $j < q/4$. We then add back all elements $q - 6k \in B_0$ with $k \notin B_0$.

Proposition 3. *Property 1 with $B = B_1$ holds for all $0 \leqslant t \leqslant q$.*

Proof. Because Property 1 is symmetric in $t \leftrightarrow q - t$, we can suppose $3 \nmid t$. Also, in view of Proposition 2 we only need suppose that $0 \neq t$ is even, since $B_1^{\mathrm{old}} \subseteq B_1$. We have two cases:

$\mathrm{ord}_2(t)$ even: If $t \leqslant q/2$, then $t \in B_0$. If $t \notin B_1^{\mathrm{old}}$, then either $t = q - 3j$ with $j \in B_0, j < q/4$, or $t = q - 2i$ with $i \in B_0$.

In the first case, we have $2 \mid j$, hence

$$t = q - 3j \Longrightarrow t = q - 2i \quad \text{with } i = \frac{3j}{2} \in B_0 \ \ .$$

We therefore only need focus on $t = q - 2i$, where $i \in B_0$. Since $\mathrm{ord}_2(t) = \mathrm{ord}_2(2i) = \mathrm{ord}_2(i) + 1$ we deduce that $\mathrm{ord}_3(i) \equiv 1 \pmod 2$, in particular that $3 \mid i$. Calling $k = i/3$, we have $k \notin B_0$ and $t = q - 6k$, therefore $t \in B_1$.

On the other hand, if $t > q/2$, then, since $q - t < q/2$ and $\mathrm{ord}_2(q-t) = \mathrm{ord}_2(t)$, if $\mathrm{ord}_3(q-t)$ is even, then, reasoning as above with $q - t \in B_0$ in place of t, we find that $q - t \in B_1$.

If $t > q/2$ and $\mathrm{ord}_3(q-t)$ is odd, then $q - t = 3b$ with $b \in B_0$. If $b \notin B_1^{\mathrm{old}}$, then either $b = q - 3j$ with $j \in B_0, j < q/4$, or $b = q - 2i$ with $i \in B_0$. The former case is impossible, since we would get the contradiction

$$q = b + 3j < \frac{q}{6} + \frac{3q}{4} < q \ .$$

In the latter case,

$$b = q - 2i \quad \text{with } i \in B_0 \ .$$

As above $\mathrm{ord}_2(i) + 1 = \mathrm{ord}_2(b) = \mathrm{ord}_2(q-t) = \mathrm{ord}_2(t)$, therefore $\mathrm{ord}_2(i)$ is odd and $\mathrm{ord}_3(i)$ is odd; in particular $3 \mid i$. Writing $k = i/3$, we have $k \notin B_0$ and $b = q - 6k$, therefore $b \in B_1$.

$\mathrm{ord}_2(t)$ **odd:** Then $t = 2b$, where $b \in B_0$. If $b \notin B_1^{\mathrm{old}}$, then either $b = q - 3j$ with $j \in B_0, j < q/4$, or $b = q - 2i$ with $i \in B_0$. In the latter case, as before, $\mathrm{ord}_2(i) \equiv 1 \pmod 2$ and therefore $3 \mid i$. Calling $k = i/3$, we have $k \notin B_0$ and $b = q - 6k$, therefore $b \in B_1$.

The former case is slightly more complicated, where we have

$$b = q - 3j \ , \quad j < \frac{q}{4} \ , j \in B_0 \ . \tag{7}$$

If $2 \mid j$, then, calling $k = j/2 \notin B_0$, we find that $b \in B_1$ as before. It may however be the case that $2 \nmid j$, that is $2 \nmid b$. By (7), $b \equiv q \pmod 3$, hence $q \not\equiv t = 2b \pmod 3$. In this case, $\mathrm{ord}_2(q-t) = \mathrm{ord}_2(t) = 1$, hence $b' = (q-t)/2 \in B_0$. If $b' \notin B_1^{\mathrm{old}}$, then, noticing that $2 \nmid b'$,

$$b' = q - 3j' \ , \quad j' < \frac{q}{4} \ , j' \in B_0 \ . \tag{8}$$

Putting (7) and (8) together,

$$q - 3j' = b' = \frac{q-t}{2} = \frac{q}{2} - b = 3j - \frac{q}{2} \ ,$$

from which

$$3j + 3j' = \frac{3q}{2} \iff j + j' = \frac{q}{2} \ ,$$

which is impossible because $j, j' < q/4$.

□

We define the new $B = B_1$, which will allow us to prove a precise estimate of its cardinality, as now B no longer depends on the elliptic curve. Table 2 in the appendix lists our new bucket set constructions for $q = 2^c, 10 \leqslant c \leqslant 31$.

Algorithm 4 Construction of the new auxiliary set B_1

```
Input: $B_0, q$
Output: $B_1$
$B_1 = B_0$
for $\frac{q}{4} \leqslant i < \frac{q}{2}$ do
    if $i \in B_0$ and $q - 2i \in B_0$ then
        $B_1 = B_1.\text{remove}(q - 2i)$
for $\frac{q}{6} \leqslant i < \frac{q}{4}$ do
    if $i \in B_0$ and $q - 3i \in B_0$ then
        $B_1 = B_1\text{remove}(q - 3i)$
for $\frac{q}{12} \leqslant i < \frac{q}{6}$ do
    if $i \notin B_0$ and $q - 6i \in B_0$ then
        $B_1 = B_1.\text{append}(q - 6i)$
return $B_1$
```

3.1 Analysis of the Size of B

The set $B = B_1$ can also be constructed by removing the following two subsets from B_0:

1. $B_2 = \{t = q - 2i \in B_0 \colon i \in B_0,\ i < q/2,\ 3 \nmid i\}$, and
2. $B_3 = \{\theta = q - 3j \in B_0 \colon j \in B_0,\ j < q/4,\ 2 \nmid j\}$.

The sets B_2 and B_3 are disjoint, since all elements of the former are even, while all elements of the latter are odd.

Lemma 1. *The cardinalities of the sets B_2 and B_3 (denoted as $|B_2|$ and $|B_3|$) satisfy*

$$|B_2| = \left|\left\{1 \leqslant t \leqslant \frac{q}{2} \colon \operatorname{ord}_2(t) \equiv \operatorname{ord}_3(t) \equiv 1 \pmod 2\right\}\right| ,$$

and

$$|B_3| = \left|\left\{\frac{q}{2} < u \leqslant \frac{3q}{4} \colon 2 \nmid u,\ \operatorname{ord}_3(u) \equiv 1 \pmod 2\right\}\right| .$$

Proof. If $t \in B_2$, $\operatorname{ord}_2(t) \equiv \operatorname{ord}_2(i) + 1 \equiv 1 \pmod 2$, hence $\operatorname{ord}_3(t) \equiv 1 \pmod 2$. Vice-versa, if $0 < t \leqslant q/2$ satisfies $\operatorname{ord}_2(t) \equiv \operatorname{ord}_3(t) \equiv 1 \pmod 2$, then $t \in B_2$. This shows that B_2 can in fact be described by the set on the right-hand side of the first equation of the lemma.

Regarding B_3, whenever $\theta = q - 3j \in B_0$ with $j \in B_0$, $j < q/4$ and j odd, then $q/4 < \theta \leqslant q/2$, and therefore $q/2 \leqslant u = q - \theta < 3q/4$, $\operatorname{ord}_2(u) = 0$ and $\operatorname{ord}_3(u) \equiv 1 \pmod 2$. Vice-versa, any odd $q/2 \leqslant u < 3q/4$ (note that u cannot equal any of those end values) such that $\operatorname{ord}_3(u)$ is odd will correspond to $\theta = q - u \in B_3$. □

Lemma 2. *Let $Q \in \mathbb{N}$, e, f be nonnegative integers. Define*

$$S_Q^{e,f} = \{1 \leqslant t \leqslant Q \colon \operatorname{ord}_2(t) = e,\ \operatorname{ord}_3(t) = f\},$$

then

$$|S_Q^{e,f}| = \frac{Q}{2^e 3^{f+1}} + O(1) \ .$$

Proof. By the inclusion-exclusion principle,

$$\begin{aligned} S_Q^{e,f} &= \{1 \leqslant t \leqslant Q \colon 2^e 3^f \mid t\} - \{1 \leqslant t \leqslant Q \colon 2^{e+1} 3^f \mid t\} \\ &\quad - \{1 \leqslant t \leqslant Q \colon 2^e 3^{f+1} \mid t\} \cup \{1 \leqslant t \leqslant Q \colon 2^{e+1} 3^{f+1} \mid t\} \ . \end{aligned}$$

Taking cardinalities,

$$\begin{aligned} |S_Q^{e,f}| &= \left\lfloor \frac{Q}{2^e 3^f} \right\rfloor - \left\lfloor \frac{Q}{2^{e+1} 3^f} \right\rfloor - \left\lfloor \frac{Q}{2^e 3^{f+1}} \right\rfloor + \left\lfloor \frac{Q}{2^{e+1} 3^{f+1}} \right\rfloor \\ &= \left(\frac{1}{2^e 3^f} - \frac{1}{2^{e+1} 3^f} - \frac{1}{2^e 3^{f+1}} + \frac{1}{2^{e+1} 3^{f+1}} \right) Q + O(1) \\ &= \frac{Q}{2^e 3^f} \left(1 - \frac{1}{2} \right) \left(1 - \frac{1}{3} \right) + O(1) \\ &= \frac{Q}{2^e 3^{f+1}} + O(1) \ . \end{aligned}$$

□

The Size of $B_0 \backslash B_2$ Applying Lemma 2 with $Q = q/2$, we compute

$$\begin{aligned} |B_0 \backslash B_2| &= \sum_{\substack{e \geqslant 0 \text{ even} \\ f \geqslant 0 \text{ even}}} |S_{q/2}^{e,f}| = \frac{q}{2} \sum_{\substack{0 \leqslant e \leqslant \log_2 q \\ e \text{ even}}} \sum_{\substack{0 \leqslant f \leqslant \log_3 q \\ f \text{ even}}} \frac{1}{2^e 3^{f+1}} + O(\log^2 q) \\ &= \frac{q}{6} \sum_{\epsilon \geqslant 0} \frac{1}{4^\epsilon} \sum_{\phi \geqslant 0} \frac{1}{9^\phi} + O(\log^2 q) = \frac{q}{4} + O(\log^2 q) \ . \end{aligned} \tag{9}$$

The Size of B_3 Similarly, by applying Lemma 2 with $Q = 3q/4, q/2$, and we have,

$$\begin{aligned} |B_3| &= \sum_{f \geqslant 0 \text{ odd}} |S_{3q/4}^{0,f}| - |S_{q/2}^{0,f}| \\ &= \left(\frac{3q}{4} - \frac{q}{2} \right) \sum_{f \geqslant 0 \text{ odd}} \frac{1}{3^{f+1}} + O(\log q) \\ &= \frac{q}{32} + O(\log q) \ . \end{aligned} \tag{10}$$

Computation of the Size of B Since $B = (B_0 \backslash B_2) \backslash B_3$ and $B_2 \cap B_3 = \varnothing$, using (9) and (10), we compute

$$|B| = |B_0 \backslash B_2| - |B_3| = \frac{q}{4} - \frac{q}{32} + O(\log^2 q) = \frac{7q}{32} + O(\log^2 q) \ ,$$

where $q = 2^c$. Note that $7/32 = 0.21875$.

