Research Article

Jiao Han and Jincheng Zhuang*

DLP in semigroups: Algorithms and lower bounds

https://doi.org/10.1515/jmc-2021-0049
received November 17, 2021; accepted July 23, 2022

Abstract: The discrete logarithm problem (DLP) in semigroups has attracted some interests and serves as the foundation of many cryptographic schemes. In this work, we study algorithms and lower bounds for DLP in semigroups. First, we propose a variant of the deterministic algorithm for solving the cycle length of torsion elements and show the lower bound of computing the DLP in a semigroup. Then, we propose an algorithm for solving the multiple discrete logarithm (MDL) problem in the semigroup and give the lower bound for solving the MDL problem by considering the MDL problem in the generic semigroup model. Besides, we solve the multidimensional DLP and product DLP in the semigroup.

Keywords: discrete logarithm problem, semigroups, Torsion elements, cycle length, lower bounds

MSC 2020: 11T71, 11Y16

1 Introduction

The discrete logarithm problem (DLP) in finite groups is widely used in cryptography. It furnishes the foundation for the security of diverse schemes such as the Diffie–Hellman cryptographic protocol [1]. There exist many general purpose algorithms to solve DLP in the finite group G of order N, for instance, Shank's baby-step-giant-step algorithm [2] and Pollard's Rho algorithm [3], which perform $O(\sqrt{N})$ group operations. On the other hand, Shoup [4] established that the lower bound of algorithms for solving the DLP in finite groups in the generic group model is $\Omega(\sqrt{p})$, where p is the largest prime factor of N. Moreover, for the multiple discrete logarithm (MDL) problem with n instances, Kuhn and Struik [5] designed an algorithm by extending the Rho method with $\sqrt{np}$ complexity. Later, Yun [6] analyzed the generic hardness of the MDL problem and proved that any generic algorithm to solve the MDL problem should at least make $\Omega(\sqrt{np})$ oracle queries by studying hardness of the search-by-hyperplane-queries (SHQ) problem.

Besides, researchers have considered other variants of the DLP problem for the sake of enabling new cryptographic functionalities or generating hard instances of the DLP faster than previous methods. For example, Brands [7] first proposed the multidimensional DLP (also called representation problem). Given $x_1, x_2, \ldots, x_n, y \in G$ and $S_1, S_2, \ldots, S_n \subseteq \mathbb{Z}$, the multidimensional DLP is to determine $m_i \in S_i$ such that $y = x_1^{m_1} x_2^{m_2} \cdots x_n^{m_n}$. Knuth [8] proposed the product DLP. Given $x, y \in G$ and $S_1, S_2, \ldots, S_n \subseteq \mathbb{Z}$, the product DLP is to compute $m_i \in S_i$ such that $y = x^{m_1 m_2 \cdots m_n}$.

In recent years, many researchers have studied cryptographic schemes in the environment of semigroups and semirings. Monico [9] generalized the original DLP to the DLP in semigroups and semirings, and proposed the semigroup action problem (SAP). At the same time, he explained that the DLP can be extended to the SAP

* **Corresponding author: Jincheng Zhuang,** School of Cyber Science and Technology, Shandong University, Qingdao 266237, China; Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Qingdao 266237, China; Quancheng Laboratory, Jinan 250103, China, e-mail: jzhuang@sdu.edu.cn
Jiao Han: School of Cyber Science and Technology, Shandong University, Qingdao 266237, China; Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Qingdao 266237, China

and used the SAP to define a new Diffie–Hellman key exchange protocol and ElGamal cryptosystem. Monico et al. [10] proposed to use the finite Abelian semigroup as the platform for the Diffie–Hellman key exchange protocol. Kahrobaei et al. [11] studied the Diffie–Hellman protocol in the semigroup of matrices over a group ring and discussed the computational Diffie–Hellman and decisional Diffie–Hellman problem in this setting. Habeeb et al. [12] described a new key exchange protocol based on a noncommutative semigroup. In addition, Sakalauskas [13–15] also introduced some signature schemes based on the problem in monoids and semirings. Goel et al. [16] introduced several cryptographic schemes in semigroups and semirings in detail. Thus, it is of importance to study DLP in semigroups. In this article, we focus on studying the DLP in semigroups.

Researchers have developed some algorithms for solving DLP in semigroups by extending known algorithms for DLP in finite groups. One obstacle to this process is that inverse elements may not exist in semigroups. This makes the previous algorithm not directly applicable to semigroups. But when a special element called the torsion element is used as a base, many known algorithms can be effectively promoted. Let S be a semigroup, an element $g \in S$ is a torsion element if the set $\langle g\rangle = \{g^m : m \in \mathbb{Z}\}$ is a finite subsemigroup. In other words, a certain part of $\langle g\rangle$ is cyclic. The smallest positive integer s such that $g^s = g^m$, for some $m \in \mathbb{N}$, $s < m$ is called the cycle start. The smallest positive integer L such that $g^{s+L} = g^s$ is called the cycle length. The DLP problem in semigroup S is: given a torsion element $g \in S$ and $h \in \langle g\rangle$, where the order of $\langle g\rangle$ is N, find $m \in [1, N]$ such that $h = g^m$.

