Solving ECDLP via List Decoding

Fangguo Zhang[1,2] and Shengli Liu[3]

[1] School of Data and Computer Science,
Sun Yat-sen University, Guangzhou 510006, China
[2] Guangdong Key Laboratory of Information Security,
Guangzhou 510006, China
[3] Dept. of Computer Science and Engineering,
Shanghai Jiao Tong University, Shanghai 200240, China
`isszhfg@mail.sysu.edu.cn`

Abstract. We provide a new approach to the elliptic curve discrete logarithm problem (ECDLP). First, we construct Elliptic Codes (EC codes) from the ECDLP. Then we propose an algorithm of finding the minimum weight codewords for algebraic geometry codes, especially for the elliptic code, via list decoding. Finally, with the minimum weight codewords, we show how to solve ECDLP. This work may provide a potential approach to speeding up the computation of ECDLP.

Keywords: *Elliptic curves discrete logarithms, Elliptic code, List decoding, Minimum weight codewords*

1 Introduction

ECC and ECDLP. In the 1980s, Koblitz [18] and Miller [21] opened the door of elliptic-curve cryptography (ECC). Since the introduction of ECC, the elliptic-curve analogues of cryptographic primitives, like public-key encryption, digital signature, key agreement, etc., were set up and deployed widely in information systems, due to the smaller key sizes and more efficient implementations than their traditional siblings with the same security level. In the last decades, ECC primitives have permeated in cryptographic protocols and deployed in a variety of applications.

The security kernel of ECC is the hardness of the *elliptic curve discrete logarithm problem* (ECDLP). Let $\mathcal{E}$ be an elliptic curve defined over a finite field $\mathbb{F}_q$ and $\mathcal{E}(\mathbb{F}_q)$ be the additive group over $\mathcal{E}$. Let $P \in \mathcal{E}(\mathbb{F}_q)$ be a point of prime order p, and let $\langle P \rangle$ be the subgroup generated by P. If $Q \in \langle P \rangle$, then $Q = sP$ for some integer s $(0 \leq s < p)$, and $s := \log_P Q$ is defined as the discrete logarithm of Q to the base P. The problem of finding s, given P, Q and the parameters of $\mathcal{E}$, is called ECDLP. Up to date, Pollard ρ method [25] with complexity $O(\sqrt{p})$ and its refinements are known as the most efficient solutions to ECDLP, except for some special elliptic curves [37, 38, 39, 40, 41, 42]. A good survey of recent works on ECDLP can be found in [9].

ECDLP and Minimum Distance of Elliptic Code. Algebraic geometry codes (AG codes) were introduced in 1977 by V.D. Goppa [11] as a class of linear codes. Elliptic Codes belong to AG codes, and they are constructed from elliptic curve, i.e., algebraic curves of genus $g = 1$. For any $[n, k]$ elliptic code $\mathcal{C}$ constructed from $\mathcal{E}$ over $\mathbb{F}_q$, the minimum distance of $\mathcal{C}$ is either $d = n - k$ or $d = n-k+1$ [29]. Meanwhile, the minimum distance of $\mathcal{C}$ is closely related to the solution of the ECDLP over $\mathcal{E}$. This connection was first noticed by Driencourt and Michon [6], and rediscovered by Cheng [4]. This brought us a new hope of solving ECDLP: it is possible for us to solve ECDLP over $\mathcal{E}$ if we found a codeword of minimum distance for the elliptic code over $\mathcal{E}$. However, computing the minimum distance of a linear code is one of the fundamental problems in algorithmic coding theory. Vardy [34] showed that it is an NP-hard problem for general linear codes, while Cheng [4] proved that it is still NP-hard (under **RP**-reduction) for elliptic codes. Obviously, the problem of finding Minimum Weight Codewords for a linear code is NP hard as well, since a codeword of minimum weight uniquely determine the minimum distance of this linear code. As a result, it is unlikely for us to design an algorithm of finding codewords of minimum weight in polynomial time, perhaps even not in subexponential time. However, for some NP-hard problems, some algorithms of exponential time do beat the trivial exhaustive search solution.

List Decoding. List decoding is a powerful decoding algorithm for linear error-correction codes. It has a longer history than elliptic-curve cryptography and dates back to the works of Elias [8] and Wozencraft [36] in the 1950s. The breakthroughs of list decoding were due to Goldreich and Levin [10] for the Hadamard code, and to Sudan [26] for the Reed-Solomon(RS) codes. For any $[n, k, d]$ linear code, a well-known fact is that if the number of errors t satisfies $t \leq \lceil (d-1)/2 \rceil$, then there must exist a unique codeword within distance $\lceil (d-1)/2 \rceil$ from the received vector. If $t > (d-1)/2$, however, unique decoding is usually impossible. In 1997, Sudan [26] proposed "List Decoding algorithm" and applied it to Reed-Solomon codes to break the barrier of $t > (d-1)/2$ by allowing the algorithm outputting a list of codewords. Later, Shokrollahi and Wasserman [30] extended Sudan's list decoding algorithm to algebraic-geometry codes. In 1999, Guruswami and Sudan [12] improved the bound of t to $n - \sqrt{nk}$ for both RS and AG codes. Up to now, the list decoding algorithm is one of the most powerful decoding methods for RS and AG codes.

Beyond its application in the field of coding theory, it also led to new developments in complexity theory and cryptography. For instance, it results in new constructions of hardcore predicates from one-way permutations, amplifying hardness of boolean functions, construction of extractors [27], computation of the discrete logarithm over finite fields [5], and constructions of cryptographic schemes [17], etc.

Our contribution. In this paper, we consider a new approach to the solution of ECDLP, and provide the first try of using list decoding to solve ECDLP. We believe that our work merely scratches the surface of the potential power of list decoding techniques in solving ECDLP, and expect more results on this topic in the near future. Our contributions are listed as follows:

1. We present a general algorithm of finding Minimum Weight Codewords for any linear code that is list decodable. Meanwhile, we show a specific algorithm of finding minimum weight codewords for AG codes using list decoding.
2. We show how to list decode elliptic codes and designed an algorithm of finding minimum distance codewords for elliptic codes.
3. Our work provides the first method of solving ECDLP via list decoding, which is of theoretical significance.

Organization. The rest of our paper is organized as follows. In Section 2, we review some preliminaries that will be used in our construction. In Section 3, we show how to use List Decoding to find Minimum Weight Codewords of algebraic codes, especially of elliptic codes. In Section 4, we present an algorithm of solving ECDLP problems via List Decoding and give the corresponding analysis. Finally, Section 5 concludes this paper,

2 Preliminaries

If n is a positive integer, define $[n] := \{1, 2, \dots, n\}$. Let $\mathcal{S}$ be a set, then $s \leftarrow \mathcal{S}$ denotes choose an element s from $\mathcal{S}$ uniformly at random. If $\mathsf{Alg.}$ is an algorithm, then $(b_1, b_2, \dots, b_i) \leftarrow \mathsf{Alg.}(a_1, a_2, \dots, a_j)$ means that the algorithm takes $a_1, a_2, \dots, a_j$ as input and outputs $b_1, b_2, \dots, b_i$.