3.2 The Maximum Difference Between Neighbors of B

Proposition 4. *Let $b_1 < b_2 < \cdots < b_{|B|}$ denote the elements of B, sorted in increasing order. Then for all $1 \leqslant r < |B|$,*

$$b_{r+1} - b_r \leqslant 6.$$

Proof. Consider the set

$$S = \{m \leqslant q/2 \colon m \equiv \pm 1 \pmod 6\} \ .$$

Then $S \subseteq B_0$, since integers in S are coprime to 6. Let $m \in S$. If $m \notin B$, then $m \in B_3$ so that $m \equiv q \pmod 3$. The neighbors $m_- < m < m_+$ of m in S are spaced in such a way that

$$\{m - m_-, m_+ - m\} = \{2, 4\} \ .$$

But then $m_\pm \not\equiv q \pmod 3$, hence $m_\pm \notin B_3$ and therefore $m_\pm \in B$. This shows that consecutive elements of B are never more than 6 integers apart. □

3.3 On the Length of the Recoding

The original LFG method called for an additional set (called B_2 in [11]) to specifically force the scalar recoding to be of the same length as its q-ary expansion, see [11, Algorithm 6]. With our modification (Algorithm 5), the length can be one digit longer, namely $h + 1$. However, this last digit a_h can only be 0 or 1, therefore, only points $mq^j P_i$ and $q^h P_i$ for $m \in M, 0 \leqslant j \leqslant h-1, 1 \leqslant i \leqslant n$ need to be precomputed, for a total of $3nh + n$ points.

Algorithm 5 Adjusted scalar recoding

```
Input: $\{a_j\}_{0 \leqslant j \leqslant h-1}$, $0 \leqslant a_j < q$ such that $a = \sum_{j=0}^{h-1} a_j q^j$.
Output: $\{(\epsilon_j m_j, b_j)\}_{0 \leqslant j \leqslant h}$, $\epsilon_j \in \{\pm 1\}$, $m_j \in M$, $b_j \in B$ such that $a = \sum_{j=0}^{h} \epsilon_j m_j b_j q^j$.
$a_h \leftarrow 0$
for $j = 0$ to $h - 1$ do
    Obtain $\epsilon_j \in \{\pm 1\}, m_j \in M, b_j \in B, \alpha_j \in \{0, 1\}$ such that $a_j = \epsilon_j m_j b_j + \alpha_j q$ in (1)
    $a_{j+1} = \alpha_j + a_{j+1}$                ▷ Note that $a_h = 0$ or 1
return $\{(\epsilon_j m_j, b_j)\}_{0 \leqslant j \leqslant h}$
```

4 Construction of Optimal Bucket Sets for Efficient MSM Computation

In this section, we generalize the LFG construction of bucket sets B to provide examples of optimal-sized sets. We start by generalizing Property 1.

Property 2. *Let p be a prime. Given $q = p^c$, for all $0 \leqslant t \leqslant q$, there exist $b \in B$ and $m \in M$ such that*

$$t = mb \quad or \quad q - t = mb \ .$$

We refer to B as the bucket set and M as the (unsigned) multiplier set. In the previous sections, $B = B_1$, $M = \{1, 2, 3\}$ and $p = 2$. In this context, a simple cardinality argument shows the following.

Theorem 1 (Lower bound on the bucket set size)**.** *If Property 2 holds, then*

$$2 \cdot |B| \cdot |M| \geqslant q \ .$$

Proof. We remark that there are at most $|B| \cdot |M|$ integers t of the form mb for $m \in M, b \in B$, and similarly for the t's of the form $q - mb$. The conclusion follows: if Property 2 holds, all $0 \leqslant t \leqslant q$ must be representable in one of these two ways. □

The set M determines the number of precomputed points[4], which, in the notation of LFG is $|M|nh$. Hence, if $|M| = 2$, then $|B| \geqslant q/4$ and if $|M| = 3$, $|B| \geqslant q/6$. The case $|M| = 1$ is Pippenger's variant, which is therefore optimal [11, Table 1]. We now describe an optimal bucket set B for $|M| = 2$, satisfying $|B| = q/4 + O(1)$.

4.1 Optimal Bucket Set for $|M| = 2$

Let p be an odd prime and $q = p^c$ for $c \in \mathbb{N}$. Define $M = \{1, 2\}$ and let B consist of 0 together with all integers $0 < b < q/2$ of the form $b = p^k\beta$, where $0 \leqslant k \leqslant c - 1$ is an integer and β is a quadratic residue mod q (in particular, it is coprime to p). We show the following.

Theorem 2 (Optimal Bucket for 2-Multipliers)**.** *If p is an odd prime such that*

$$\left(\frac{-1}{p}\right) = -\left(\frac{2}{p}\right) = 1 \ ,$$

then Property 2 holds for $M = \{1, 2\}$ and B as described above.

Remark. $p = 5$ is the first such prime; the requirement of the theorem is equivalent to $p \equiv 5 \pmod 8$.

Remark. Note that $\beta \in \mathbb{Z}$ coprime to p is a quadratic residue mod p^c if and only if it is a quadratic residue mod p. This follows from Hensel's lemma, as any root of the equation $x^2 \equiv \beta \pmod p$ is simple, hence lifts to a unique root mod p^c.

Proof. We divide the proof into several cases. We first deal with $0 < t < q$ coprime to p.

[4] Since precomputed points are of the form mq^jP_i, for $m \in M$, $0 \leqslant j \leqslant h - 1$ and $1 \leqslant i \leqslant n$, it is not important to require that q be a power of 2.

$t < q/2$ **and** $(\frac{t}{p}) = 1$**:** $t \in B$, so there's nothing to show.

$t < q/2$ **even and** $(\frac{t}{p}) = -1$**:** $t/2 = b \in B$, so $t = 2b$.

$t < q/2$ **odd and** $(\frac{t}{p}) = -1$**:** $q-t$ is even and $(\frac{q-t}{p}) = -1$. Therefore, $(q-t)/2 = b \in B$ and $q - t = 2b$.

$t > q/2$ **even and** $(\frac{t}{p}) = 1$**:** in this case $q - t \in B$.

$t > q/2$ **even and** $(\frac{t}{p}) = -1$**:** here, $t/2 = b \in B$, so $t = 2b$.

$t > q/2$ **odd:** $q - t$ is even and either $q - t \in B$ or $(q - t)/2 \in B$.

Now to the general case (we suppose $t \neq 0, q$), when $t = p^k\tau$ $(0 \leqslant k < c)$, where $p \nmid \tau$. The condition

$$t = mb \quad \text{or} \quad q - t = mb$$

is equivalent to

$$p^k\tau = mb \quad \text{or} \quad p^c - p^k\tau = mb \ .$$

Choosing $b < q/2$ of the form $p^k\beta$, with $p \nmid \beta$ and $\beta < p^{c-k}/2$ quadratic residue mod p^c (equivalently, mod p^{c-k}), the previous equation reads

$$\tau = m\beta \quad \text{or} \quad p^{c-k} - \tau = m\beta \ .$$

By our initial work, this condition is satisfied for some $\beta \in B$ when $p \equiv 5 \pmod 8$. □

We now show $|B| = q/4 + O(1)$.

Theorem 3. *The bucket set B in this section has cardinality*

$$|B| = \frac{q-1}{4} + 1 \ .$$

Proof. We begin with a lemma.

Lemma 3. *Let $p \equiv 1 \pmod 4$ be prime and $k \in \mathbb{N}$. The number of quadratic residues mod p^k less than $p^k/2$ is $p^{k-1}(p-1)/4$.*

Proof. The number of quadratic residues mod p^k is $p^{k-1}(p-1)/2$, since they form a cyclic group of index 2 inside the group of invertible classes mod p^k, of order $\varphi(p^k) = p^{k-1}(p-1)$. Also, the subset of those quadratic residues $< p^k/2$ is in bijection with its complement, via the map $t \mapsto p^k - t$, using the fact that -1 is a quadratic residue mod p^k. This leads to the result. □

Returning to the proof of the theorem, we partition B as

$$B = \{0\} \bigcup \bigcup_{k=0}^{c-1} \{t = p^k\beta \colon 0 < \beta < p^{c-k}/2 \text{ is a quadratic residue mod } p^{c-k}\} \ .$$

Taking cardinalities, and using the previous lemma, we find

$$|B| = 1 + \sum_{k=0}^{c-1} \frac{p^{c-k-1}(p-1)}{4} = 1 + \frac{p-1}{4} \cdot \frac{p^c-1}{p-1} = 1 + \frac{p^c-1}{4} \ .$$

□

Remark. Since b is a quadratic residue mod p^c if and only if it is a quadratic residue mod p, we deduce that elements of B are never more than p integers apart, generalizing Proposition 4 to this context (where we can take $p = 5$).

4.2 A General Construction of Optimal Buckets

Here we show the way to modify Property 1 (or 2) in order to obtain a scalar multiplication algorithm with a runtime of essentially $n(h+1) + \frac{q}{2|M|}$ point operations, by precomputing $|M|n(h+1)$ points.

We propose the following modification. As usual, we let $q = p^c$, where p is prime and $c \in \mathbb{N}$.