Utilizing the cyclic characteristics of the torsion element, Banin and Tsaban [17] proposed a technique to reduce the semigroup DLP to the DLP in a cycle group. They show that the cycle length of the torsion element plays an analogous significant role in the order of the finite group. In particular, as long as we find the cycle length of the torsion element, we can effectively solve the DLP in semigroups. A method for solving the cycle length of torsion elements was first proposed by Monico [9], which is a probabilistic algorithm. Later, Banin and Tsaban [17] gave another probabilistic algorithm, which used the DLP oracle. Recently, Tinani and Rosenthal [18] proposed a deterministic algorithm for solving the cycle length of a torsion element and a deterministic algorithm for computing the DLP in semigroups. In addition to the classical algorithm, Childs and Ivanyos [19] provided a quantum algorithm for solving DLP in semigroups.

This article continues the study of DLP in semigroups. We present a variant algorithm for solving the cycle length of a torsion element and analyze the lower bound of the algorithm for computing DLP in a semigroup. Besides, we give the algorithm and complexity analysis of computing MDL, multidimensional DLP, and product DLP in a semigroup.

Article organization: This article is organized as follows. In Section 2, we provide some preliminaries. In Section 3, we give a variant of deterministic algorithm for solving the cycle length of a torsion element and show the lower bound of algorithms for solving DLP in semigroups. In Section 4, we analyze the generic hardness of the MDL problem in semigroups. In Section 5, we study several variants of DLP, including the multidimensional DLP and the product DLP. In Section 6, we conclude this article.

2 Preliminaries

Recall that the DLP problem in semigroup S is: given a torsion element $g \in S$ and $h \in \langle g\rangle$, where the order of $\langle g\rangle$ is N, find $m \in [1, N]$ such that $h = g^m$. The following lemmas are the properties of torsion elements, which will be used in the design of algorithms.

Lemma 1. [17] *Let S be a semigroup and $g \in S$ be a torsion element. Suppose L and s are cycle length and cycle start of g, respectively, and t is the minimal positive integer such that $tL \geq s$. Then the set $G = \{g^{s+k} : 0 \leq k \leq L\}$ is a cyclic group whose order is L. Furthermore, the identity element of G is g^{tL} and the generator of G is g^{tL+1}.*

Lemma 2. [9] *Let S be a semigroup and $g \in S$ be a torsion element with cycle length L and cycle start s. We have*

$$g^x = g^y \;\Leftrightarrow\; x \equiv y \pmod{L},$$

where x and y are all integers that satisfy x and $y \geq s$.

Lemma 3. [18] *Let S be a semigroup, $g \in G$ be a torsion element with cycle length L and cycle start s and $G = \{g^{s+k} : k \geq 0\}$. For fixed positive integers a, b, b', if $g^{mL+a} = g^b \in G$ and $g^{a-nL} = g^{b'} \in G$, where m is the smallest integer such that $g^{mL+a} \in G$ and n is the largest integer such that $g^{a-nL} \in G$, we have*

$$mL + a \leq b, a - nL \leq b'.$$

Banin and Tsaban [17] proposed a method to reduce the DLP in the semigroup S to the DLP in the cyclic group. The premise of this reduction method is that the cycle length and the cycle start of the torsion element is known. Thus, by combining the algorithm for solving cycle length and cycle start with the reduction method, we can obtain a complete algorithm for solving the DLP in semigroups. So how to calculate the cycle length is of critical importance.

Let S be a semigroup and $g \in S$ be a torsion element whose order is N. In 2002, Monico [9] first presented a probabilistic algorithm to solve the cycle length of g. Tinani and Rosenthal [18] analyzed the probability of the algorithm and concluded that the probability that this algorithm succeeds is bounded below by $\left(1 - \frac{1}{B}\right)^{\log a}$, where B is the fixed bound that satisfies a certain condition and a is the g greatest common divisor of the calculations in the algorithm. In 2016, Banin and Tsaban [17] gave another probabilistic algorithm that used DLP oracle in the process to compute the cycle length. Tinani and Rosenthal [18] showed that the algorithm requires $O((\log N)^2)$ semigroup operations. Recently, Tinani and Rosenthal [18] proposed a deterministic algorithm for solving cycle length, which is described as follows.

(1) Initialize $M = 1$, set $p = \lceil \sqrt{M} \rceil$.
(2) Compute $g^M, g^{M+1}, \ldots, g^{M+p-1}$ and store these pairs $(M + i, g^{M+i})$. If there exist $0 \leq i < p$ such that $g^M = g^{M+i}$, then $L = i$. Otherwise proceed to the next step.
(3) For $0 \leq j \leq p$, compute $g^{M+p}, g^{M+2p}, \ldots, g^{M+jp}$ and look up whether there is a matching value in the table. If g^{M+i} in the table such that $g^{M+i} = g^{M+jp}$, then $L = jp - i$. If the collision is not found, set $M = 2M$ and continue this process.

They showed that the time complexity is $O(\sqrt{N} \log N)$ semigroup multiplications, and the space complexity is $O(\sqrt{N})$ semigroup elements.

Next, we describe the definition of the SHQ problem proposed in [6], which will be used in Section 4 where we study the generic hardness of the MDL problem in semigroups.