2.1 Elliptic Curve and Elliptic Curve Discrete Logarithm Problems

Let $\mathbb{F}_q$ be a finite field of q elements. An elliptic curve $\mathcal{E}$ over $\mathbb{F}_q$ is a cubic curve defined by Weierstrass equation

$$\mathcal{E}:\ y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6\ (a_i \in \mathbb{F}_q).$$

The set of $\mathbb{F}_q$-rational points of $\mathcal{E}$ is defined as

$$\mathcal{E}(\mathbb{F}_q) := \{(x, y) \in \mathbb{F}_q \times \mathbb{F}_q : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6\} \cup \{\mathcal{O}\},$$

where $\mathcal{O}$ is the point of infinity,

Equipped with the so-called "chord-and-tangent" rule, $\mathcal{E}(\mathbb{F}_q)$ becomes an abelian group [[32], III.2]. Note that if the characteristic of the finite field is larger than 3, the Weierstrass equation of an elliptic curve $\mathcal{E}$ can be transformed into a short but isomorphic one

$$\mathcal{E} : y^2 = x^3 + ax + b,$$

where $a, b \in \mathbb{F}_q$, $4a^3 + 27b^2 \neq 0 \in \mathbb{F}_q$. For detailed information about elliptic curves, we refer the reader to Silverman's book [32].

Let p be a prime integer which is coprime to q. Let GenG be an elliptic curve group genaration algorithm. Taking as input a security parameter 1^κ, GenG outputs q which defines a finite field $\mathbb{F}_q$, an Elliptic Curve $\mathcal{E}$ over $\mathbb{F}_q$, and a point $P \in \mathcal{E}(\mathbb{F}_q)$ of order p. Denote by $\langle P \rangle$ the group of order p generated by P. If $Q \in \langle P \rangle$, it must holds that $Q = sP$ for some integer s, $0 \leq s < p$, which is called the logarithm of Q to the base P and denoted by $\log_P Q$. The problem of finding s, given P, Q and the parameters of $\mathcal{E}$, is known as the Elliptic Curve Discrete Logarithm Problem (ECDLP).

The ECDLP problem is a well-known hard problem. It is an essential base for elliptic curve cryptography and pairing-based cryptography, and has been a major research area in computational number theory and cryptography for the last several decades.

2.2 Linear Error Correction Codes

An $[n, k]$ linear error correction code $\mathcal{C}$ over finite field $\mathbb{F}_q$ is a set of codewords, where each codeword contains n elements of $\mathbb{F}_q$ and all codewords constitute a linear space of dimension k over $\mathbb{F}_q$. Therefore, each codeword can be expressed as a vector of length n over $\mathbb{F}_q$. Given a codeword $\boldsymbol{c} = (c_1, c_2, \ldots, c_n) \in \mathbb{F}_q^n$, its Hamming weight, denoted by $\mathsf{wt}(\boldsymbol{c})$, is defined to be the number of non-zero coordinates, i.e.,

$$\mathsf{wt}(\boldsymbol{c}) = |\{i \mid c_i \neq 0, 1 \leq i \leq n\}|.$$

The distance of two codewords $\boldsymbol{c}_1, \boldsymbol{c}_2$, denoted by $dis(\boldsymbol{c}_1, \boldsymbol{c}_2)$, counts the number of coordinates in which they differ. The minimum distance $d(\mathcal{C})$ of $\mathcal{C}$ is the minimal value of the distances between any two different codewords. In formula,

$$d(\mathcal{C}) := \min_{\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathcal{C}, \boldsymbol{c}_1 \neq \boldsymbol{c}_2} dis(\boldsymbol{c}_1, \boldsymbol{c}_2).$$

By the linearity of $\mathcal{C}$, we know that $d(\mathcal{C})$ is determined by the minimum Hamming weight among all non-zero codewords in $\mathcal{C}$, i.e.,

$$d(\mathcal{C}) = \min_{\boldsymbol{c} \in \mathcal{C} \setminus \{0\}} \mathsf{wt}(\boldsymbol{c}).$$

If a linear $[n, k]$ code $\mathcal{C}$ has d as the minimum distance, then $\mathcal{C}$ is called a $[n, k, d]$ linear code.

For any linear $[n, k]$ code $\mathcal{C}$ over finite field $\mathbb{F}_q$. Suppose that $\mathbf{0} = (0, ..., 0)$ is the transmitted (causal) codeword, and $\boldsymbol{e}$ is a received vector. Define $f(\boldsymbol{e}, t) := |\{\boldsymbol{c} \in \mathcal{C} \setminus \{0\} : |\boldsymbol{e} - \boldsymbol{c}| \leq t\}|$ as the number of noncausal codewords within distance t centered around $\boldsymbol{e}$. If $f(\boldsymbol{e}, t) = m$, then $\boldsymbol{e}$ is m-tuply falsely decodable. Define $D(u, t) := \sum_{|\boldsymbol{e}|=u} f(\boldsymbol{e}, t)$ as the total number of falsely decodable words of weight u, counting on all possible received vectors of weight u. By the linearity of $\mathcal{C}$, for any causal codeword $\boldsymbol{c}$ and any error pattern $\boldsymbol{e}$, $f(\boldsymbol{e}, t)$ also denotes the number of noncausal codewords within distance t centered around the received vector $\boldsymbol{r} = \boldsymbol{c} + \boldsymbol{e}$. According to [2, 20], we have the following results.

Theorem 1 ([2, 20]). *If $|\boldsymbol{e}| = u$, then the average number of noncausal codewords in a decoding sphere of radius t over all error patterns of weight u is given by*

$$\bar{L}(u,t) = \frac{D(u,t)}{\binom{n}{u}(q-1)^u}.$$

For an $[n,k,d]$ RS code, Berlekamp and Ramsey proved that $D(u,t) = \binom{d}{t}\binom{n}{d}(q-1)$ if $u+t=d$, hence

$$\bar{L}(u,t) = \frac{1}{(q-1)^{u-1}}\binom{n-u}{t} \quad \text{if } u+t = n-k+1.$$

2.3 Algebraic-Geometry Codes and elliptic codes

Algebraic-Geometry (AG) Codes are linear error correction codes defined on algebraic curves. The first AG code was due to Goppa [11] who proposed the so-called "Goppa Code". Algebraic-Geometry Codes can be viewed as generalizations of Reed-Solomon codes. Over the years, AG codes attracted much attention since some AG codes results in linear codes with parameters beating the Gilbert-Varshamov bound [11, 31, 33].

Let $\mathbb{F}_q$ be a finite field with q elements and $\mathcal{X}$ be an absolutely irreducible curve over $\mathbb{F}_q$ of genus g. Let $\mathbb{F}_q(\mathcal{X})$ denote the function field defined over $\mathcal{X}$.

A divisor D on a curve $\mathcal{X}$ is a formal sum of points $D = \sum_P n_P P$ on the curve $\mathcal{X}$, where $n_P \in \mathbb{Z} \setminus \{0\}$ for a finite number of points on $\mathcal{X}$. Here n_P denotes the multiplicity of the point P on the curve. The degree of a divisor $D = \sum_P n_P P$ is defined as the sum of n_P, i.e., $deg(D) := \sum_P n_P$. The support of a divisor $supp(D)$ is the set of points with nonzero coefficients. A divisor is called *effective* if all coefficients are non-negative.