Property 3. *For all $t \in \mathbb{Z}$, there exist $\epsilon_t \in \{\pm 1\},\ m_t \in M,\ b_t \in B$, such that*

$$t \equiv \epsilon_t m_t b_t \pmod{q} \ .$$

We now let $B \subseteq [0, q-1]$ be a bucket set such that $0 \in B$ and

$$M = \{1, 2, \ldots, |M|\} \ . \tag{11}$$

Assuming Property 3 holds in this case, we rewrite [11, Algorithm 6] to accommodate a recoding without *a priori* restricting α_j:

$$a_j = \epsilon_j m_j b_j + \alpha_j q \ , \quad \epsilon_j \in \{\pm 1\},\ m_j \in M,\ b_j \in B,\ \alpha_j \in \mathbb{Z} \ . \tag{12}$$

The result is the following new scalar recoding Algorithm 6.

Algorithm 6 New scalar recoding

```
Input: $\{a_j\}_{0 \leqslant j \leqslant h-1}$, $0 \leqslant a_j < q$ such that $a = \sum_{j=0}^{h-1} a_j q^j$.
Output: $\{(\epsilon_j m_j, b_j)\}_{0 \leqslant j \leqslant h}$, $\epsilon_j \in \{\pm 1\}$, $m_j \in M$, $b_j \in B$ such that $a = \sum_{j=0}^{h} \epsilon_j m_j b_j q^j$.
1: $a_h \leftarrow 0$
2: for $j = 0$ to $h-1$ do
3:     Obtain $\epsilon_j, m_j, b_j, \alpha_j$ as in (12) such that $a_j = \epsilon_j m_j b_j + \alpha_j q$      ▷ $|\alpha_j| \leqslant |M|$
4:     $a_{j+1} = \alpha_j + a_{j+1}$                                        ▷ Now $|a_{j+1}| < q + |M|$
5: Obtain $\epsilon_h, m_h, b_h$ such that $a_h = \epsilon_h m_h b_h$          ▷ $|a_h| \leqslant |M|, b_h = 0, 1, \alpha_h = 0$
6: return $\{(\epsilon_j m_j, b_j)\}_{0 \leqslant j \leqslant h}$
```

We need to show that Algorithm 6 terminates after Line 5. This is done by addressing the statements found in the comments.

Proposition 5. *In Algorithm 6, we have, for $-1 \leqslant j \leqslant h-1$ (where we define $\alpha_{-1} = 0$), after Line 4,*

$$\begin{cases} |\alpha_j| \leqslant |M| \ , \\ |a_{j+1}| < q + |M| \ , \\ |a_h| \leqslant |M| \ . \end{cases}$$

Proof. The first two statements are proved together by induction on $j \geqslant -1$. The base step is clear, since $\alpha_{-1} = 0$ and a_0 is not modified in Line 4. Supposing $|\alpha_j| \leqslant |M|$ and $|a_{j+1}| < q + |M|$, from Line 3 we deduce

$$|\alpha_{j+1}| \leqslant \frac{m_{j+1}b_{j+1}}{q} + \frac{|a_{j+1}|}{q} \leqslant |M|\left(1 - \frac{1}{q}\right) + 1 - \frac{1}{q} + \frac{|M|}{q} = |M| + 1 - \frac{1}{q} \ ,$$

and therefore, since α_{j+1} is an integer, $|\alpha_{j+1}| \leqslant |M|$. In addition, in Line 4, the new value of a_{j+2} is $a_{j+2} + \alpha_{j+1}$, therefore we can bound the updated value as

$$|a_{j+2}| < |M| + q \ .$$

This completes the inductive step. Finally, note that, as the initial value $a_h = 0$ is updated in Line 4 to $a_h + \alpha_{h-1} = \alpha_{h-1}$, we have the stricter bound $|a_h| = |\alpha_{h-1}| \leqslant |M|$. □

The new scalar recoding allows us to run Pippenger's algorithm as before [11, Algorithms 4 and 3] with at most

$$\big(n(h+1) + |B| + d - 4\big)$$

curve additions – where d is the maximal distance between consecutive elements of B – and the help of

$$n(h+1)|M|$$

precomputed points. The main advantage of the recoding given by Algorithm 6 is that it allows us to use a bucket set B of optimal size $\frac{q}{2|M|} + O(1)$.

Theorem 4. *Let μ be a positive integer, $p > 2$ be prime with $p \equiv 1 \pmod{2\mu}$. Assume $\{\pm 1, \cdots, \pm\mu\}$ form a complete set of representatives of $(\mathbb{Z}/p)^*$ modulo 2μ-th powers. Then, for any $c \in \mathbb{N}$, Property 3 holds for $q = p^c$, the multiplier set $M = \{1, 2, \ldots, \mu\}$ and the bucket set*

$$B = \{0\} \bigcup_{k=0}^{c-1} \left\{0 < b < q \colon b = p^k\beta, \text{ where } 0 < \beta < p^{c-k} \text{ is a } 2\mu\text{-th power modulo } p^{c-k}\right\} .$$

Moreover, the maximal distance between consecutive integers in B is p and

$$|B| = \frac{q}{2\mu} + O(1) = \frac{q}{2\,|M|} + O(1) \ .$$

Proof. We claim that, for any $\kappa \in \mathbb{N}$, the set $S = \{\pm 1, \cdots, \pm\mu\}$ constitutes a complete set of representatives of $(\mathbb{Z}/p^\kappa)^\times$ (the invertible classes modulo p^κ) modulo 2μ-th powers. Indeed, since $2\mu \mid p^{\kappa-1}(p-1)$, knowing that $(\mathbb{Z}/p^\kappa)^\times$ is cyclic, the group

$$(\mathbb{Z}/p^\kappa)^\times / \big((\mathbb{Z}/p^\kappa)^\times\big)^{2\mu}$$

has order 2μ. Moreover, for $r, s \in S$, by Hensel's lemma, the equation

$$r \equiv sx^{2\mu} \pmod{p^\kappa} \text{ is solvable in } \mathbb{Z} \iff r \equiv sx^{2\mu} \pmod{p} \text{ is solvable in } \mathbb{Z} \ .$$

By assumption, this shows that if $r \neq s$, they represent different classes and thus proving our claim. In other words, we have a partition

$$(\mathbb{Z}/p^\kappa)^\times = \bigcup_{1 \leqslant m \leqslant \mu} \left(m \left((\mathbb{Z}/p^\kappa)^\times \right)^{2\mu} \bigcup -m \left((\mathbb{Z}/p^\kappa)^\times \right)^{2\mu} \right) \ .$$

Let $t \in \mathbb{Z}$. As seen in the proof of Theorem 2, write $t = p^k \tau$ where $p \nmid \tau$. If $k \geqslant c$, then $t \equiv 0 \equiv 1 \cdot 1 \cdot 0 \pmod{q}$. Otherwise, let $\kappa = c - k \in \mathbb{N}$. From our claim, solving in β (a 2μ-th power modulo p^κ) the equation

$$\tau \equiv \epsilon m \beta \pmod{p^\kappa}$$

for some $1 \leqslant m \leqslant \mu$ and $\epsilon \in \{\pm 1\}$ will yield, for $b = p^k \beta \in B$, an expression

$$t \equiv p^k \tau \equiv \epsilon m p^k \beta \equiv \epsilon m b \pmod{q} \ ,$$

thus showing Property 3.

To count the elements of B, as in Theorem 3 note that B is already defined as a disjoint union. Therefore

$$|B| = 1 + \sum_{\kappa=1}^{c} \frac{p^{\kappa-1}(p-1)}{2\mu} = 1 + \frac{p^c - 1}{p-1} \cdot \frac{p-1}{2\mu} = 1 + \frac{p^c - 1}{2\mu} = \frac{q}{2\mu} + O(1) \ .$$

Finally,

$$\left((\mathbb{Z}/p^c)^\times \right)^{2\mu} \subseteq B \ ,$$

where on the left we consider representatives in $[1, q-1]$, and we have seen via Hensel's lemma that the condition that b be a 2μ-th power mod q is equivalent to b being a 2μ-th power mod p. This proves the claim on the maximal distance of elements of B. □

Remark. A simple cardinality argument similar to Theorem 1 shows that any bucket set B satisfying Property 3 is such that $|B| \geqslant q/(2\,|M|)$. Therefore Theorem 4 is optimal.

We want to provide a criterion for finding primes p satisfying the hypotheses of Theorem 4.

Proposition 6. *Let $\mu \in \mathbb{N}$ and suppose that $p = 2\mu + 1$ is prime. Then the set $\{\pm 1, \cdots, \pm \mu\}$ form a complete set of representatives of $(\mathbb{Z}/p)^\times / \left((\mathbb{Z}/p)^\times \right)^{2\mu}$.*

Proof. We have, by Fermat's little theorem,

$$\left((\mathbb{Z}/p)^\times \right)^{2\mu} = \{1\} \ ,$$

and $\mu = \frac{p-1}{2}$, so

$$\{\pm 1, \cdots, \pm \mu\} = \left\{ \pm 1, \cdots, \pm \frac{p-1}{2} \right\} = (\mathbb{Z}/p)^\times \ .$$

□

Remark. The first few values of μ, namely $1, 2, 3, 5, 6, 8, 9, 11$, provide via Proposition 6 optimal bucket sets of cardinality $q/(2\mu) + O(1)$ in Theorem 4. Table 3 in the appendix lists our new bucket set constructions for $q = 7^c, 4 \leqslant c \leqslant 11$.

We will now show that, on a $j = 0$ elliptic curve (with equation $y^2 = x^3 + b$), our new property ideally allows to divide the storage requirement by 3 when using units of the endomorphism ring. For instance, with $n(h+1)$ stored points (similar to Pippenger's variant), one can execute a variant of Pippenger's algorithm in essentially $n(h+1) + q/5$ point operations. This is not yet optimal (we would like to reach $nh + q/6$), but constitutes an improvement over the Pippenger variant runtime of $nh + q/2$.