Definition 1. [6] Let $\mathbb{Z}_L^n$ be the n-dimensional affine space in the finite field $\mathbb{Z}_L$. We also assume $n = o(L)$: consider a family of numbers, so that there is a main parameter λ, and both n and L are functions of λ, and $n(\lambda)/L(\lambda) \to 0$, as $\lambda \to \infty$. The affine hyperplane H in $\mathbb{Z}_L^n$ can be written in the form of $c_1X_1 + c_2X_2 + \cdots + c_nX_n = b$, where $c_1, c_2, \ldots, c_n, b \in \mathbb{Z}_L$ with $c_i \neq 0$ for some i, and $X_1, X_2, \ldots, X_n$ are the canonical coordinate functions in $\mathbb{Z}_L^n$. For the point $\overrightarrow{a}$ in affine space, we define

$$\mathbf{H}(\overrightarrow{a}, H) = \begin{cases} 1 & \overrightarrow{a} \in H, \\ 0 & \text{otherwise}. \end{cases}$$

The SHQ problem is defined as follows: Choose uniform random point $\overrightarrow{a} \in \mathbb{Z}_L^n$ and guess the position of a correctly.

Given access to a hyperplane query oracle $\mathbf{H}(\overrightarrow{a}, H)$, the advantage of a solver A to solve SHQ is defined as follows:

$$\mathbf{Adv}_{p,n}^{shq}(A) = \mathbf{Pr}\left[A^{\mathbf{H}(\overrightarrow{a},H)}(p, n) = \overrightarrow{a}\right].$$

If there is a constant $c > 0$ that satisfies $\mathbf{Adv}_{N,n}^{mdl}(A) \geq c$ for any λ, we say that A has a constant advantage in solving SHQ.

3 Algorithm for DLP in a semigroup

3.1 Our algorithm for cycle length

In this section, we propose a variant of the deterministic algorithm based on [18], where semigroup is finite with known order, which is reasonable in the application. The key observation is that we can avoid some duplicated collision test by adjusting the starting point of the baby step. Our variant algorithm for cycle length is expressed as Algorithm 1.

Algorithm 1. A variant algorithm for cycle length

Input A semigroup S with $|S| = H$, a torsion element $g \in S$.
Output Cycle length L of g.
1: Initialize $M = 1$, $k = 0$, an empty table A, $T = g^H$.
2: Let $m = \lceil \sqrt{M} \rceil$,
3: Compute Tg^k, $Tg^{k+1}, \ldots, Tg^{m-1}$ and add to table A.
4: **if** $\exists 1 \le i < j \le m$ such that $Tg^i = Tg^j$, $L = j - i$; **break**.
5: **else for** $1 \le j \le m$:
6: Compute Tg^{jm}, and test equality with the elements in table A.
7: **if** $\exists 1 \le i \le m - 1$ such that $Tg^{jm} = Tg^i$, $L = jm - k - i$.
8: **else** let $M = 2 \cdot M$, $k = m$ and go back to step 2.

Next, we analyze the variant and compare it with the original algorithm of Tinani and Rosenthal [18]. Notice that the key difference is the setting of the starting point of the baby step. The original algorithm starts from 1, while the variant starts from a relatively larger point. The advantage is that the variant decreases both the time and space complexity and the number of collision test.

Suppose the order of g is N. When $M = N$, both algorithms must be able to find the cycle length. In other words, the two algorithms iterate at most $\log N$ times in the worst case. We compare the worst complexity of the two algorithms in Table 1. For example, in the baby step, the original algorithm need to perform

$$1 + \sqrt{2} + \sqrt{2^2} + \cdots + \sqrt{N} \approx (2 + \sqrt{2})\sqrt{N}$$

semigroup element power operations. As a comparison, the variant needs to perform $\sqrt{N}$ semigroup element power operations.

We illustrate the algorithm with the following toy example.

Table 1: Time and space complexity of two deterministic algorithms for solving the cycle length. Suppose the order of the torsion element g is N. We call the first and second computational stages in the two algorithms the baby step and the giant step. The measurement units of time and space complexities are semigroup element power operation and semigroup elements, respectively

	Tinani and Rosenthal's algorithm [18]	Our algorithm
The baby step	$\approx(2 + \sqrt{2})\sqrt{N}$	$\sqrt{N}$
The giant step	$\approx(2 + \sqrt{2})\sqrt{N}$	$\approx(2 + \sqrt{2})\sqrt{N}$
Total time complexity	$\approx(4 + 2\sqrt{2})\sqrt{N}$	$\approx(3 + \sqrt{2})\sqrt{N}$
Space complexity	$\sqrt{N}$	$\sqrt{N}$

Example 1. Assuming $S = (\mathbb{Z}_{836}, \times)$, compute the cycle length L of $\langle 6\rangle \subset S$.

(1) Tinani and Rosenthal's algorithm [18]:

– $M = 1, q = 1$. The baby step: $6^M = 6$, store $(6, 1)$. The giant step: $6^{M+1} = g^2 = 36$.

– $M = 2, q = 2$. The baby step: $6^M = 6^2 = 36$, $6^{M+1} = 6^3 = 216$, store $(36, 2), (216, 3)$. The giant step: $6^{M+2} = 6^4 = 460$, $6^{M+4} = 6^6 = 676$.

…

– $M = 128, q = 12$. The baby step: $6^M = 6^{128} = 796$, $6^{M+1} = 6^{129} = 596, \ldots, 6^{M+11} = 6^{139} = 156$, store $(796, 128)$, $(129, 129), \ldots, (156, 139)$. The giant step: $6^{M+12} = 6^{140} = 100$, $6^{M+24} = 6^{152} = 168, \ldots 6^{M+96} = 6^{224} = 548$.

$$6^{M+6} = 6^{M+96} \Rightarrow L = 90.$$

(2) Our algorithm : $T = g^{836} = 16$.