For each point $P \in \mathcal{X}$ and any $f \in \mathbb{F}_q(\mathcal{X}) \setminus \{0\}$, we can abstract the notion of evaluation of f at P (denoted by $v_P(f)$) by local parameter and discrete valuation function $v_P : \mathbb{F}_q(\mathcal{X}) \to \mathbb{Z} \cup \{\infty\}$. A point P is said to be a zero of multiplicity m if $v_P(f) = m > 0$, a pole of multiplicity $-m$ if $v_P(f) = m < 0$,

Any function $f \in \mathbb{F}_q(\mathcal{X}) \setminus \{0\}$ can be associated with a so-called principal divisor. The principle divisor of $f \in \mathbb{F}_q(\mathcal{X})$ is defined as $div(f) := \sum_P v_P(f)P$. According to [31](Theorem I.4.11), the degree of a principal divisor is always 0, i.e., $deg(div(f)) = 0$.

Let $G = \sum_P n_P P$ be any divisor of degree k on $\mathcal{X}$. Denote by $\mathcal{L}(G)$ all rational functions $f \in \mathbb{F}_q(\mathcal{X})$ such that the divisor $div(f)+G$ is effective, together with the zero function, i.e.,

$$\mathcal{L}(G) := \{f \mid div(f) + G \text{ is effective}\} \cup \{0\}. \tag{1}$$

By the Riemann-Roch theorem, $\mathcal{L}(G)$ is a vector space over $\mathbb{F}_q$ of finite dimension and its dimension is given by $dim(\mathcal{L}(G)) := k - g + 1$, where g is the genus of $\mathcal{X}$.

Given an irreducible curve $\mathcal{X}$ and the function field $\mathbb{F}_q(\mathcal{X})$ defined over $\mathcal{X}$, let $P_1, P_2, ..., P_n$ be distinct rational points on $\mathcal{X}$. The n points determine a divisor $D := P_1 + P_2 + \ldots + P_n$. Let G be an arbitrary divisor on $\mathcal{X}$ such that $\{P_1, P_2, ..., P_n\} \cap supp(G) = \emptyset$. An AG code $\mathcal{C}(D, G)$ is defined by the following injective mapping $\mathsf{ev} : \mathcal{L}(G) \to \mathbb{F}_q^n$ with

$$\mathsf{ev}(f) := (f(P_1), f(P_2), \ldots, f(P_n))$$

Hence $\mathcal{C}(D, G) = \mathsf{image}(\mathsf{ev})$. If $G = \sum_P n_P P$ is a divisor of degree k, then $\mathcal{C}(D, G)$ is an $[n, k - g + 1, d]$ code over $\mathbb{F}_q$ and $d \geq n - k + 1 - g$. The basic properties of AG codes can be found in [33, 31, 23].

Elliptic Codes. Elliptic curves can be regarded as a special class of algebraic curves of genus $g = 1$, hence Elliptic Codes are just AG codes constructed from elliptic curve. Let $\mathcal{E}$ be an elliptic curve over $\mathbb{F}_q$ and $\mathbb{F}_q(\mathcal{E})$ be the elliptic function field. Recall that there exists an additive abelian group $\mathcal{E}(\mathbb{F}_q)$ with the group operation defined by the "chord-and-tangent" rule on $\mathcal{E}$. As a result, principal divisors on elliptic curve $\mathcal{E}$ satisfy the following property as shown in the following theorem.

Theorem 2. *[28][32] Let $\mathcal{E}$ be an elliptic curve over over $\mathbb{F}_q$. Let $D = \sum_{P \in \mathcal{E}(\mathbb{F}_q)} n_P P$ be a divisor of $\mathcal{E}$. Then D is a principal divisor if and only if $\sum_{P \in \mathcal{E}(\mathbb{F}_q)} n_P = 0$ and $\sum_{P \in \mathcal{E}(\mathbb{F}_q)} n_P \cdot P = \mathcal{O}$, where $n_P \cdot P$ denotes the scalar multiplication over the Elliptic Curve group $\mathcal{E}(\mathbb{F}_q)$ and the summation in $\sum_{P \in \mathcal{E}(\mathbb{F}_q)} n_P \cdot P$ is implemented with the addition defined over group $\mathcal{E}(\mathbb{F}_q)$.*

Given an elliptic curve $\mathcal{E}$ defined over $\mathbb{F}_q$, and let $\mathbb{F}_q(\mathcal{E})$ be the elliptic function field. Let $P_1, P_2, \ldots, P_n \in \mathcal{E}(\mathbb{F}_q)$. Define $D := P_1 + P_2 + \ldots + P_n$ be divisors on $\mathcal{E}$. Let G be another divisor on $\mathcal{E}$ such that $0 < deg(G) = k < n$ and $supp(D) \cap supp(G) = \emptyset$. The elliptic code $\mathcal{C}(D, G)$ is defined by G and D with

$$\mathcal{C}(D, G) := \{(f(P_1), \ldots, f(P_n)) \mid f \in \mathcal{L}(G)\} \subseteq \mathbb{F}_q^n,$$

where $\mathcal{L}(G)$ is defined in (1).

The minimum distance of an $[n, k]$ EC code is either $d = n-k$ or $d = n-k+1$, as shown in [29] [6]. If $d = n-k$, the EC code is a Maximum Distance Separable (MDS) code, otherwise it is an Almost MDS(AMDS) code.

An $[n, k]$ EC code $\mathcal{C}(D, G)$ is an AMDS code iff there exists k elements $P_{i_1}, \ldots, P_{i_k} \in Supp(D)$ such that divisor

$$P_{i_1} + \ldots + P_{i_k} - G$$

is a principle divisor according to [29] .

2.4 List Decoding of Algebraic-Geometry Codes

In 1999, Guruswami and Sudan [12] proposed a list decoding algorithm for both RS and AG codes. The algorithm is able to efficiently output a list of codewords which lie in the sphere of radius up to $t = n - \sqrt{nk}$ centered around the perturbed (noisy) codeword (i.e., received vector). More precisely, the list decoding algorithm $\mathsf{ListDecode}(\mathcal{C}, \boldsymbol{r}, t)$ takes as input a linear $[n, k]$ code $\mathcal{C}$, a received vector $\boldsymbol{r}$ and a parameter $t \leq n - \sqrt{nk}$, and it outputs a list of codewords whose Hamming distances to $\boldsymbol{r}$ are at most t.

Now we recall the Guruswami-Sudan list coding algorithm $\mathsf{ListDecode}(\mathcal{C}, \boldsymbol{r}, t)$ for an $[n, k, d]$ AG-code $\mathcal{C}_{\mathcal{L}}(D, G)$ [13], where $D = P_1 + P_2 + \ldots + P_n$ and G is a one-point divisor of a curve $\mathcal{X}$ of genus g, i.e., $G = \alpha Q$ and $Q \notin supp(D)$. Assume $\alpha > 2g - 2$, then $dim(\mathcal{L}(\alpha Q)) = k = \alpha - g + 1$ by the Riemann-Roch theorem.

The Guruswami-Sudan list decoding consists of three steps: **initialization**, **interpolation** and **root finding**. We will give a brief (and basic) description of the algorithm. We refer the reader to [12] and [13] for details.

The Guruswami-Sudan List Decoding Algorithm: $\mathsf{ListDecode}(\mathcal{C}, \boldsymbol{r}, t)$.
Input: An AG-code $\mathcal{C}_{\mathcal{L}}(D, G)$ determined by Curve $\mathcal{X}$ over $\mathbb{F}_q$ and divisors $G = \alpha Q$ and D, a received vector $\boldsymbol{r} = (r_1, \ldots, r_n)$ and an error bound t, which determines the maximal number of coordinates in which a codeword disagrees with vector $\boldsymbol{r}$ in order for the codeword to be included on the output list.
Output: a list $\Omega_{\boldsymbol{r}}$ of codewords such that $\mathsf{dis}(\boldsymbol{r}, \boldsymbol{c}) \leq t$.