5 Combining Efficient Endomorphisms with Optimal Buckets for Efficient MSM Computation

Many families of pairing-friendly curves over $\mathbb{F}_p$ have j-invariant equal to zero. They have an equation $y^2 = x^3 + b$ for some $b \in \mathbb{F}_p$. Therefore, they are endowed with an endomorphism ω such that $\omega^3(P) = P$ for all P on the elliptic curve. We can write $\omega(x, y) = (\zeta_3 x, y)$, where $\zeta_3 \in \mathbb{F}_p$ such that $\zeta_3^3 = 1$. The computation of ω can therefore be done on the fly, and corresponding points do not need to be stored, which is now what we want to take advantage of.

The endomorphism ring of these curves is isomorphic to $\mathbb{Z}[\omega]$, where $\omega = \frac{-1+i\sqrt{3}}{2}$ is a complex cube root of unity (using the same letter as for the fast endomorphism). We will modify Property 3 to allow ϵ_t to be an endomorphism unit in $\mathbb{Z}[\omega]$, resulting in the following property:

Property 4. *For all $t \in \mathbb{Z}[\omega]$, there exist $\epsilon_t \in U = \{\pm 1, \pm\omega, \pm\omega^2\}$, $m_t \in M$, $b_t \in B$, such that*

$$t \equiv \epsilon_t m_t b_t \pmod{q} \ .$$

Let's consider the first implementation of this idea. Let $M = \{1\}$, $q = p^c$ with $c \geqslant 2$, where $p = 2 - \omega$. Note that $p \mid 7$ in $\mathbb{Z}[\omega]$. Also, $\mathbb{Z}[\omega]$ is a (norm-)Euclidean ring, and hence any $t \in \mathbb{Z}[\omega]$ has a representative in $\mathbb{Z}[\omega]/q$ of modulus less than $|q| = 7^{c/2}$. Finally, $|\mathbb{Z}[\omega]/q| = 7^c$. We now show that any $t \in \mathbb{Z}[\omega]$ has a base q expansion of length bounded by $h + 1 = \lceil \frac{\log|t|}{\log|q|} \rceil + 1$, where each q-digit of bounded by $|q|/\sqrt{3} + 1$.

Lemma 4. *Let $\alpha, \beta \in \mathbb{Z}[\omega]$ and $\beta \neq 0$. There exist $\delta, \rho \in \mathbb{Z}[\omega]$ such that*

$$\alpha = \beta\delta + \rho \ , \qquad (13)$$

where $|\rho| \leqslant |\beta|/\sqrt{3}$. Additionally, for $\beta = q$, given any base-q expansion of $a \in \mathbb{Z}[\omega]$

$$a = a_0 + a_1 q + \cdots + a_m q^m \ , \qquad (14)$$

with $|a_k| \leqslant |q|/\sqrt{3} + 1$ for $0 \leqslant k \leqslant h - 1$, there exists an equivalent base-q expansion

$$a = a'_0 + a'_1 q + \cdots + a'_h q^h \ ,$$

with $a'_k = a_k$ $0 \leqslant k \leqslant h-1$ *and* $|a'_h| < 2$*, having length* $h + 1 = \lceil \frac{\log|a|}{\log|q|} \rceil + 1$*.*

Proof. Rewrite (13) as

$$\tau = \delta + \varepsilon \ , \quad \delta \in \mathbb{Z}[\omega] \ , \quad |\varepsilon| \leqslant \frac{1}{\sqrt{3}} \ ,$$

where $\tau = \alpha/\beta$ and $\varepsilon = \rho/\beta$. Since τ belongs to an equilateral triangle of side length one with vertices in $\mathbb{Z}[\omega]$, we can select δ to be the closest vertex to τ, which will therefore be at distance at most $1/\sqrt{3}$ (if τ is located at the center of the equilateral triangle). An implementation of this choice is described in Algorithm 7.

Regarding the length of the expansion, let $\delta_0 = a$ and, for $0 \leqslant n \leqslant h-1$,

$$\delta_n = q\delta_{n+1} + a_n$$

with a_n selected in some way (i.e. Euclidean division) as in (14) and $|a_n| \leqslant |q|/\sqrt{3} + 1$. In particular,

$$|\delta_{n+1}| \leqslant \frac{|\delta_n|}{|q|} + \frac{1}{\sqrt{3}} + \frac{1}{|q|} \ ,$$

so that, by induction,

$$\begin{aligned} |\delta_k| &\leqslant \frac{|a|}{|q|^k} + \left(1 + \frac{1}{|q|} + \cdots + \frac{1}{|q|^{k-1}}\right)\frac{1}{\sqrt{3}} + \left(\frac{1}{|q|} + \cdots + \frac{1}{|q|^k}\right) \\ &< \frac{|a|}{|q|^k} + \frac{|q|}{(|q|-1)\sqrt{3}} + \frac{1}{|q|-1} \ . \end{aligned} \tag{15}$$

Since $|a|/|q|^h \leqslant 1$ and $|q| \geqslant 5$, we deduce $|\delta_h| < 2 < |q|$. If $\delta_h \neq 0$, an integer division gives

$$\delta_h = q \times 0 + a_h$$

with $|a_h| < 2$ and $\delta_{h+1} = 0$.

□

Let

$$\Delta = \left\{ z \in \mathbb{C} \colon |z| \leqslant \frac{|q|}{\sqrt{3}} \right\} \ .$$

We have just shown that every class in $\mathbb{Z}[\omega]/q$ has a representative in Δ. The next lemma shows that the same can be said about $\Delta_+ = \Delta + 1$, the translate of Δ by one unit to the right.

Lemma 5. *Every class in* $\mathbb{Z}[\omega]/q$ *has a representative in* Δ_+*.*

Proof. Let $r = |q|/\sqrt{3}$. Let $z \in \Delta \backslash \Delta_+$. Then $|z - 1| \leqslant r + 1$. Trace a circle centered at z with radius $r\sqrt{3}$. It will intersect the boundary of Δ_+ at points A, B. Call $\theta = \widehat{AzB}$ (see Fig. 1). We claim that $\theta > \pi/3$ when $r \geqslant 4$. To see this,

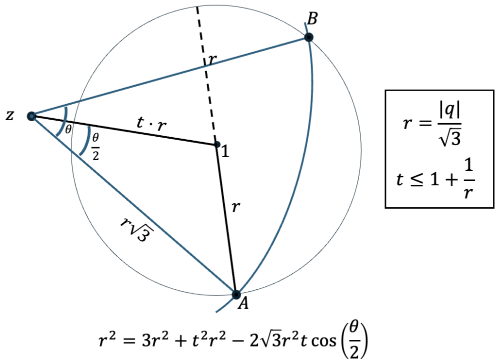


Fig. 1. Circle from Lemma 5

consider the triangle with vertices $1, z, A$. Let $t = |z-1|/r \leqslant 1 + 1/r$. By the law of cosines we have

$$r^2 = 3r^2 + r^2t^2 - 2\sqrt{3}\,r^2t\cos(\theta/2) \iff \cos(\theta/2) = \frac{1}{\sqrt{3}}\left(\frac{1}{t} + \frac{t}{2}\right) < \frac{\sqrt{3}}{2} \ ,$$

since the function $f(t) = t^{-1} + t/2$ is decreasing in $[1, \sqrt{2}]$ and when $r \geqslant 4$ we have $1 < t < \sqrt{2}$.

Since $\theta > \pi/3$, there exists $u \in U$ such that $z + uq$ lies on the circular arc $\widehat{AB}$, and therefore in Δ_+. □

The next theorem is the analog of Theorem 4 with $\mu = 3$, with one important difference that we will bring out in the remark after its proof.

Theorem 5. *We have*

$$(\mathbb{Z}[\omega]/p)^\times = \left\{\pm1, \pm\omega, \pm\omega^2\right\} \ .$$

For any $c \in \mathbb{N}$, Property 4 holds for $q = p^c$, the multiplier set $M = \{1\}$ and the bucket set

$$B = \{0\} \cup \bigcup_{k=0}^{c-1} B_k \ ,$$

where

$$B_k = \left\{b \in \Delta_+ \cap \mathbb{Z}[\omega] \colon b = p^k\beta, \ \textit{where}\ \beta \equiv 1 \pmod{p}\right\} \ .$$

Moreover,

$$|B| = \frac{7^c\pi}{9\sqrt{3}} + o(7^c) \approx \frac{7^c}{5} \ .$$

Proof. Note initially that,

$$(\mathbb{Z}[\omega]/p)^\times = \left\{\pm1, \pm\omega, \pm\omega^2\right\} \ .$$

Indeed, by the considerations after the statement of Property 4, the cardinalities of the left- and right-hand sides of the previous equality match. Moreover, any two distinct elements of $\{\pm 1, \pm\omega, \pm\omega^2\}$ are not congruent modulo p, because their difference has algebraic norm either 1, 3 or 4, coprime to 7. Consider a base-q expansion (14). A digit a_n can be written uniquely as

$$a_n = p^k \alpha \quad \text{where } k \geqslant 0,\, \alpha \in \mathbb{Z}[\omega] \text{ and } p \nmid \alpha \ .$$

Therefore, there exists a unique $\epsilon \in U$ such that $\alpha \equiv \epsilon \pmod{p}$. Posing $\beta' = \alpha\epsilon^{-1}$, we have $\beta' \equiv 1 \pmod{p}$ and

$$a_n = \epsilon p^k \beta' = \epsilon b \quad \text{where } b \in B_k \ ,$$

where we have used Lemma 5 to justify that $p^k\beta' \equiv p^k\beta \pmod{q}$ with $b = p^k\beta \in B_k$. Finally, note that, as in Theorem 4, the sets B_k together with $\{0\}$ partition B. We have

$$B_k = \left\{ z \in \mathbb{Z}[\omega] \colon p^k(1+pz) \in \mathit{\Delta} + 1 \right\}$$

is in bijection with

$$B'_k = \frac{B_k}{p^{k+1}} = \left\{ z \in \mathbb{Z}[\omega] \colon z \in \frac{\mathit{\Delta}}{p^{k+1}} + \frac{1}{p^{k+1}} - \frac{1}{p} \right\} \tag{16}$$

Precisely estimating the cardinality of lattice points inside a disk is a well-known and difficult problem, related to Dirichlet's divisor problem and the ideal theorem. For our purposes, suffice it to say that this number is asymptotic to the area of the disk divided by the area of a fundamental parallelogram of the lattice (see [10, VI.3 Theorem 3 p.132]). Hence

$$|B_k| = \frac{\pi 7^{c-k-1}/3}{\sqrt{3}/2} + o\big(7^{c-k-1}\big) = \frac{2\pi 7^{c-k-1}}{3\sqrt{3}} + o\big(7^{c-k-1}\big) \ .$$

Finally, a calculation similar to the proof of Theorem 4 shows that (using $f(x) \sim g(x)$ to denote $\lim_{x\to\infty} f(x)/g(x) = 1$)

$$|B| \sim \sum_{\kappa=0}^{c-1} \frac{2\pi 7^\kappa}{3\sqrt{3}} = \frac{7^c\pi}{9\sqrt{3}} + o(7^c) \ .$$

□

Remark. The actual magnitude of the error term is still unknown. It is conjectured that it is $O(7^{c/4+\epsilon})$ (best possible), but current results are for an exponent a bit lower than $O(7^{c/3})$. We would have liked to get a clean estimate of $\sim 7^c/6$, as in Theorem 4, but that would have led us to an irregularly shaped set B, where the existence of a Hamiltonian path would prove too daunting. Instead, we opted to allow some redundancy in B (about 20%) to pay for the benefit of working in a symmetric set.