– $M = 1, m = 1, k = 0$. The baby step: $T = 6^{836} = 16$, store $(16, 0)$. The giant step: $T6^1 = 6^{837} = 96$.

– $M = 2, m = 2, k = 1$. The baby step: $T6^1 = 6^{837} = 96$, store $(96, 1)$. The giant step: $T6^2 = 6^{838} = 576$, $T6^4 = 6^{840} = 672$.

– $M = 4, m = 2, k = 2$. The baby step: No calculation required. The giant step: No calculation required.

– $M = 8, m = 3, k = 2$. The baby step: $T6^2 = 6^{838} = 576$, store $(576, 2)$. The giant step: $T6^3 = 6^{839} = 112$, $T6^6 = 6^{842} = 784$, $T6^9 = 6^{845} = 472$.

…

– $M = 128, m = 12, k = 8$. The baby step: $T6^8 = 6^{844} = 636$, $T6^9 = 6^{845} = 472$, $T6^0 = 6^{846} = 324$, $T6^{11} = 6^{847} = 272$, store $(636, 8)$, $(472, 9)$, $(324, 10)$, $(272, 11)$. The giant step: $T6^{12} = 6^{848} = 796$, $T6^{24} = 6^{860} = 100$, ... $T6^{96} = 6^{932} = 784$.

$$6^{836+6} = 6^{836+96} \Rightarrow L = 90.$$

The concrete complexity of calculating the example is summarized in Table 2

In the original randomized algorithm [17], they computed collision based on DLP oracle. But the details are not presented. In Appendix A, we complement the details.

3.2 The lower bound

Shoup [4] proposed the generic algorithm and proved the lower bound of DLP in a group. Following Shoup, we can model a generic group using a random infective function $\sigma : \mathbb{Z}_N \to \{0, 1\}^* = \mathcal{L}$. We then write the elements of an order-N group as $\{\sigma(1), \sigma(2), \ldots, \sigma(N)\}$, instead of the usual $\{g, g^2, \ldots, g^N\}$. We often say

Table 2: Complexity of Example 1

	Tinani and Rosenthal's algorithm [18]	Our algorithm
The baby step	38	12
The giant step	34	31
Total time complexity	72	43
Space complexity	12	12

$i \in \mathbb{Z}_N$ is the "discrete log" of $\sigma(i)$. A generic algorithm takes input a list of $(\sigma(x_1), \ldots, \sigma(x_L))$ and has access to a group oracle O_σ: $O_\sigma(\sigma(x_i), \sigma(x_j)) = \sigma(x_i + x_j)$. Generic discrete logarithm algorithm takes as input $(\sigma(1), \sigma(x))$, representing (g, g^x), make queries to O_σ, outputs x. Utilize the generic algorithm and the polynomial zero-point theorem, and it can be shown that the lower bound for computing DLP in an N-order group is $\Omega(N)$.

A group can be regarded as a special semigroup, so it is easier to calculate the DLP in a group than in a semigroup. In other words, the lower bound of algorithms for calculating the DLP in a semigroup will not be lower than the DLP in a group. Combining with the result of Shoup, we can draw the following conclusion.

Theorem 1. *Suppose the order of semigroup $G = \langle g \rangle$ generated by g is N, the lower bound of algorithms for solving the DLP in this semigroup is $\Omega(\sqrt{N})$.*

4 The multiple DLP

4.1 The algorithm for multiple DLP

In this section, we will consider the multiple DLP in a semigroup. We propose an algorithm based on the method of [5]. Recall the definition of the MDL as follows.

Definition 2. Let S be a semigroup and $g \in S$ be a torsion element with cycle length L and cycle start s, where the order of $\langle g \rangle$ is N. The MDL problem is: given $h_1, h_2, \ldots, h_n \in \langle g \rangle$, find out $a_i \in [1, N]$ such that $h_i = g^{a_i}$, $1 \le i \le n$.

Note that there are algorithms for finding the cycle length L and the cycle start s as described in the previous section and Banin and Tsaban [17]. Hence, parameters L and s are included in the input, and the order of g satisfies $N = s + L$.

The main idea of the algorithm for n instances DLP in the semigroup is: First, reduce n instances to the MDL problem in the finite group G_g. Then use the extended Pollard Rho algorithm to solve the MDL problem. Finally, return the result to the semigroup. The full algorithm is given in Algorithm 2.

Algorithm 2. Algorithm for MDL problem

Input A semigroup S, a torsion element $g \in S$, with cycle length L and cycle start s, and $h_1, h_2, \ldots, h_n \in S$ with $h_i = g^{m_i}$.

Output m_i such that $h_i = g^{m_i}$.

1: Compute $t = \lceil \frac{s}{L} \rceil$ and let $g' = g^{tL+1} \in G_g$.

2: Find the minimum integers $0 \le a_i \le t$ such that $h_i' = h \cdot g^{a_i L} \in G_g$ using binary search.

3: Use extended Pollard Rho Algorithm for Computing Multiple Discrete Logarithms to compute $m_i' \in \{0, 1, \ldots, L-1\}$ such that $(g')^{m_i'} = h_i'$.

4: Find the minimum integers $b_i \ge 0$ such that $g^{(tL+1)m_i' - b_i L} \in G_g$ using binary search.