Initialization.

0.1 $\Omega_{\boldsymbol{r}} := \emptyset$.

0.2 Compute list decoding parameters l from n, t and g, where $l \geq \alpha$.

0.3 Fix a **pole basis** $\{\phi_{j_1} : 1 \leq j_1 \leq l - g + 1\}$ of $\mathcal{L}(lQ)$ such that ϕ_{j_1} has at most $j_1 + g - 1$ poles at Q.

0.4 For each P_i, $1 \leq i \leq n$, find a **zero basis** $\{\psi_{j_3, P_i} : 1 \leq j_3 \leq l - g + 1\}$ of $\mathcal{L}(lQ)$ such that P_i is a zero of ψ_{j_3, P_i} with multiplicity (or at least) $j_3 - 1$.

0.5 Compute the set $\{a_{P_i, j_1, j_3} \in \mathbb{F}_q : 1 \leq i \leq n, 1 \leq j_1, j_3 \leq l - g + 1\}$ such that for every i and every j_1, we have $\phi_{j_1} = \sum_{j_3} a_{P_i, j_1, j_3} \psi_{j_3, P_i}$.

Interpolation. Set $s = \frac{l-g}{\alpha}$. Find a nonzero polynomial $H \in \mathcal{L}(lQ)[T]$ of the form

$$H[T] = \sum_{j_2=0}^{s} \sum_{j_1=1}^{l-g+1-\alpha j_2} h_{j_1, j_2} \phi_{j_1} T^{j_2}.$$

Root Finding. Find all roots $h \in \mathcal{L}(\alpha Q)) \subseteq \mathcal{L}(lQ))$ of $H[T]$. For each h, check if $h(P_i) = r_i$ for at least $n - t$ values of $i \in \{1, 2, \ldots, n\}$, and if so, put h in $\Omega_{\boldsymbol{r}}$.

Return $\Omega_{\boldsymbol{r}}$.

3 Finding Minimum Weight Codewords Using List Decoding

By means of List Decoding with proper parameters, it is possible for us to find a minimum weight codeword. Beforehand, we introduce two lemmas. The first lemma tells us the property of list decoding when $d = u + t$, where d is the minimum distance of a $[n, k]$ linear code, u is the number of errors in the received vector (i.e., the Hamming distance between the received vector and causal codeword is u) and t is the error bound of list decoding. The second lemma analyzes the average number of falsely decodable (noncausal) codewords when $d = u + t$ and $u \le t$.

Lemma 1. *For any linear $[n, k, d]$ code $\mathcal{C}$, let $\boldsymbol{c}' = \boldsymbol{c} + \boldsymbol{e}$ be a received vector with causal codeword $\boldsymbol{c} \in \mathcal{C}$ and error vector $\boldsymbol{e}$ with $\mathsf{wt}(\boldsymbol{e}) = u$. Denote the output of the list decoding algorithm $\mathsf{ListDecode}(\mathcal{C}, \boldsymbol{c}', t)$ by set $\Omega_{\boldsymbol{c}'}$.*

1. *If $|\Omega_{\boldsymbol{c}'} \setminus \{\boldsymbol{c}\}| \ge 1$, then for any codewords $\boldsymbol{c}_1 \in \Omega_{\boldsymbol{c}'} \setminus \{\boldsymbol{c}\}$, it holds that $dis(\boldsymbol{c}_1, \boldsymbol{c}) \le u + t$.*
2. *If $u + t = d$ and $u \le t$, then either $\Omega_{\boldsymbol{c}'} = \{\boldsymbol{c}\}$ or $|\Omega_{\boldsymbol{c}'}| \ge 2$. If the latter case happens, then for all $\boldsymbol{c_1} \in \Omega_{\boldsymbol{c}'} \setminus \{\boldsymbol{c}\}$, we have $\hat{\boldsymbol{c}} = \boldsymbol{c} - \boldsymbol{c}_1$ is the minimum weight codeword.*

Proof. 1. List decoding algorithm $\mathsf{ListDec}(\mathcal{C}, \boldsymbol{c}', t)$ will output codewords in the sphere of radius t centered around $\boldsymbol{c}'$. If $|\Omega_{\boldsymbol{c}'} \setminus \{\boldsymbol{c}\}| \ge 1$, we have that $dis(\boldsymbol{c}_1, \boldsymbol{c}') \le t$. Together with the fact $dis(\boldsymbol{c}, \boldsymbol{c}') = \mathsf{wt}(\boldsymbol{e}) = u$, we have $dis(\boldsymbol{c}, \boldsymbol{c}_1) \le u + t$ by the triangle inequality.
2. If $u \le t$, then $dis(\boldsymbol{c}, \boldsymbol{c}') = u \le t$. As a result, $\boldsymbol{c} \in \Omega_{\boldsymbol{c}'}$ always holds. The linearity of code $\mathcal{C}$ ensures that $\hat{\boldsymbol{c}} := \boldsymbol{c} - \boldsymbol{c}_1 \in \mathcal{C}$. Hence $\mathsf{wt}(\hat{\boldsymbol{c}}) \ge d$. If $u + t = d$, then $d \le \mathsf{wt}(\hat{\boldsymbol{c}}) = dis(\boldsymbol{c}, \boldsymbol{c}_1) \le u + t = d$, which means $\mathsf{wt}(\hat{\boldsymbol{c}}) = d$ and $\hat{\boldsymbol{c}} = \boldsymbol{c} - \boldsymbol{c}_1$ is the minimum weight codeword.

Recall that $\bar{L}(u, t)$ denotes the average number of noncausal codewords in a decoding sphere of radius t over all error patterns of weight u. In [2], Berlekamp and Ramsey presents how to compute $\bar{L}(u, t)$ for RS codes when $u + t = d$ and $u \le t$ (see Theorem 1). Now we can generalize this result to any $[n, k, d]$ linear code. Specifically, we obtain $\bar{L}(u, t)$ for elliptic codes when $u + t = d$ and $u \le t$.

Lemma 2. *For any $[n, k, d]$ linear code in which the number of minimum weight codewords is μ, the average number of noncausal codewords in a decoding sphere of radius t over all error patterns of weight u satisfies*

$$\bar{L}(u, t) = \frac{\mu \cdot \binom{d}{t}}{\binom{n}{u} (q-1)^u} \quad \text{if } u + t = d \text{ and } u \le t. \tag{2}$$

Specifically, for an $[n, k]$ Elliptic Code $\mathcal{C}(G, D)$ where G is a divisor of degree k and $D = P_1 + P_2 + \ldots + P_n$. If $u + t = d$ and $u \le t$, then

$$\bar{L}(u, t) = \begin{cases} \frac{1}{(q-1)^{u-1}} \binom{n-u}{t} & \text{if } d = n - k + 1; \\ \frac{\lambda \cdot \binom{u+t}{t}}{\binom{n}{u} (q-1)^{u-1}} & \text{if } d = n - k, \end{cases} \tag{3}$$

where λ denotes the number of subsets $\mathcal{J} = \{i_1, i_2, \ldots, i_k\} \subseteq \{1, 2, \ldots, n\}$ such that $G - \sum_{j \in \mathcal{J}} P_j$ is a principal divisor.