Algorithm 7 Euclidean division

```
Input: $\kappa = k_1 + k_2\omega$, $q = q_1 + q_2\omega$ with $k_1, k_2, q_1, q_2 \in \mathbb{Z}$.
Output: CEA$(\kappa, q) = (\delta, \rho)$, where $\delta, \rho \in \mathbb{Z}[\omega]$ with $\kappa = q\delta + \rho$ and $|\rho|^2 \leqslant |q|^2/3$.
$U + V\omega \leftarrow \kappa/q$
$\delta \leftarrow \lfloor U \rceil + \lfloor V \rceil \omega$
$N \leftarrow |U + V\omega - \delta|^2$
$\delta' \leftarrow \delta + 1$
$N' \leftarrow |U + V\omega - \delta'|^2$
if $N' < N$ then
    $\delta \leftarrow \delta'$
    $N \leftarrow N'$
$\delta' \leftarrow \delta' + \omega$
$N' \leftarrow |U + V\omega - \delta'|^2$
if $N' < N$ then
    $\delta \leftarrow \delta'$
    $N \leftarrow N'$
$\delta' \leftarrow \delta' - 1$
$N' \leftarrow |U + V\omega - \delta'|^2$
if $N' < N$ then
    $\delta \leftarrow \delta'$
return $\delta$, $\rho = \kappa - q\delta$
```

Algorithm 8 now replaces Algorithm 6 in computing a recoding amenable to the bucket algorithm, without any additional precomputation than the Pippenger variant.

Theorem 5 can be seen as a version of Theorem 4 when $\mu = 3$, with nh precomputed points instead of $3nh$. However, there is one fundamental difference, and we show how to deal with it.

Elliptic curve computations take place in a cyclic group $\mathbb{G}$ of prime order N. The parameter h is then defined – when the prime p is chosen in $\mathbb{Z}$ (for instance $p = 7$ when $\mu = 3$) – as the exponent of the smallest power of q exceeding N, in other terms,

$$h = \left\lceil \frac{\log N}{\log q} \right\rceil \ .$$

This is because we must be able to represent any scalar multiplier ($\leqslant N$) in a base q expansion of length at most h. If we now let $p = 2 - \omega$, and let as before $q = p^c$, then, since $|q| = 7^{c/2}$, the corresponding h would double, necessitating twice as many precomputed points ($2nh$, instead of nh). Although this is below the $3nh$ provided by our refinement of the LFG method, we can do better.

Note that since N is large, there is no other copy isomorphic to $\mathbb{G}$ in the elliptic curve (over the field of definition of $\mathbb{G}$). Consequently, the endomorphism ω must act as an isomorphism of $\mathbb{G}$. Therefore, given a point $P \in \mathbb{G}$, $\omega P = \lambda P$ for some $\lambda \in \mathbb{Z}/N$ with $\lambda^3 \equiv 1 \pmod{N}$. This implies that $N \equiv 1 \pmod{3}$ splits in $\mathbb{Z}[\omega]$, so $N = \nu_1\nu_2$, with ν_1, ν_2 primes in $\mathbb{Z}[\omega]$. Since, denoting $\mathbf{0}$ the point at infinity,

$$\mathbf{0} = NP = \nu_1\nu_2 P \ ,$$

Algorithm 8 Complex scalar recoding

```
Input: $a \in \mathbb{Z}[\omega]$, $q = p^c = (2-\omega)^c$ with $c \in \mathbb{N}$.          $\rhd\ h = \lceil \frac{\log|a|}{\log|q|} \rceil$
Output: $\{(\epsilon_j, b_j)\}_{0 \leqslant j \leqslant h}$, $\epsilon_j \in U$, $b_j \in B$ such that $a = \sum_{j=0}^{h} \epsilon_j b_j q^j$.
$\kappa \leftarrow a$
$h \leftarrow \lceil \frac{\log|a|}{\log|q|} \rceil$
List $\leftarrow \{\}$
for $j = 0$ to $h$ do
    $(\rho, \delta) \leftarrow \mathrm{CEA}(\kappa, q)$
    if $\rho = 0$ then
        $\epsilon_j = 1$, $b_j = 0$
    else
        $r \leftarrow \rho$
        $k \leftarrow 0$
        while $p \mid r$ do
            $r \leftarrow r/p$
            $k \leftarrow k + 1$
        Define $\epsilon_j \in U$ so that $\epsilon_j \equiv r \pmod{p}$
        $b_j = \rho\epsilon_j^{-1}$                    $\rhd\ b_j \equiv 1 \pmod{p}$ and $|b_j| \leqslant |p|^{c-k}/\sqrt{3}$
        $b \leftarrow b_j$
        while $|p^k b - 1|^2 > 7^c/3$ do
            $b \leftarrow b_j + p^{c-k}u$ , $(u \in U)$          $\rhd$ modify $b_j$ to make $p^k b_j \in \Delta_+$
        $b_j \leftarrow b$
    List $\leftarrow$ List.append$((\epsilon_j, b_j))$
    $\kappa \leftarrow \delta$
return: List
```

we deduce that either $\nu_1 P = \mathbf{0}$ or $\nu_2 P = \mathbf{0}$. Let ν represent the corresponding prime. Then, $|\nu| = \sqrt{N}$ and, if $\rho \equiv a \pmod{\nu}$, then $aP = \rho P$. Using Lemma 4, we can ensure that $|\rho| \leqslant |\nu|/\sqrt{3}$. The bottom line is that we can represent any scalar $a \leqslant N$, having an expansion of length h in base 7^c, by an equivalent expansion (of ρ) in base $q = (2-\omega)^c$ of the *same length* h, or $h+1$. In particular, it is sufficient to precompute $h+1$ powers of q.

Remark. In almost all cases, one can actually show that h powers suffice, and the recoding will have exactly the same length h. This can be done as follows: in (15), use the following upper bound

$$|\delta_h| \leqslant \frac{|a|}{|q|^h} + \frac{|q|}{(|q|-1)\sqrt{3}} + \frac{1}{|q|-1} \leqslant \frac{\sqrt{N}}{|q|^h\sqrt{3}} + \frac{|q|}{(|q|-1)\sqrt{3}} + \frac{1}{|q|-1} \ .$$

Since h may in some cases be bigger than $\log\sqrt{N}/\log|q|$ by as much as 1, and $|q|$ gets large, this will result in a bound $|\delta_h| < 1$, which will force $\delta_h = 0$ and the expansion to be one digit shorter.

Theorem 6. *In Theorem 5, it is possible to label the elements of B as $B = \{b_0 = 0, b_1 = 1, \ldots b_{|B|-1}\} \subseteq \mathbb{Z}[\omega]$, in such a way that*

$$|b_{k+1} - b_k|^2 \leqslant 7 \ , \quad \textit{for all } 1 \leqslant k \leqslant |B| - 2 \ .$$

Moreover, if $|b_{k+1} - b_k|^2 = 7$, then $b_{k+1} - b_k = \epsilon p$, where $\epsilon \in U = \{\pm 1, \pm\omega, \pm\omega^2\}$.

Proof. We first focus on the subset $B_0 \subseteq B$, as defined in Theorem 5. Using (16), we describe a Hamiltonian path with edges of length 1 in B_0'. Note that

$$B_0' = \left\{ z \in \mathbb{Z}[\omega] \colon |z| \leqslant r = |p|^{c-1}/\sqrt{3} \right\} \quad .$$

In particular, B_0' is symmetric about the real and imaginary axes (hence about the origin), as well under rotations by $\pi/3$ centered at the origin. We introduce some additional notation. The set B_0' is a union of horizontal intervals

$$\mathcal{L}(a, b) = \{z \in \mathbb{Z}[\omega] \colon -a \leqslant \Re z \leqslant a \text{ and } \Im z = \Im b\}$$

called *layers*. The left endpoint of $\mathcal{L}(a, b)$ is $-a + i\Im b$, its right endpoint $a + i\Im b$. A *neighbor* of $z \in \mathbb{Z}[\omega]$ is one of the points $z + \epsilon$ where $\epsilon \in U$. For $S \subseteq \mathbb{Z}[\omega]$ symmetric about both coordinate axes and under rotations by $\pi/3$ centered at the origin, a point on the *boundary* ∂S of S is a point of S which doesn't have all its neighbors in S.

Ziggurat property: For S, possibly empty as above and symmetric about both coordinate axes and under rotations by $\pi/3$ centered at the origin, we say S satisfies the ziggurat property if

(ZP1) for any two layers $\mathcal{L}(a, b), \mathcal{L}(c, d)$ with $0 \leqslant \Im b < \Im d$ we have $c \leqslant a + 1/2$,

(ZP2) if $\mathcal{L}(a, \omega)$ is the first layer above the layer on the real axis $\mathcal{L}(x, 0)$, then its right endpoint $a + i\sqrt{3}/2$ is a neighbor of the right endpoint x of $\mathcal{L}(x, 0)$.

A simplified visual representation of the ziggurat satisfying ZP1 is shown on the left side of Fig. 5, while the right side depicts the ziggurat after the points along the boundary ∂S have been counted and subsequently removed.