5: **Return** $m_i = m_i'(tL + 1) - (a_i + b_i)L$.

Theorem 2. *Let S be a semigroup, g be a torsion element whose cycle start is s and cycle length is L, the order of subsemigroup $\langle g\rangle$ is N. Given $h_1, h_2, \ldots, h_n \in \langle g\rangle$. Then Algorithm 2 solves n instances DLP based on g in the semigroup S using $O(\sqrt{nL} + n(logN)^2)$ semigroup multiplications.*

Proof. Recall that the group $G_g = \{g^{s+k} : k \geq 0\}$ is a cycle group whose order is L and we have the conclusion that the G_g is generated by g^{tL+1} with identity g^{tL}. We can define the parameter t by formula $t = \lceil \frac{s}{L} \rceil$, and we can use formula $g' = g^{tL+1}$ to make $g' \in G_g$. Evidently we can find suitable a_i so that $h_i' = h_i \cdot g^{a_i L}$ belongs to the cycle group G_g. So far we have reduced the DLP $m_i = log_g h_i$ in the semigroup S to the DLP $m_i' = log_{g'} h_i'$ in the group G_g. Thus, we can use extended Pollard Rho Algorithm [5] for computing multiple discrete logarithms to compute m' in $O(\sqrt{nL})$ semigroup multiplications.

For every m_i', we obtain

$$h_i' = (g')^{m'} \Rightarrow h_i \cdot g^{a_i L} = (g^{tL+1})^{m'} \Rightarrow g^{m_i} \cdot g^{a_i L} = (g^{tL+1})^{m'} \Rightarrow g^{m_i + a_i L} = g^{(tL+1)m'}.$$

We have the maximal integers b_i such that $g^{(tL+1)m_i' - b_i L} \in G_g$. Now,

$$g^{(tL+1)m_i' - b_i L} = g^{m_i'(tL+1)} = (g')^{m_i'} = h_i' = h_i \cdot g^{a_i L} = g^{m_i} \cdot g^{a_i L} = g^{a_i L + m_i}.$$

From Lemma 3, for minimum integers a_i and maximal integers b_i, we can see

$$m_i'(tL + 1) - b_i L \leq a_i L + m_i, a_i L + m_i \leq m_i'(tL + 1) - b_i L.$$

So $m_i = m_i'(tL + 1) - (a_i + b_i)L$. Because $m' \leq L$ and $tL \leq L + s$, $b_i \leq L + s + 1 = N + 1$. Since we use binary search to find a_i and b_i that satisfy the conditions, they cost $O((\log N)^2)$ steps in the whole algorithm.

In summary, Algorithm 2 to solve the n instances DLP performs $O(\sqrt{nL} + n(\log N)^2)$ semigroup multiplications. □

4.2 Generic hardness of MDL

In a finite group with order L, Yun [6] obtained the generic hardness of the MDL problem in finite groups through analyzing the advantages of the SHQ problem solver and the relationship between the MDL problem and the SHQ problem. He proved if A is the solver of any generic MDL problem, then a solver B of the SHQ problem can be constructed through A, and satisfying $\mathbf{Adv}_{L,n}^{mdl}(A) \leq \mathbf{Adv}_{L,n}^{shq}(B)$. Besides, he showed that if A is a solver for any generic MDL problem with at most m queries, then $\mathbf{Adv}_{L,n}^{mdl}(A) \leq \frac{1}{L} + \frac{1}{2}(\frac{e(m+n+1)^2}{2nL})^n$. Afterward, through combining the analysis in the section 6.1 of [6], Yun concluded that if solver A solves the MDL problem in group G with a constant advantage, it should make $\Omega(\sqrt{nL})$ queries.

Let $g \in S$ be a torsion with $|\langle g\rangle| = N$ and the cycle length of g be L. That is the order of the cycle group $G = \langle g^{tL+1}\rangle \subseteq \langle g\rangle$ is L. If we can calculate the DLP in $\langle g\rangle$, we must be able to calculate the DLP in G. Then the generic hardness of the MDL problem in $\langle g\rangle$ must not be lower than that in G. Thus, the generic hardness of MDL in $\langle g\rangle$ is $\Omega(\sqrt{nL})$.

5 Variants of the DLP

In this section, we consider some variants of the DLP over semigroups. Since there are algorithms for finding the cycle length L and the cycle start s as described in the previous section and in the study by Banin et al. [17], the parameters L and s are included in the input.

5.1 The multidimensional DLP

In the classical case, variants of the DLP can be used to design e-money protocols, etc. For example, Brands [7] designed an offline electronic cash system based on multidimensional DLP in a group. Experts also use such problems to design applications in different contexts. So we consider variants of the DLP in a semigroup. In this section, we propose the definition of the multidimensional DLP in a semigroup and give an effective algorithm to solve this problem.

Definition 3. Let S be a finite Abelian semigroup. The multidimensional DLP in S is: given $g_1, g_2, \ldots, g_n$ with cycle start s_i and cycle length L_i, respectively, $h \in S$ and $S_i \subseteq [1, N_i]$, where $g_1, g_2, \ldots, g_n$ be torsion elements and N_i is the order of $\langle g_i \rangle$. Find $a_i \in S_i$, if they exist, such that $h = g_1^{a_1} g_2^{a_2} \cdots g_n^{a_n}$.