Proof. Recall that if $\mathbf{0} = (0, ..., 0)$ is the transmitted (causal) codeword, and $\boldsymbol{e}$ is a received vector, then $f(\boldsymbol{e}, t) := |\{\boldsymbol{c} \in \mathcal{C} \setminus \{0\} : |\boldsymbol{e} - \boldsymbol{c}| \leq t\}|$ counts the number of noncausal codewords within distance t centered around $\boldsymbol{e}$. Meanwhile, $\bar{L}(u,t) = \frac{\sum_{|\boldsymbol{e}|=u} f(\boldsymbol{e},t)}{\binom{n}{u}(q-1)^u}$ according to [20].

By Lemma 1, if $u + t = d$ and $u \leq t$, then either $\Omega_{\boldsymbol{e}} = \{\mathbf{0}\}$ or $|\Omega_{\boldsymbol{e}}| \geq 2$. If $|\Omega_{\boldsymbol{e}}| \geq 2$, then we have the following facts.

- For each $\boldsymbol{c}_1 \in \Omega_{\boldsymbol{e}} \setminus \{\boldsymbol{c}\}$, the codeword $\boldsymbol{c}_1$ is a noncausal codeword and it must be a codeword of minimum weight d.
- For each $\boldsymbol{c}_1 \in \Omega_{\boldsymbol{e}} \setminus \{\boldsymbol{c}\}$, define $\boldsymbol{e}' = \boldsymbol{c}_1 - \boldsymbol{e}$, then $\boldsymbol{e}'$ is of weight t. Meanwhile the indices of ones in $\boldsymbol{e}$ and $\boldsymbol{e}'$ must be disjoint, i.e., $\{i \mid e_i = 1, i \in [n]\} \cap \{i \mid e'_i = 1, i \in [n]\} = \emptyset$.

There might be many error patterns $\boldsymbol{e}$ resulting in the same codeword of minimum weight. For each codeword of minimum weight, there are exactly $\binom{d}{t}(= \binom{d}{u})$ choices of $\boldsymbol{e}$ of weight u. If there are totally μ codewords of minimum weight, then there are totally $\mu \cdot \binom{d}{t}$ vector $\boldsymbol{e}$ of weight u, each of which exactly results in a noncausal codeword in its sphere.

Eq (2) holds since there are totally $\binom{n}{u}(q-1)^u$ vectors of weight u.

For an $[n, k]$ Elliptic Code, the minimal distance d is either $n-k+1$ or $n-k$. If $d = n - k + 1$, then the Elliptic code is MDS code, the number of the minimal weight codewords is $\binom{n}{d} \cdot (q-1)$. Hence $\bar{L}(u,t) = \frac{\binom{d}{t} \cdot \binom{n}{d} \cdot (q-1)}{\binom{n}{u}(q-1)^u} = \frac{\binom{n-u}{t}}{(q-1)^{u-1}}$, which is consistent to the result for RS codes in [2].

Now we consider the case of $d = n - k$. Given a subset $\mathcal{J} = \{i_1, i_2, \ldots, i_k\} \subseteq \{1, 2, \ldots, n\}$, define a divisor as $D' = \sum_{j \in J} P_j - G$. If D' is a principal divisor, then there exists a function $f \in \mathcal{L}(G)$ such that $D' = div(f)$ due to the fact that $D' + G$ is effective. For such an $f \in \mathcal{L}(G)$, we have $f(P_{i_j}) = 0$ with $j \in [k]$. Consequently, the Hamming weight of the codeword $\boldsymbol{c} = (f(P_{i_1}), f(P_{i_2}), \ldots, f(P_{i_n}))$ is $n-k$, which suggests that $\alpha \cdot \boldsymbol{c}$ is a codeword of minimum weight for all $\alpha \in \mathbb{F}_q^*$. If there are λ subsets $\mathcal{J} = \{i_1, i_2, \ldots, i_k\} \subseteq \{1, 2, \ldots, n\}$ such that $\sum_{j \in \mathcal{J}} P_j - G$ is a principal divisor, then there are $(q-1)\lambda$ codewords of minimum weight. Consequently $\bar{L}(u,t) = \frac{\binom{d}{t} \cdot \lambda \cdot (q-1)}{\binom{n}{u}(q-1)^u}$ according to Eq (2). ∎

Lemma 1 suggests us a way of finding a codeword of minimum weight codeword. If the minimum distance d of $\mathcal{C}$ is known, we can obtain such a codeword of minimum weight, as long as $u + t = d$, $u < t$ and the list decoding algorithm outputs a list of size at least two. According to this idea, we design an algorithm of finding minimum weight codewords for a code $\mathcal{C}$ with unknown minimum distance d, as shown in the next subsection. Lemma 2 helps us the analyze the success probability of the algorithm.

3.1 How to Find Codewords of Minimum Weight

When the minimum distance d is unknown, the intuition is to try a guess d' of the minimum distance. Now we design an algorithm named $\mathsf{FindCodeword}$ which takes as input a guess d' of the minimum distance, an error weight u and a bound t_m of the decoding radius of List Decoding for code $\mathcal{C}$. Firstly, randomly choose a codeword $\boldsymbol{c}$ from $\mathcal{C}$ and an random error of weight u. Compute the perturbed vector $\boldsymbol{c}' := \boldsymbol{c} + \boldsymbol{e}$. Then invoke the List Decoding algorithm to decode the perturbed vector $\boldsymbol{c}'$ to output a list $\Omega_{\boldsymbol{c}'}$ of codewords . By linearity, for every $\boldsymbol{c}_i \in \Omega_{\boldsymbol{c}'}$, $\boldsymbol{c}_i - \boldsymbol{c}$ is a codeword of $\mathcal{C}$. We hope that one of $\boldsymbol{c}_i - \boldsymbol{c}$ is a minimum weight codeword. Below we describe the algorithm and then analyze the probability that the algorithm outputs such a minimum weight codeword.

```
Algorithm FindCodeword($\mathcal{C}, u, d', t_m$):
Input: A $[n,k]$ linear code $\mathcal{C}$ which is list-decodable up to $t_m$ errors; two parameters $u, d' \in \mathbb{Z}^+$ with $u < t_m < d'$.
Output: Abort symbol $\perp$ or a codeword $\hat{\boldsymbol{c}} \in \mathcal{C}$.
Procedure: 1. If $d' - u > t_m$, return $\perp$.
    2. Randomly choose a codeword $\boldsymbol{c} \in \mathcal{C}$.
    3. Randomly choose an error pattern $\boldsymbol{e}$ such that $\mathsf{wt}(\boldsymbol{e}) = u$. Compute
       $\boldsymbol{c}' := \boldsymbol{c} + \boldsymbol{e}$. Set $\Omega_{\boldsymbol{c}'} := \emptyset$.
    4. Invoke $\Omega_{\boldsymbol{c}'} \leftarrow \mathsf{ListDecode}(\mathcal{C}, \boldsymbol{c}', d' - u)$.
    5. If $\Omega_{\boldsymbol{c}'} \setminus \{\boldsymbol{c}\} = \emptyset$, Return($\perp$). Otherwise for each codeword $\boldsymbol{c}_i \in \Omega_{\boldsymbol{c}'} \setminus \{\boldsymbol{c}\}$
       and compute $\hat{\boldsymbol{c}}_i := \boldsymbol{c}_i - \boldsymbol{c}$, where $i = 1, 2, \ldots, |\Omega_{\boldsymbol{c}'}| - 1$.
    6. Choose $\hat{\boldsymbol{c}}$ of minimal weight from $\{\hat{\boldsymbol{c}}_1, \ldots, \hat{\boldsymbol{c}}_{|\Omega_{\boldsymbol{c}'}|-1}\}$.
    7. Return($\hat{\boldsymbol{c}}$).
```