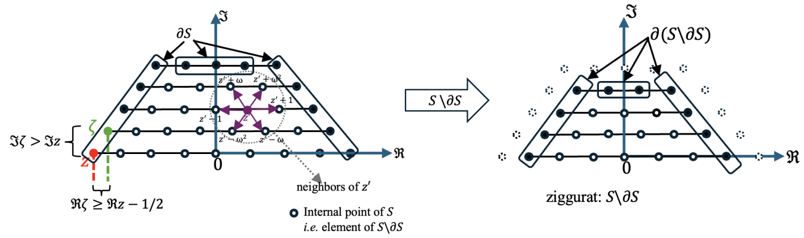


Fig. 2. Simplified representation of a ziggurat S

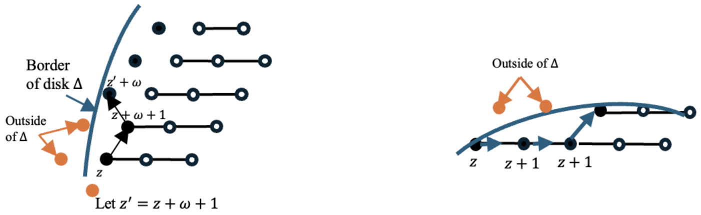
Informally, this means that the layers above the real axis don't jut out from layers below them. The following result is at the heart of our construction.

Proposition 7. *If S satisfies the ziggurat property, then ∂S has a Hamiltonian circuit with edges of length one. Additionally, $S\backslash\partial S$ also satisfies the ziggurat property.*

Proof. Note that the definition of ∂S implies it's symmetric with respect to both coordinate axes as well as under rotations by $\pi/3$ about the origin, hence the same is true for $S\backslash\partial S$. We will describe how to choose the vertices $z_1, \ldots, z_m$ of the Hamiltonian circuit ∂S. By ZP2, $a = x \pm 1/2$. If $a = x - 1/2$, define $z_1 = x$, otherwise, let $z_1 = x + \omega + 1$, the right endpoint of $\mathcal{L}(a, \omega)$. Then move counterclockwise. At each step n, until reaching the top layer,

1. if possible, jump up to the right endpoint of the next layer: $z_{n+1} = z_n + \omega$ or $z_{n+1} = z_n + \omega + 1$, otherwise
2. move left on the layer: $z_{n+1} = z_n - 1$.

Fig. 5 illustrates the various scenarios that arise when counting points along the border. After reaching the top layer, when $\Re z_n$ becomes negative, proceed symmetrically down until reaching $-z_1$. Then use symmetry about the origin to complete the circuit. A moment of thought will convince the reader that we are capturing all points of ∂S in this fashion.



a) Points along a regular border

b) Points along a flat border

Fig. 3. Different types of border

Regarding ZP1 for $S\backslash\partial S$: we argue by contradiction. Let $\mathcal{L}(a, b), \mathcal{L}(c, d)$ be two layers of $S\backslash\partial S$ with $0 \leqslant \Im b < \Im d$ we have $c \geqslant a + 1$. Let e_a (resp. e_c) be the right endpoints of $\mathcal{L}(a, b)$ (resp. $\mathcal{L}(c, d)$). We have $\Re e_a = a$ and $\Re e_c = c$. Any point $z \in \partial(S\backslash\partial S)$ has a neighbor $z^* \notin S\backslash\partial S$. Since $z^* \in S$ (otherwise $z \in \partial S$, impossible), we conclude that $z^* \in \partial S$. In particular, e_a has a neighbor $e_a^* \in \partial S$. By symmetry, one can always choose $\Im e_a^* \geqslant 0$. Also, all neighbors of e_b are in S, in particular, $e_b + 1$ and $e_b + \omega + 1$. But one of these two will have real part $\geqslant \Re e_a^* + 1$ and lie on a layer above e_a^*, thus violating ZP1 for S.

Finally, ZP2 is also true for $S\backslash\partial S$. In fact, the right endpoints of both $\mathcal{L}(a, \omega)$ and $\mathcal{L}(x, 0)$ are the only points in ∂S, as long as they are different from a unit. Indeed, by symmetry, if the Hamiltonian path on ∂S went from z to $z - 1$ on one of these layers, then by symmetry it would have to travel from $(\omega + 1)z$ to $(\omega + 1)(z - 1)$. But $\Im(\omega + 1)z > \Im(\omega + 1)(z - 1)$, contrary to the construction of the Hamiltonian path. This concludes the proof of Proposition 7.

□

Proposition 7 can be repeated by induction. Start from $S_0 = B'_0$ and note that S_0 satisfies the ziggurat property since in a disk, the layers above a diameter naturally satisfy ZP1 and ZP2 (look at the slopes of tangents to the circle at those layers). Then, defining $S_{n+1} = S_n \backslash \partial S_n$, we can recursively construct a Hamiltonian circuit for ∂S_n. Our selection of the starting point for each circuit implies that the last point of ∂S_n is always a neighbor of the first point of ∂S_{n+1}, thus allowing us to construct a path spiraling inwards, as a spider spins its web. By symmetry, the last point of the path will be the origin. An implementation of this idea appears as Algorithm 10, where we have rescaled the set B'_0 back to $B_0 = pB'_0 + 1$.

Regarding the sparser sets B_k for $k \geqslant 1$, each point $z \in \bigcup_{k=1}^{c-1} B_k$ will lie in an equilateral triangle with at least two vertices in B_0 (and at most one such point z per equilateral triangle). By construction, one of the edges of the Hamiltonian path for B_0 will be a side $z_n z_{n+1}$ of this equilateral triangle. It suffices then to modify the path to one including z, for instance $z_n z z_{n+1}$, without increasing the edge length.

Finally, since we want our set to start from $b_0 = 0, b_1 = 1$, we relabel $b_k = z_{|B|-k}$ for $1 \leqslant k \leqslant |B| - 1$. This concludes the proof of Theorem 6.

🕷

Thanks to the previous theorem, using this relabeling on the bucket set B, one can replace Algorithm 1 in the LFG method with Algorithm 9.

Algorithm 9 Subsum accumulation algorithm (with endomorphisms)

```
Input: $B = \{0\} \cup \{b_1, b_2, \ldots, b_{|B|-1}\}$ as in Prop. 6, $S_1, S_2, \ldots, S_{|B|-1}$
Output: $S = b_1 S_1 + \cdots + b_{|B|-1} S_{|B|-1}$
Define a length 25 array $\mathsf{tmp} = [0] \times 25$
for $i = |B| - 1$ to 1 by $-1$ do
    $\mathsf{tmp}[0] = \mathsf{tmp}[0] + S_i$
    $k = b_i - b_{i-1}$
    if $|k|^2 \geqslant 1$ then
        $\mathsf{tmp}[k] = \mathsf{tmp}[k] + \mathsf{tmp}[0]$
return $\sum_{\substack{|k|^2 \leqslant 7 \\ (3+\omega) \nmid k}} k \cdot \mathsf{tmp}[k]$
```

Remark. Since the only primes (up to units) of $\mathbb{Z}[\omega]$ with norm at most 7 are $\omega - 1$ (above 3) and $2 - \omega$, (above 7, excluding the other prime $3 + \omega$), there are at most $d = 24$ nonzero integers $k \in \mathbb{Z}[\omega]$ such that $|k|^2 \leqslant 7$, which explains the first step in Algorithm 9.

Sorting Algorithm in B_0: Here we provide an algorithm (Algorithm 10) to reorder the points in B_0. The idea behind it is to start with the point z_1 closest to the disk Δ and move on to the next point closest to z_1 that lies within the

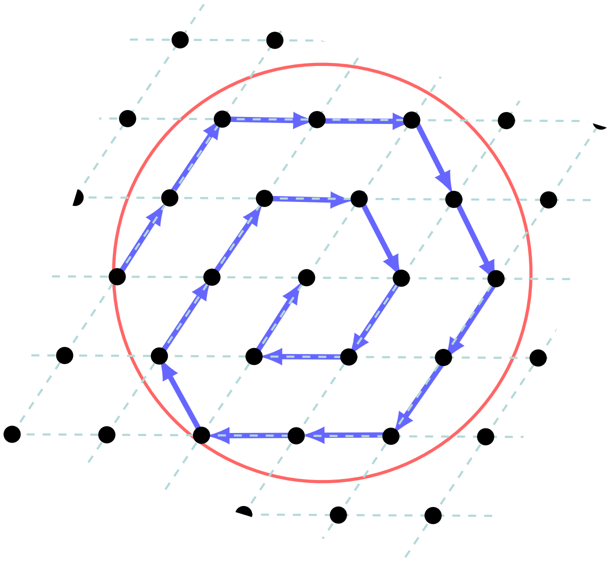


Fig. 4. Hamiltonian path for B_0

disk and whose distance from the border of the disk is minimal. A simplified visualization of the Hamiltonian path is shown in Fig. 4.

6 Performance Analysis and Implementation

In this section, we first provide more details on the optimal bucket construction, and then analyze the performance of the proposed approaches and present our implementation results. We provide the code in the Github repository[5].

To conduct evaluation and compare with LFG methd [11] in a fair way, we choose the BLS12-381 curve and the group order is

$$r = \texttt{0x73eda753299d7d483339d80809a1d80553bda402fffe5bfeffffffff00000001},$$

which determines the upper bound of scalars involved in $S_{n,r}$. BLS12-381 is a pairing-friendly curve with embedded degree 12 and defined by the equation

$$E(\mathbb{F}_p) : y^2 = x^3 + 4,$$

where p is the 381-bit field characteristic. Two additive rational point groups $\mathbb{G}_1 \subset E(\mathbb{F}_p)$ and $\mathbb{G}_2 \subset E(\mathbb{F}_{p^2})$ over which bilinear pairings are defined have the same prime order r.