Because inverse elements may not exist in semigroups, we cannot directly apply methods that are used in the finite group to solve this problem. Therefore, we consider how to reduce the DLP in a semigroup to a DLP in a finite group, so as to solve the multidimensional DLP in the semigroup. The main idea is as follows.

The order of subsemigroup $\langle g_i \rangle$ that is generated by g_i is N_i. As for, we have known the set of $G_i = \{g_i^{s_i+k} : k \geq 0\}$ is a cycle group and $g_i^{t_i L_i}$ is the identity element of G_i, where t_i is the smallest positive integer that makes $g_i^{t_i L_i} \in G_i$. Fixed $t_i = \lceil \frac{s_i}{L_i} \rceil$, define $g_i' = g_i^{t_i L_i + 1} \in G_i$. In group G_i, it has an inverse element $(g_i^{t_i L_i + 1})^{-1}$ such that $g_i'(g_i')^{-1} = g_i^{t_i L_i + 1} \cdot (g_i^{t_i L_i + 1})^{-1} = g_i^{t_i L_i}$. Use binary search to find the minimum integers $0 \leq b_i \leq t_i$ such that $h' = h g_1^{b_1 L_1} g_2^{b_2 L_2} \cdots g_n^{b_n L_n} \in \langle \bigcup G_i \rangle$, where $\langle \bigcup G_i \rangle$ is a subsemigroup of S, which is generated by $\bigcup G_i$. Next, we can reduce the multidimensional DLP to find $a_1', a_2', \ldots, a_n'$ such that

$$h' = (g_1')^{a_1'} (g_2')^{a_2'} \cdots (g_n')^{a_n'}.$$

Let $m = \lceil \frac{n}{2} \rceil$. Because every g_i' has an inverse element in G_i, we can see

$$\begin{aligned} h' \left(g_{m+1}'^{a_{m+1}'}\right)^{-1} \cdots \left(g_n'^{a_n'}\right)^{-1} &= g_1'^{a_1'} \cdots g_m'^{a_m'} \left[g_{m+1}'^{a_{m+1}'} \left(g_{m+1}'^{a_{m+1}'}\right)^{-1}\right] \cdots \left[g_n'^{a_n'} \left(g_n'^{a_n'}\right)^{-1}\right] \\ &= g_1'^{a_1'} \cdots g_m'^{a_m'} g_{m+1}^{t_{m+1} L_{m+1}} \cdots g_n^{t_n L_n}, \end{aligned}$$

where $a_i' \in [0, L_i - 1]$. Once we obtain a_i' using Baby-Step-Giant-Step algorithm such that $h' = (g_1')^{a_1'} (g_2')^{a_2'} \cdots (g_n')^{a_n'}$, the relationship between a_i and a_i' can be represented by

$$\begin{aligned} h' &= h g_1^{b_1 L_1} g_2^{b_2 L_2} \cdots g_n^{b_n L_n} = g_1^{a_1 + b_1 L_1} g_2^{a_2 + b_2 L_2} \cdots g_n^{a_n + b_n L_n} = g_1'^{a_1'} g_2'^{a_2'} \cdots g_n'^{a_n'} \\ &= g_1^{a_1'(t_1 L_1 + 1)} g_2^{a_2'(t_2 L_2 + 1)} \cdots g_n^{a_n'(t_n L_n + 1)}. \end{aligned}$$

Next, use binary search to find the maximal integers c_i such that $g_i^{a_i'(t_i L_i + 1) - c_i L_i} \in G_i$. Eventually, we come to the conclusion that

$$a_i = a_i'(t_i L_i + 1) - (b_i + c_i) L_i.$$

Theorem 3. *Let S be an Abelian semigroup, $g_1, g_2, \ldots, g_n, h \in S$, and $g_1, g_2, \ldots, g_n$ be torsion elements whose cycle start $s_1, s_2, \ldots, s_n$ and cycle length $L_1, L_2, \ldots, L_n$ are known and the order of subsemigroups $\langle g_1 \rangle, \langle g_2 \rangle, \ldots, \langle g_n \rangle$, which are generated by $g_1, g_2, \ldots, g_n$ are $N_1, N_2, \ldots, N_n$, respectively. According to the aforementioned method, we can find $a_i \in S_i, 1 \leq i \leq n$ such that $h = g_1^{a_1} g_2^{a_2} \cdots g_n^{a_n}$ using $O(L^m)$ semigroup multiplications and $O(L^m)$ semigroup elements of storage, where $S_i \subseteq \mathbb{Z}$, $L = \max\{L_1, L_2, \ldots, L_n\}$, $m = \lceil \frac{n}{2} \rceil$.*

Proof. After the aforementioned series of operations, we reduce the DLP to find a_i' such that $h' = g_1'^{a_1'} g_2'^{a_2'} \cdots g_n'^{a_n'}$ and we can obtain the following equation:

$$h' \left(g_{m+1}'^{a_{m+1}'}\right)^{-1} \cdots \left(g_n'^{a_n'}\right)^{-1} = g_1'^{a_1'} \cdots g_m'^{a_m'} g_{m+1}^{t_{m+1} L_{m+1}} \cdots g_n^{t_n L_n},$$

where $g_{m+1}^{t_{m+1}L_{m+1}}, \ldots, g_n^{t_nL_n}$ are fixed values and $a_i' \in [0, L_i - 1]$. When we find a_i' using the Baby-Step-Giant-Step algorithm, we can choose different $a_1', a_2', \ldots, a_m'$ to calculate the right side of the equation and store them. Obviously, we need $O(L^m)$ semigroup multiplications and $O(L^m)$ semigroup elements of storage. Then we can choose different $a_{m+1}', a', \ldots, a_n'$ to calculate the left side of the equation and look for a collision. If a collision occurs, then we find suitable $a_1', a_2', \ldots, a_n'$ to make $h' = g_1'^{a_1'} g_2'^{a_2'} \cdots g_n'^{a_n'}$ hold and we need at most $O(L^m)$ semigroup multiplications. Finally, we can calculate $a_1, a_2, \ldots, a_n$ through $a_i = a_i'(t_iL_i + 1) - (b_i + c_i)L_i$.