According to [12], the Guruswami-Sudan list decoding algorithm is applicable when $t_m = n - \sqrt{nk}$, and the complexity of $\mathsf{ListDecode}$ is $O(\lambda^6 n^3)$ for any AG codes (here λ is the designed list size). The computational complexity of Algorithm $\mathsf{FindCodeword}(\mathcal{C}, u, d')$ is dominated by $\mathsf{ListDecode}$, hence is of $O(\lambda^6 n^3)$ as well. There are many works aiming to improve the computational complexity of Guruswami-Sudan list decoding algorithm. For example, Beelen et al. [1] defined a general class of one-point algebraic-geometry codes and proposed a more efficient algorithm for the interpolation step in the Guruswami-Sudan list decoder and the complexity was improved to $O(\lambda^5 n^2 \log^2(\lambda n) \log\log(\lambda n))$.

Suppose that the minimum distance of the $[n,k]$ code $\mathcal{C}$ is d. In the case of $u \leq d/2$, $d - u \leq t_m$, we analyze the probability that Algorithm $\mathsf{FindCodeword}(\mathcal{C}, u, d)$ successfully outputs a codeword of minimum weight.

Theorem 3. *For a $[n, k, d]$ linear code $\mathcal{C}$, let μ be the number of minimum weight codewords. If $u \le d/2$ and $d - u \le t_m$, then*

$$\Pr\left[\hat{\boldsymbol{c}} \leftarrow \mathit{FindCodeword}(\mathcal{C}, u, d, t_m) : \mathsf{wt}(\hat{\boldsymbol{c}}) = d\right] \approx \frac{\mu \cdot \binom{d}{u}}{\binom{n}{u}(q-1)^u}, \tag{4}$$

where μ is the number of minimum weight codewords in $\mathcal{C}$.

Proof. It directly follows from Lemma 2. ▮

3.2 The Final Algorithm of Finding Minimal Weight Codewords

With a correct guess of d, Algorithm $\mathsf{FindCodeword}(\mathcal{C}, u, d)$ might be able to output a codeword of minimum weight with some probability (determined by (4)) according to Theorem 3. So we will try to guess the distance with $d' = 3, 4, \ldots, n - k + 1$. Given a specific guess d' of the distance, we will invoke $\mathsf{FindCodeword}(\mathcal{C}, u, d')$ multiple times. This leads to our final algorithm of finding minimal weight codewords as shown below.

Algorithm $\mathsf{MinWeiCodeword}(\mathcal{C}, \Gamma, t_m, T_m)$:

Input: a $[n, k]$ linear code $\mathcal{C}$ which is list-decodable with an unknown minimum distance d; A set Γ which is a subset of $\{3, 4, \ldots, n - k + 1\}$. We assume that the elements in Γ is in ascending order. t_m is the bound determined by the list decoding algorithm and T_m is the maximal number of invoking $\mathsf{FindCodeword}(\mathcal{C}, u, d')$.

Output: abort symbol $\perp$ or a codeword $\hat{\boldsymbol{c}} \in \mathcal{C}$.

```
$\hat{\boldsymbol{c}} := \perp$; $\mathsf{wt}(\hat{\boldsymbol{c}}) := n$
For each $d' \in \Gamma$ (taking $d'$ in ascending order)
    For $u = d' - t_m$ to $\lfloor d'/2 \rfloor$
        For $i = 1$ to $T_m$
            $\hat{\boldsymbol{c}}' \leftarrow \mathsf{FindCodeword}(\mathcal{C}, u, d', t_m)$;
            If $\mathsf{wt}(\hat{\boldsymbol{c}}') < \mathsf{wt}(\hat{\boldsymbol{c}})$ then $\hat{\boldsymbol{c}} := \hat{\boldsymbol{c}}'$.
Return($\hat{\boldsymbol{c}}$)
```

For an $[n, k, d]$ code $\mathcal{C}$, as long as $d \in \Gamma$, the guess of d' takes the value of d sooner or later. In case of $d' = d$, the probability that Algorithm $\mathsf{FindCodeword}(\mathcal{C}, u, d)$ outputs a minimum weight codeword is given by (4) according to Theorem 3. In $\mathsf{MinWeiCodeword}(\mathcal{C}, \Gamma, T_m)$, there are T_m times of invocations of $\mathsf{FindCodeword}(\mathcal{C}, u, d)$ and u can take values from $d - t_m$ up to $\lfloor d/2 \rfloor$. Therefore, $\mathsf{MinWeiCodeword}$ successfully outputs a minimum distance codeword with probability at least

$$\begin{aligned} &\Pr\left[\hat{\boldsymbol{c}} \leftarrow \mathsf{MinWeiCodeword}(\mathcal{C}, \{d\}, t_m, T_m) : \mathsf{wt}(\hat{\boldsymbol{c}}) = d\right] \\ &\ge 1 - \prod_{u=d-t_m}^{\lfloor d/2 \rfloor} \left(1 - \frac{\mu \cdot \binom{d}{u}}{\binom{n}{u}(q-1)^u}\right)^{T_m}. \end{aligned} \tag{5}$$

If the minimum distance d of $\mathcal{C}$ is known, then we can set $\Gamma = \{d\}$, then $\mathsf{MinWeiCodeword}$ successfully outputs a minimum distance codeword with probability at least

$$\Pr\left[\hat{\boldsymbol{c}} \leftarrow \mathsf{MinWeiCodeword}(\mathcal{C}, \{d\}, t_m, T_m) : \mathsf{wt}(\hat{\boldsymbol{c}}) = d\right] \approx 1 - \left(1 - \frac{\mu \cdot \binom{d}{u}}{\binom{n}{u}(q-1)^u}\right)^{T_m}. \tag{6}$$

This approach applies to all list decodable codes. For some linear $[n, k, d]$ codes over $\mathbb{F}_q$, when the choices of n, k, d, q, t_m make (5) noticeable, then it is possible for us to find a codeword of minimum weight in polynomial time with the help of Algorithm $\mathsf{MinWeiCodeword}(\mathcal{C}, \Gamma, t_m, T_m)$.

For any list decodable $[n, k, d]$ code, if we already know d or have a correct guess of d, Algorithm $\mathsf{FindCodeword}(\mathcal{C}, u, d)$ might be able to output a codeword of minimum weight with some probability (determined by (4)). Due to the fact that $d = u + t$, given d we can always choose u as small as possible to make the probability in (4) bigger, as long as $t = d - u$ is allowable in the list decoding algorithm. For AG code, The Guruswami-Sudan list decoding algorithm can make t up to be $t_m = \lceil n - \sqrt{nk} \rceil$(This bound is called by GS bound or Johnson bound). If new development on list decoding makes t_m exceed the current bound of $\lceil n - \sqrt{nk} \rceil$, then Algorithm $\mathsf{FindCodeword}(\mathcal{C}, u, d)$ will become more efficient by setting smaller values for u. For example, if we have an efficient list decoding algorithm to correct the maximum fraction of errors, i.e., $t_m = n - k$ (this is called by the Singleton bound) for some codes, then the codeword of minimum weight of these such codes can be efficiently computed using Algorithm $\mathsf{FindCodeword}(\mathcal{C}, u, d)$.