[5] `https://github.com/xinxin-crypto/MSM_blst_ext`

Algorithm 10 Sorting Algorithm for B_0

```
Input: $S = \Delta \cap p\mathbb{Z}[\omega] = \{pz_1, \ldots, pz_m\}$ unsorted, $r = |q|/\sqrt{3}$
Output: $B_0 = \{b_1, \ldots, b_m\}$, s.t. $b_1 = 1$ and $|b_{j+1} - b_j| \leqslant \sqrt{7}$ for $1 \leqslant j \leqslant m - 1$
for $1 \leqslant k \leqslant m$ do
    if $|z_k| > |z_1|$ then
        Interchange $z_k \leftrightarrow z_1$
    Set $\hat{b}_1 := pz_1$.
for $1 \leqslant i \leqslant m - 1$ do
    Take $pu_\nu \in \{\pm p, \pm\omega p, \pm\omega^2 p\}$, $\nu \in \{1, \ldots, 6\}$.
    Compute $\hat{b}_i + pu_\nu \in S$ with $|\hat{b}_i + pu_\nu| \leqslant r$ and $|\hat{b}_i + pu_\nu| = \max_{\mu \in \{1,\ldots,6\}} |\hat{b}_i + pu_\mu|$
    Set $\hat{b}_{i+1} := \hat{b}_i + pu_\nu$
    $S = S \backslash \{\hat{b}_i\}$
for $1 \leqslant i \leqslant m$ do
    $b_i = \hat{b}_{m+1-i} + 1$
return: $B_0 = \{b_1, \ldots, b_m\}$.
```

6.1 New Bucket Construction for the Repaired LFG Method

The New Bucket Set Constructions for $q = 2^c, 10 \leqslant c \leqslant 31$. Table 2 lists our new bucket set constructions for $q = 2^c, 10 \leqslant c \leqslant 31$, where the maximum difference d between neighbors of $|B|$ is always equal to 6. When compared to the bucket sets constructed in [11], our bucket sets do not rely on a specific elliptic curve. Since the time complexity for computing $S_{n,r}$ is approximately $nh + |B|$, a smaller $|B|$ results in lower time complexity given the same h. Hence, radices 2^c for $c \in \{21, 23, 25, 27, 28, 30, 31\}$ are abandoned in the table.

The New Bucket Set Constructions for $q = 7^c, 4 \leqslant c \leqslant 11$. Table 3 lists our new bucket set constructions for $q = 7^c, 4 \leqslant c \leqslant 11$, where the maximum difference d between neighbors of $|B|$ is always equal to 7.

6.2 Bucket Constructions for the Endomorphism Method

This paper considers bucket constructions for $q = (2 - \omega)^c$. We start with an example with $c = 2$ and then provide general results with more values for c.

Bucket Set Construction Example for $c = 2$. Given $a = a_0 + a_1\omega \in \mathbb{Z}[\omega]$, an element in B_0 can be represented as

$$\begin{aligned} b &= 1 + (2 - \omega)(a_0 + a_1\omega) \\ &= (1 + 2a_0 + a_1) + (3a_1 - a_0)\omega \end{aligned}$$

The norm of b can be computed as

$$\begin{aligned} |b|^2 &= (1 + 2a_0 + a_1)^2 + (3a_1 - a_0)^2 - (1 + 2a_0 + a_1)(3a_1 - a_0) \\ &= 7(a_0^2 + a_1^2 - a_0a_1) + 5a_0 - a_1 + 1 \end{aligned}$$

The concrete construction procedure of B is described as follows:

Table 2. New Bucket Sets Constructions for $q = 2^c, 10 \leqslant c \leqslant 31$.

q	h	$\|B\|$	$\|B\|/q$
2^{10}	26	226	0.22070
2^{11}	24	448	0.21875
2^{12}	22	897	0.21899
2^{13}	20	1791	0.21863
2^{14}	19	3587	0.21893
2^{15}	17	7167	0.21872
2^{16}	16	14340	0.21881
2^{17}	15	28672	0.21875
2^{18}	15	57346	0.21876
2^{19}	14	114686	0.21875
2^{20}	13	229380	0.21875
2^{21}	13	458750	0.21875
2^{22}	12	917508	0.21875
2^{23}	12	1835005	0.21875
2^{24}	11	3670018	0.21875
2^{25}	11	7340030	0.21875
2^{26}	10	14680067	0.21875
2^{27}	10	29360126	0.21875
2^{28}	10	58720261	0.21875
2^{29}	9	117440511	0.21875
2^{30}	9	234881027	0.21875
2^{31}	9	469762045	0.21875

Table 3. New Bucket Sets Constructions for $q = 7^c, 4 \leqslant c \leqslant 11$

q	h	$\|B\|$	$\|B\|/q$
7^4	23	401	0.167
7^5	19	2802	0.167
7^6	16	19609	0.167
7^7	13	137258	0.167
7^8	12	960801	0.167
7^9	11	6725602	0.167
7^{10}	10	47079209	0.167
7^{11}	9	329554458	0.167

1. Searching B_0. For searching B_0, we need to find all elements $\beta = 1 + pz, z \in \mathbb{Z}[\omega]$ with $|pz|^2 \leqslant \lfloor \frac{7^2}{3} \rfloor$. The search results are summarized in Table 4.
2. Searching B_1. For searching B_1, we need to find all elements $p\beta$, where $\beta \equiv 1 \bmod p$ and $|p\beta - 1|^2 \leqslant \lfloor \frac{7^2}{3} \rfloor$. We obtain $B_1 = \{2 - \omega\}$.
3. Obtaining B. To summarize, the bucket set B for $c = 2$ is shown below:

   $$B = \{0, 1, -1 + \omega, -2 - 2\omega, 2 - \omega, 2 + 3\omega, -3\omega, 4 + 2\omega, 3 - \omega\}.$$

   The size of B is 9.
4. Sorting B. For sorting B, we need to construct a Hamiltonian path as shown in Fig. 6.2. After traversing the Hamiltonian path in the reverse order, we

can obtain the sorted bucket set $B = \{0, 1, -2 - 2\omega, -3\omega, 2 - \omega, 3 - \omega, 4 + 2\omega, 2 + 3\omega, -1 + \omega\}$.

Table 4. Searching B_0 when $c = 2$

	(z_0, z_1)	β	$\|\beta\|^2$
1	$(0, 0)$	1	1
2	$(-1, 0)$	$-1 + \omega$	3
3	$(-1, -1)$	$-2 - 2\omega$	4
4	$(0, 1)$	$2 + 3\omega$	7
5	$(0, -1)$	-3ω	9
6	$(1, 1)$	$4 + 2\omega$	12
7	$(1, 0)$	$3 - \omega$	13

General Bucket Construction Results for the Endomorphism Method. Table 5 lists our new bucket set constructions for $q = (2 - \omega)^c, 2 \leqslant c \leqslant 10$. We also visualize the bucket construction for $c = 2$ and $c = 3$ in Fig. 6.2 and Fig. 6.2, respectively.

Table 5. New Bucket Sets Constructions for $q = (2 - \omega)^c, 2 \leqslant c \leqslant 10$

q	$\|q\|^2$	h	$\|B\|$	$\|B\|/7^c$
$(2 - \omega)^2$	7^2	46	9	0.184
$(2 - \omega)^3$	7^3	31	71	0.207
$(2 - \omega)^4$	7^4	23	470	0.196
$(2 - \omega)^5$	7^5	19	3237	0.196
$(2 - \omega)^6$	7^6	16	22565	0.192
$(2 - \omega)^7$	7^7	13	157951	0.192
$(2 - \omega)^8$	7^8	12	1104420	0.192
$(2 - \omega)^9$	7^9	11	7731415	0.192
$(2 - \omega)^{10}$	7^{10}	10	54117148	0.192

6.3 Theoretical Analysis

The storage cost of precomputation results and the computation complexity of computing MSM with precomputation of different algorithms are summarized in Table 6. From the summary given in Table 1, it is easy to see that the choice of h will affect the overall performance. To evaluate the MSM schemes, we first search for the optimal value of h to minimize the computation cost, and then calculate the corresponding storage[6] and computation cost. The repaired LFG

[6] Using compression techniques, we need to store one coordinate and one bit.

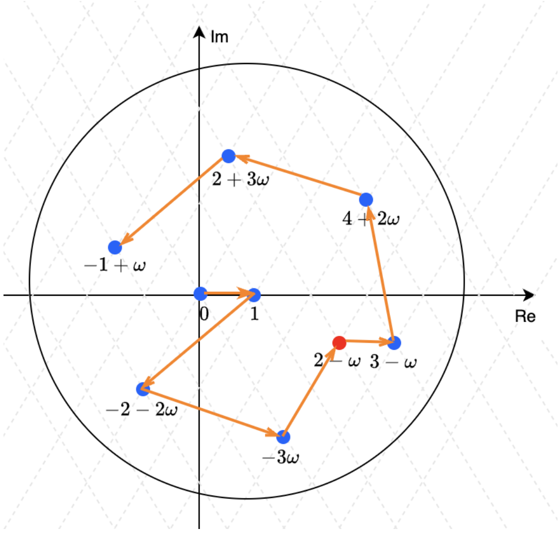


Fig. 5. The construction of the Hamiltonian Path for B when $c = 2$

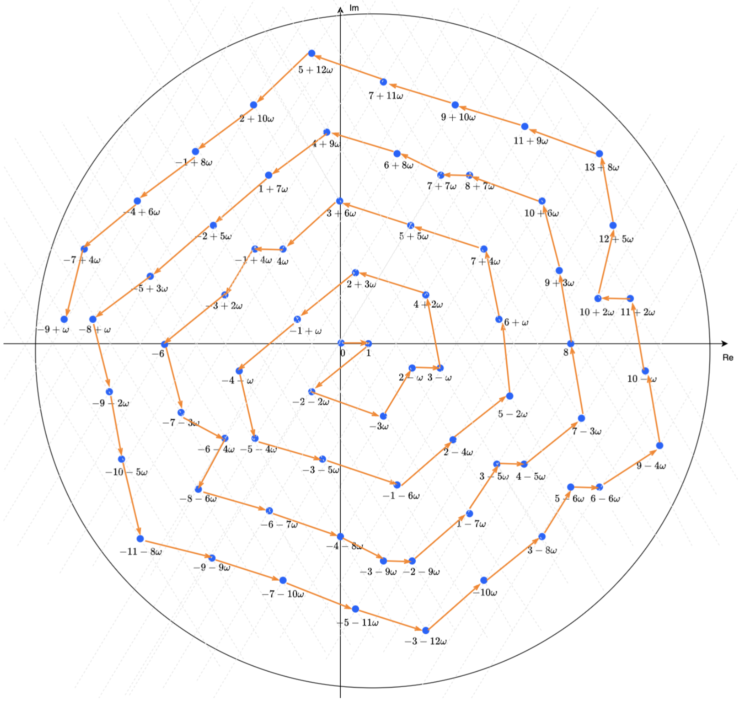


Fig. 6. The construction of the Hamiltonian Path for B when $c = 3$

algorithm ($p = 2$) and our algorithm ($p = 7$) have similar storage costs while our algorithm gains margin computation performance improvements in several cases. With efficient endomorphism, our algorithm ($p = 2-\omega$) achieves significant storage savings without sacrificing the computation efficiency.