In a word, we can solve the multidimensional DLP in semigroup S using $O(L^m)$ semigroup multiplications and $O(L^m)$ semigroup elements of storage. □

5.2 The product DLP

In this section, we consider the product DLP in a semigroup. First, we give the concept of the product DLP in a semigroup. Then, we present how to solve this problem.

Definition 4. Let S be a semigroup, the product discrete logarithm in S is: given a torsion element $g \in S$ cycle start s and cycle length L, $h \in \langle g \rangle$ and $S_i \subseteq [1, N]$, where the order of $\langle g \rangle$ is N and $i = 1, 2, \ldots, n$. Find $a_i \in S_i$, if they exist, such that $h = g^{a_1 a_2 \cdots a_n}$.

The Baby-Step-Giant-Step algorithm [2], which compromises time and space complexity can be used to compute product DLP in a group. Naturally, we will consider whether we can use such a method to solve the product DLP in a semigroup. The main idea is as follows.

The order of subsemigroup $\langle g \rangle$, which is generated by g is N. As for, we have known the set of $G = \{g^{s+k} : k \geq 0\}$ is a cycle group and g^{tL} is the identity element of G, where t is the smallest positive integer that makes $g^{tL} \in G$. Compute $t = \lceil \frac{s}{L} \rceil$ and let $g' = g^{tL+1} \in G$. Use binary search to find the minimum integer $0 \leq b \leq t$ such that $h' = hg^{bL} \in G$. Thus, we have reduced the DLP to find a' such that $h' = g^{a'}$, where $a' \in [0, L - 1]$, and we can use Baby-Step-Giant-Step algorithm to find a'. Then we obtain

$$h' = hg^{bL} = g^{a_1 a_2 \cdots a_n} g^{bL} = g'^{a'} = g^{(tL+1)a'}.$$

Next, use binary search to find the maximal integer c such that $g^{a'(tL+1)-cL} \in G$. We have $a_1 a_2 \cdots a_n = (tL + 1)a' - (b + c)L$, where $a_i \in S_i$. Let $m = \lceil \frac{n}{2} \rceil$, then the relation of $a_1, a_2, \ldots, a_n$ can be obtained by

$$a_1 a_2 \cdots a_m = [(tL + 1)a' - (b + c)L] \frac{1}{a_{m+1}} \frac{1}{a_{m+2}} \cdots \frac{1}{a_n}.$$

To obtain specific values, we can present distinct $a_1, a_2, \ldots, a_m$ to calculate the left side of the equation and store them. Then we can present distinct $a_{m+1}, a_{m+2}, \ldots, a_n$ to calculate the right side of the equation and look for a collision. If a collision occurs, then we seek out suitable $a_1, a_2, \ldots, a_n$ such that $h = g^{a_1 a_2 \cdots a_n}$.

Theorem 4. *Let S be a semigroup, $g, h \in S$, and g be a torsion element whose cycle start s and cycle length L are known and the order of the subsemigroup $\langle g \rangle$, which is generated by g is N. Suppose that elements of G are represented using $O(\log L)$ bits and the group operations can be performed in $O((\log L)^2)$. According to the aforementioned method, we can find $a_i \in S_i$, $1 \leq i \leq n$ such that $h = g^{a_1 a_2 \cdots a_n}$ using $O((\log N)^m N^m)$ bits operations and $O((\log N)^m N^m)$ bits of storage, where $S_i \subseteq [1, N]$, $m = \lceil \frac{n}{2} \rceil$.*

Proof. When we reduce the product DLP to find a' such that $h' = g'^{a'}$, where $g', h' \in G$, $a' \in [0, L - 1]$, and we can use Shank's Baby-Step-Giant-Step algorithm to solve it with $O(\sqrt{L}(\log L)^2)$ bits operations and $O(\sqrt{L}(\log L)^2)$ bits of storage. In the same way, when using Baby-Step-Giant-Step algorithm to find

$a_1, a_2, \ldots, a_n$, it is not difficult to discover that requires $O((\log N)^m N^m)$ bits operations and $O((\log N)^m N^m)$ bits of storage. So, the aforementioned algorithm for the product DLP in semigroup S needs $O((\log N)^m N^m)$ bits operations and $O((\log N)^m N^m)$ bits of storage. □

6 Conclusion

The security of many cryptographic schemes has been based on the DLP in semigroups. Algorithms for DLP in the finite group can be extended to the semigroup by appropriate adjust. In this work, we present algorithms and lower bounds of the DLP and MDL in the semigroup and consider other variants including the multidimensional DLP and the product DLP in the semigroup.

Acknowledgments: This work was partially supported by the National Key Research and Development Project of China (Grant No. 2018YFA0704702) and the Major Basic Research Project of Natural Science Foundation of Shandong Province, China (Grant No. ZR202010220025). The authors thank the anonymous reviewers for very helpful suggestions which improved the paper.