There do exist some codes, such as Folded Reed-Solomon Codes or Folded AG codes, that achieve or approach Singleton bound of $t_m = n - k$ for every code rate k/n [14, 15, 24]. However, it seems impossible for elliptic codes to have effective list decoding algorithm to achieve or approach the Singleton bound, otherwise $\mathcal{P} = \mathcal{NP}$ is proved (due to the fact that the minimum distance problem of elliptic codes is NP-hard under **RP**-reduction[4])!

3.3 Instantiation from elliptic code $\mathcal{C}[G, D]$

Now we employ $\mathsf{MinWeiCodeword}(\mathcal{C}, \Gamma = \{n - k\}, T_m)$ to find minimum weight codewords of an $[n, k, d]$ elliptic code $\mathcal{C}[G, D]$ with $d = n - k$ (recall that it is an easy problem if $d = n - k + 1$). The essential step is the invocation of Guruswami-Sudan list decoding algorithm. Now we show the implementation of Guruswami-Sudan list decoding for one-point elliptic codes (as far as we know, no work is available suggesting the concrete implementations of Guruswami-Sudan list decoding for EC codes).

For an elliptic curve $\mathcal{E}$ defined over $\mathbb{F}_q$, let $G = k\mathcal{O}$ be a divisor of degree k and $D = P_1 + P_2 + \ldots + P_n$ where P_i's are rational points over $\mathcal{E}$. Then $\mathcal{C}(G, D)$ is an elliptic code. The first and important step of the Guruswami-Sudan list decoding algorithm for elliptic codes is finding out two types basis of $\mathcal{L}(l\mathcal{O})$: the **pole basis** and **zero basis**.

It is easy to obtain the **pole basis** of $\mathcal{L}(l\mathcal{O})$, which is $\{\phi_1, \phi_2, ..., \phi_l\} := \{1, x, y, x^2, xy, x^3, x^2y, .., x^iy^j \mid j = 0 \ or \ 1, \ 2i + 3j = l)\}$.

For each P_i, $1 \leq i \leq n$, we will find a **zero basis** $\{\psi_{j_3,P_i} : 1 \leq j_3 \leq l\}$ of $\mathcal{L}(l\mathcal{O})$ such that P_i is a zero of ψ_{j_3,P_i} with multiplicity (or at least) $j_3 - 1$. Consider the principle divisor

$$div(f_{m,P_i}) = mP_i + (-m \cdot P_i) - (m+1)\mathcal{O}.$$

If $m < l$, then $div(f_{m,P_i}) + l\mathcal{O}$ is effective, hence $div(f_{m,P_i}) \in \mathcal{L}(l\mathcal{O})$. Meanwhile, P_i is a zero of $div(f_{m,P_i})$ with multiplicity m. Set

$$\psi_{1,P_i} = 1, \psi_{2,P_i} = div(f_{1,P_i}), ..., \psi_{l,P_i} = div(f_{l-1,P_i}),$$

then for point P_i, we obtain a **zero basis** of $\mathcal{L}(l\mathcal{O})$. To compute the rational function f_{m,P_i} from the divisor $mP_i + (-m \cdot P_i) - (m+1)\mathcal{O}$, we can use the method described in [22] and Chapter 11 in [35]. Note that $\{\phi_i\}$ and $\{\psi_{j_3,P_i}\}$ are all the bases of vector spaces $\mathcal{L}(l\mathcal{O})$, so it is easy to get the set $\{a_{P_i,j_1,j_3} \in \mathbb{F}_q : 1 \leq i \leq n, 1 \leq j_1, j_3 \leq l\}$ such that for every i and every j_1, $\phi_{j_1} = \sum_{j_3} a_{P_i,j_1,j_3} \psi_{j_3,P_i}$ holds. The **Interpolation** step and the **Root finding** step just follow the the original algorithm shown in subsection 2.4.

We show an implementation for an elliptic code via Magma[19]. Here is an example.

The finite field is $\mathbb{F}_{127}$, The elliptic curve (over $\mathbb{F}_{127}$) is $\mathcal{E} : y^2 = x^3 - 3x + 72$. The order of $\mathcal{E}(\mathbb{F}_{127})$ is 137.

Let $P = (44, 65)$ be a random point of $\mathcal{E}$. Obviously $\langle P \rangle = \mathcal{E}(\mathbb{F}_{127})$. Let $\mathcal{O}$ the infinite point. Then $131P = \mathcal{O}$.

Set divisor $G := 4\mathcal{O}$, and divisor $D := P_1 + P_2 + \ldots + P_{20}$ with $supp(D) = \{P_1 = P, P_2 = (50, 9), P_3 = (49, 90), P_4 = (105, 83), \ P_5 = (74, 43), P_6 = (114, 94), \ P_7 = (120, 125), P_8 = (40, 43), \ P_9 = (112, 60), P_{10} = (36, 97), \ P_{11} = (10, 91), P_{12} = (126, 70), \ P_{13} = (108, 126), P_{14} = (2, 57), \ P_{15} = (14, 19), P_{16} = (46, 49), \ P_{17} = (90, 87), P_{18} = (7, 93), \ P_{19} = (54, 23), P_{20} = (36, 30)\}$. We get an $[n, k]$ Algebraic Geometric Code $\mathcal{C}[G, D]$ with $n = 20, k = 4$.

According to [13, 12], set $l := 31$ in the Guruswami-Sudan list decoding algorithm for above $[n, k] = [20, 4]$ elliptic codes. It is easy to see that

$$\{\phi_1, \phi_2, ..., \phi_{31}\} = \{1, x, y, x^2, xy, x^3, x^2y, .., x^{15}, x^{14}y\}$$

is a **pole basis** of $\mathcal{L}(31\mathcal{O})$.

For each point P_i, we can obtain a **zero basis** of $\mathcal{L}(l\mathcal{O})$ using follows Magma code:

```
for j:=1 to l do
T, ZB[j+1]:=IsPrincipal(j*Divisor(P_i)+Divisor((-j)P_i)-(j+1)*Divisor(O));
end for;
```

In this way, we obtain 20 zero-bases of of $\mathcal{L}(31\mathcal{O})$. For example, a **zero basis** for $P_{20} = (36, 30)$ is

$\{1,$

$x + 91,$

$y + 94x + 15,$

$80y + 38x^2 + 85x + 29,$

$(41x + 27)y + 88x^2 + 112x + 25,$

$(124x + 91)y + 7x^3 + 62x^2 + 86x + 45,$

$(x^2 + 48x + 75)y + 40x^3 + 74x^2 + 112x + 32,$

$(114x^2 + 26x + 99)y + 34x^4 + 29x^3 + 125x^2 + 107x + 92,$

$(30x^3 + 33x^2 + 86x + 113)y + 33x^4 + 28x^3 + 59x + 61,$

$(30x^3 + 99x^2 + 82x + 24)y + 114x^5 + 121x^4 + 119x^3 + 10x^2 + 89x + 78,$

$(8x^4 + 46x^3 + 41x^2 + 46x + 105)y + 66x^5 + 115x^4 + 122x^3 + 31x^2 + 15x + 11,$