Table 6. Theoretical Comparison of Storage and Computation Cost of Computing $S_{n,r}$ over $\mathbb{G}_1$ with Different Methods

	Repaired LFG				Ours $p = 7$				Ours $p = 2 - \omega$			
n	q	h	S	C	q	h	S	C	q	h	S	C
2^{10}	2^{13}	20	2.98 MB	2.33×10^4	7^5	19	2.93 MB	2.33×10^4	$(2-\omega)^5$	19	0.98 MB	2.39×10^4
2^{11}	2^{14}	19	5.67 MB	4.45×10^4	7^5	19	5.87 MB	4.38×10^4	$(2-\omega)^5$	19	1.96 MB	4.44×10^4
2^{12}	2^{14}	19	11.34 MB	8.55×10^4	7^5	19	11.74 MB	8.47×10^4	$(2-\omega)^5$	19	3.91 MB	8.53×10^4
2^{13}	2^{16}	16	19.17 MB	1.54×10^5	7^6	16	19.95 MB	1.59×10^5	$(2-\omega)^6$	16	6.65 MB	1.63×10^5
2^{14}	2^{16}	16	38.33 MB	2.93×10^5	7^6	16	39.9 MB	2.98×10^5	$(2-\omega)^6$	16	13.3 MB	3.02×10^5
2^{15}	2^{16}	16	76.67 MB	5.71×10^5	7^6	16	79.8 MB	5.77×10^5	$(2-\omega)^6$	16	26.6 MB	5.81×10^5
2^{16}	2^{19}	14	134.56 MB	1.10×10^6	7^7	13	131.43 MB	1.05×10^6	$(2-\omega)^7$	13	43.81 MB	1.08×10^6
2^{17}	2^{20}	13	250.35 MB	2.06×10^6	7^7	13	262.86 MB	1.97×10^6	$(2-\omega)^7$	13	87.62 MB	2.01×10^6
2^{18}	2^{20}	13	500.7 MB	3.90×10^6	7^7	13	525.73 MB	3.81×10^6	$(2-\omega)^7$	13	175.24 MB	3.83×10^6
2^{19}	2^{20}	13	1 GB	7.56×10^6	7^7	13	1.05 GB	7.47×10^6	$(2-\omega)^7$	13	358.4 MB	7.51×10^6
2^{20}	2^{22}	12	1.85 GB	1.45×10^7	7^8	12	1.95 GB	1.46×10^7	$(2-\omega)^8$	12	665.6 MB	1.48×10^7
2^{21}	2^{22}	12	3.71 GB	2.82×10^7	7^8	12	3.91 GB	2.82×10^7	$(2-\omega)^8$	12	1.31 GB	2.84×10^7

S: storage cost C: computation cost in the number `Add` operations

6.4 Implementation and Experimental Evaluation Results

We only consider the computation of $S_{n,r}$ over $\mathbb{G}_1$ in our implementation, which can be easily extended to $\mathbb{G}_2$. Specifically, we implement the repaired LFG method, our general optimal bucket construction with $p = 7$ and $M = \{1, 2, 3\}$, and endomorphism with the efficient bucket construction. We use the code base[7] of [11] as much as possible to make a fair evaluation.

All experiments are done on an Apple MacBook Pro with 3.2 GHz M1 Max chip and 64GB memory, and we summarize the results in Table 7. As shown in Table 7, the repaired LFG implementation with $p = 2$ and $M = \{1, 2, 3\}$ is about 15.8% to 40.6% faster than the Pippenger one in the blst library. In addition, the general optimal bucket construction with $p = 7$ can achieve the similar performance with the repaired LFG method with $p = 2$ and the same multiplier set $M = \{1, 2, 3\}$. For certain values of n our method with $p = 7$ can achieve a modest improvement of up to 4.4% such as for $n = 2^{17}$, when compared to the repaired LFG approach. For the endomorphism case with $p = 2 - \omega$ and the multiplier set $M = \{1\}$, we conducted a proof-of-concept implementation for the case of $q = (2 - \omega)^c, c = 2$. For larger n's considered in this paper, $c = 5, 6, 7$ or 8 leads to the best performance based on theoretical complexity analysis in Table 6. While the endomorphism can be leveraged to save storage

[7] https://github.com/LuoGuiwen/MSM_blst/tree/master

cost effectively, the complex scalar recording process is much more involved than their integer counterpart. Moreover, the bucket needs to be pre-constructed and sorted for a selected base $q = (2-\omega)^c$.

Table 7. Experimental Results for Computing $S_{n,r}$ over $\mathbb{G}_1$ with Different Methods

n	Pippenger Implementation in blst[8]	Repaired LFG $p = 2$ $M = \{1, 2, 3\}$	Our Method $p = 7$ $M = \{1, 2, 3\}$	Our Method $p = 2 - \omega$, $c = 2$ $M = \{1\}$
2^{10}	15.28 ms	9.08 ms	9.07 ms	18.87 ms
2^{11}	27.40 ms	17.70 ms	17.13 ms	38.94 ms
2^{12}	49.15 ms	32.93 ms	32.78 ms	74.47 ms
2^{13}	90.50 ms	61.34 ms	62.17 ms	153.84 ms
2^{14}	166.07 ms	113.27 ms	116.27 ms	297.96 ms
2^{15}	305.24 ms	217.49 ms	217.49 ms	586.75 ms
2^{16}	556.75 ms	440.38 ms	423.15 ms	1.24 s
2^{17}	1.05 s	849.76 ms	809.67 ms	2.63 s
2^{18}	1.95 s	1.54 s	1.51 s	4.93 s
2^{19}	3.57 s	2.94 s	2.91 s	9.62 s
2^{20}	6.91 s	5.85 s	5.86 s	19.83 s
2^{21}	13.3 s	11.2 s	11.2 s	39.48 s

7 Conclusion

MSM is the major computation bottleneck for the proof generation of many pairing-based zkSNARK schemes. A major direction for MSM acceleration is making trade-offs between storage and computation. Both the popular Pippenger algorithm and the recent LFG algorithm follow this direction.

In this paper, we revised an important property proposed in the LGF algorithm and designed a more efficient MSM algorithm. The performance of the new algorithm is verified by both theoretical analysis and experiment. Furthermore, we proposed a method to find the optimal bucket size under the LGF framework.

We also introduced a bucket-amenable recoding using fast endomorphisms on $j = 0$ elliptic curves to ideally divide the storage requirement by 3, at almost no performance penalty, compared to our LFG already optimized algorithm. We showed how this method can be used to speed up Pippenger's algorithm. Future investigations will be devoted to optimizing the construction of bucket sets in the endomorphism case and to deal with larger multiplier sets.

[8] `https://github.com/supranational/blst`

References

1. Ben-Sasson, E., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., Virza, M.: Zerocash: Decentralized anonymous payments from bitcoin. In: IEEE S&P 2014. pp. 459–474. IEEE (2014)
2. Bernstein, D.J., Doumen, J., Lange, T., Oosterwijk, J.J.: Faster batch forgery identification. In: Galbraith, S., Nandi, M. (eds.) Progress in Cryptology - INDOCRYPT 2012. pp. 454–473. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
3. Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications (extended abstract). In: Simon, J. (ed.) Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA. pp. 103–112. ACM (1988)
4. Brickell, E.F., Gordon, D.M., McCurley, K.S., Wilson, D.B.: Fast exponentiation with precomputation. In: Rueppel, R.A. (ed.) Advances in Cryptology — EUROCRYPT' 92. pp. 200–207. Springer Berlin Heidelberg, Berlin, Heidelberg (1993)
5. Gennaro, R., Gentry, C., Parno, B., Raykova, M.: Quadratic span programs and succinct nizks without pcps. In: Advances in Cryptology - EUROCRYPT 2013. LNCS, vol. 7881, pp. 626–645. Springer (2013)
6. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof-systems (extended abstract). In: ACM STOC 1985. pp. 291–304. ACM (1985)
7. Groth, J.: Short pairing-based non-interactive zero-knowledge arguments. In: Abe, M. (ed.) Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6477, pp. 321–340. Springer (2010). https://doi.org/10.1007/978-3-642-17373-8_19, https://doi.org/10.1007/978-3-642-17373-8_19
8. Groth, J.: On the size of pairing-based non-interactive arguments. In: Fischlin, M., Coron, J. (eds.) Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II. Lecture Notes in Computer Science, vol. 9666, pp. 305–326. Springer (2016). https://doi.org/10.1007/978-3-662-49896-5_11, https://doi.org/10.1007/978-3-662-49896-5_11
9. Kilian, J.: A note on efficient zero-knowledge proofs and arguments (extended abstract). In: ACM STOC 1992. pp. 723–732. ACM (1992)
10. Lang, S.: Algebraic Number Theory, Graduate Texts in Mathematics, vol. 110. Springer (1986)
11. Luo, G., Fu, S., Gong, G.: Speeding up multi-scalar multiplication over fixed points towards efficient zksnarks. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2023**(2), 358–380 (2023). https://doi.org/10.46586/TCHES.V2023.I2.358-380, https://doi.org/10.46586/tches.v2023.i2.358-380
12. Maller, M., Bowe, S., Kohlweiss, M., Meiklejohn, S.: Sonic: Zero-knowledge snarks from linear-size universal and updatable structured reference strings. In: Cavallaro, L., Kinder, J., Wang, X., Katz, J. (eds.) Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019. pp. 2111–2128. ACM (2019). https://doi.org/10.1145/3319535.3339817, https://doi.org/10.1145/3319535.3339817
13. Pippenger, N.: On the evaluation of powers and related problems. In: 17th Annual Symposium on Foundations of Computer Science (sfcs 1976). pp. 258–263 (1976). https://doi.org/10.1109/SFCS.1976.21