Conflict of interest: Authors state no conflict of interest.

References

[1] Diffie W, Hellman ME, New directions in cryptography. IEEE Trans Inform Theory. 1976;22(6):644–54.
[2] Shanks D. Class number, a theory of factorization, and genera. Proc Symp Math Soc. 1971;20:41–440.
[3] Pollard J. Monte carlo methods for index computation. Math Comput. 1975;32(143):918–24.
[4] Shoup V. Lower bounds for discrete logarithms and related problems. In: International Conference on the Theory and Applications of Cryptographic Techniques. Springer; 1997. p. 256–66.
[5] Kuhn F, Struik R. Random walks revisited: Extensions of pollardas rho algorithm for computing multiple discrete logarithms. In: Selected Areas in Cryptography. Berlin Heidelberg: Springer; 2001. p. 212–29.
[6] Yun A. Generic hardness of the multiple discrete logarithm problem. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin Heidelberg: Springer; 2015. p. 817–36.
[7] Brands S. An efficient off-line electronic cash system based on the representation problem. Technical Report CS-R9323. Amsterdam: CWI; 1993.
[8] Knuth DE. Art of computer programming, volume 2: Seminumerical algorithms. 3rd ed. Addison-Wesley Professional; 1997.
[9] Monico CJ. Semirings and semigroup actions in public-key cryptography. Ph.D. thesis. University of Notre Dame Notre Dame; 2002.
[10] Maze G, Monico C, Rosenthal J. Public key cryptography based on semigroup actions. Adv Math Commun. 2007;1(4):489–507.
[11] Kahrobaei D, Koupparis C, Shpilrain V. Public key exchange using matrices over group rings. Groups-Complexity-Cryptology. 2013;5(1):97–115
[12] Habeeb M, Kahrobaei D, Koupparis C, Shpilrain V. Public key exchange using semidirect product of (semi) groups. In: International Conference on Applied Cryptography and Network Security. Springer; 2013. p. 475–86.
[13] Sakalauskas E. New digital signature scheme in gaussian monoid. Informatica. 2004;15(2):251–70.
[14] Sakalauskas E. One digital signature scheme in semimodule over semiring. Informatica. 2005;16(3):383–94.
[15] Sakalauskas E, Burba T. Digital signature scheme based on action of infinite ring. Inform Technol Control. 2004;31(2):60–4.
[16] Goel N, Gupta I, Dass B. Survey on SAP and its application in public-key cryptography. J Math Cryptol. 2020;14(1):144–52.
[17] Banin M, Tsaban B. A reduction of semigroup DLP to classic DLP. Designs Codes Cryptography 2016;81(1):75–82.
[18] Tinani S, Rosenthal J. A deterministic algorithm for the discrete logarithm problem in a semigroup. CoRR abs/2101.11500, 2021.
[19] Childs AM, Ivanyos G. Quantum computation of discrete logarithms in semigroups. J Math Cryptol. 2014;8(4):405–16.

Appendix

A The details of Banin and Tsaban's algorithm

The original description of the algorithm of Banin and Tsaban [17] is based on the DLP oracle. Now we use a specific algorithm to complement the details.

Actually, the DLP oracle is used to look for l, l' such that $g^l = g^{l'}$ in their algorithm. Given a bound $d > 1$, for each $i < d$, the algorithm needs l such collisions. Then, the greatest common factor is calculated for the multiples obtained from each collision. At this point, we can use specific algorithms to find collisions, such as the rho method, in which the use of distinguished points to find collisions is a highly effective method. We introduce detailed methods to find collisions.

The complete algorithm is: Assuming that the order of the semigroup S is N. Rho method needs to use a pseudo-random function to specify the relationship between elements. The original pseudo-random function is applied to a finite group G of known order, and the exponents can be reduced modulo $|G|$ at each step. However, the cycle length at this time is not known. Now, select random integer $e_1, e_2, \ldots e_m \in [1, N]$, and let $\beta_i = g^{e_i}$. Then y_i is uniformly distributed in $\langle g \rangle$. Then select a random hash function $h : \{0, 1\}^l \to \{1, 2, \ldots, m\}$ to divide the subset $I \subseteq \{0, 1\}^l$ of any size into m groups of the same size with a high probability. We define the function $f : x \mapsto xy_{h(x)}$, where h acts on x as a bit string instead of a semigroup element. The specific process is as follows:

(1) For $i = 1$ to m, select random $e_1, e_2, \ldots, e_m \in [1, N]$, compute $y_i = g^{e_i}$ as distinguished points.
(2) Select random $a_1 \in [1, N]$, $y = g^{a_1}$, $k = a_1$.
(3) $y \leftarrow y \cdot y_i$, $k = a_1 + e_i$, where $i = h(y)$. If y is not a distinguished point, repeat. If y is a distinguished point, store (y, k) in a table.
(4) Select random $a_1 \neq a_2 \in [1, N]$, $y = g^{a_2}$, $k = a_2$. Repeat 2 until y is a distinguished point. If y is not in the list, store the new (y, k) into the table, if (y, k') is already in the table, then
(5) $E = k - k'$.

We can use the aforementioned algorithm instead of the DLP oracle to find the multiple of the cycle length, and find the greatest common factor according to the boundary restrictions in the algorithm.