$(35x^4 + 116x^3 + 55x^2 + 29x + 72)y + 91x^6 + 89x^5 + 18x^4 + 31x^3 + 100x^2 + 37x + 38,$

$(55x^5 + 49x^4 + 102x^3 + 72x^2 + 32x + 82)y + 119x^6 + 69x^5 + 55x^4 + 87x^3 + 125x^2 + 3x + 95,$

$(80x^5 + 20x^4 + 51x^3 + 51x^2 + 39x + 48)y + 117x^7 + 57x^6 + 71x^5 + 40x^4 + 90x^3 + 59x^2 + 103x + 73,$

$(106x^6 + 105x^5 + 17x^4 + 85x^3 + 92x^2 + 107x + 13)y + 59x^7 + 126x^6 + 34x^5 + 118x^4 + 5x^3 + 59x^2 + 9x + 83,$

$(83x^6 + 100x^5 + 56x^4 + 99x^3 + 7x^2 + 26x + 11)y + 15x^8 + 53x^7 + 39x^6 + 101x^5 + 80x^4 + 3x^3 + 27x^2 + 95x + 7,$

$(32x^7+109x^6+91x^5+16x^4+66x^3+32x^2+52x+54)y+63x^8+126x^7+26x^6+87x^5+40x^4+42x^3+109x^2+112x+40,$

$(119x^7 + 10x^6 + 111x^5 + 45x^4 + x^3 + 40x^2 + 53x + 26)y + 10x^9 + 56x^8 + 108x^7 + 80x^6 + 58x^5 + 56x^4 + 101x^3 +$
$123x^2 + 43x + 28,$

$(21x^8 + 61x^7 + 78x^6 + 58x^5 + 114x^4 + 28x^3 + 95x^2 + 54x + 45)y + 63x^9 + 111x^8 + 119x^7 + 9x^6 + 88x^5 + 123x^4 +$
$112x^3 + 44x^2 + 86x + 111,$

$(29x^8 + 21x^7 + 115x^6 + 75x^5 + 98x^4 + 13x^3 + 5x^2 + 21x + 1)y + 46x^{10} + 69x^9 + 80x^8 + 18x^7 + x^6 + 81x^5 + 60x^4 +$
$100x^3 + 126x^2 + 29,$

$(64x^9 + 106x^8 + 38x^7 + 30x^6 + 20x^5 + 110x^4 + 87x^3 + 61x^2 + 16x + 84)y + 79x^{10} + 15x^9 + 10x^8 + 9x^7 + 123x^6 +$
$104x^5 + 73x^4 + 73x^3 + 126x^2 + 65x + 103,$

$(57x^9 + 80x^8 + 41x^6 + 52x^5 + 102x^4 + 81x^3 + 5x^2 + 92x + 26)y + 45x^{11} + 86x^{10} + 114x^9 + 103x^8 + 95x^7 + 11x^6 +$
$58x^5 + 50x^4 + 3x^3 + 116x^2 + 48x + 34,$

$(114x^{10} + 3x^9 + 15x^8 + 117x^7 + 116x^6 + 72x^5 + 16x^4 + 57x^3 + 83x^2 + 31x + 118)y + 2x^{11} + 33x^{10} + 74x^9 + 20x^8 +$
$112x^7 + x^6 + 83x^5 + 78x^4 + 55x^3 + 69x^2 + 47x + 67,$

$(108x^{10} + 64x^8 + 14x^7 + 95x^6 + 67x^5 + 99x^4 + 113x^3 + 124x^2 + 35x + 101)y + 59x^{12} + 5x^{11} + 34x^{10} + 2x^9 + 33x^8 +$
$95x^7 + 112x^6 + 65x^5 + 69x^4 + 33x^3 + 119x^2 + 111x + 104,$

$(104x^{11} + 125x^{10} + 109x^9 + 23x^8 + x^7 + 32x^6 + 2x^5 + 90x^4 + 5x^3 + 7x^2 + 44x + 95)y + 7x^{12} + 103x^{11} + 74x^{10} +$
$88x^9 + 81x^8 + 83x^7 + 124x^6 + 116x^5 + 39x^4 + 91x^3 + 120x^2 + 29x + 39,$

$(25x^{11} + 41x^{10} + 58x^9 + 17x^8 + 77x^7 + 43x^6 + 90x^5 + 99x^4 + 109x^3 + 58x^2 + 30x + 14)y + 80x^{13} + 97x^{12} + 93x^{11} +$
$126x^{10} + x^9 + 66x^8 + 93x^7 + 60x^6 + 58x^5 + 112x^4 + 60x^3 + 29x^2 + 22x + 27,$

$(19x^{12} + 101x^{11} + 56x^{10} + 94x^9 + 121x^8 + 60x^7 + 88x^6 + 41x^5 + 42x^4 + 71x^3 + 25x^2 + 21x + 35)y + 4x^{13} + 29x^{12} +$
$101x^{11} + 119x^{10} + 81x^9 + 110x^8 + 122x^7 + 97x^6 + 46x^5 + 121x^4 + 51x^3 + 23x^2 + 15x + 75,$

$(45x^{12} + 34x^{11} + 30x^{10} + 65x^9 + 111x^8 + 11x^7 + 96x^6 + 62x^5 + 123x^4 + 59x^3 + 39x^2 + 82x + 94)y + 117x^{14} +$
$38x^{13} + 119x^{12} + 123x^{11} + 123x^{10} + 107x^9 + 122x^8 + 80x^7 + 23x^6 + 41x^5 + 112x^4 + 58x^3 + 120x^2 + 25x + 70,$

$(60x^{13} + 45x^{12} + 77x^{11} + 54x^{10} + 49x^9 + 123x^8 + 103x^7 + 51x^6 + 91x^5 + 90x^4 + 37x^3 + 82x^2 + 115x + 119)y +$
$77x^{14} + 62x^{13} + 20x^{12} + 58x^{11} + 44x^{10} + 24x^9 + 34x^8 + 59x^7 + 77x^6 + 75x^5 + 34x^4 + 99x^3 + 9x^2 + 25x,$

$(10x^{13} + 29x^{12} + +43x^{11} + 120x^{10} + 37x^9 + 114x^8 + 57x^7 + 53x^6 + 112x^5 + 94x^4 + 60x^3 + 47x^2 + 77x + 7)y + 49x^{15} +$
$40x^{14} + 112x^{13} + 78x^{12} + 30x^{11} + 116x^{10} + 5x^9 + 61x^8 + 39x^7 + 68x^6 + 28x^5 + 5x^4 + 108x^3 + 33x^2 + 62x + 29,$

$(47x^{14}+16x^{13}+81x^{12}+25x^{11}+36x^{10}+119x^9+107x^8+120x^7+30x^6+72x^5+28x^4+125x^3+95x^2+35x+117)y+60x^{15}+102x^{14}+x^{13}+85x^{12}+113x^{11}+59x^{10}+x^9+53x^8+108x^7+99x^6+13x^5+98x^4+60x^3+27x^2+122x+98,$
$(54x^{14}+118x^{13}+10x^{12}+108x^{11}+54x^{10}+120x^9+67x^8+118x^7+6x^6+65x^5+74x^4+16x^3+95x^2+82x+119)y+10x^{16}+20x^{15}+76x^{14}+x^{13}+54x^{12}+88x^{11}+6x^{10}+102x^9+74x^8+96x^7+73x^6+110x^5+76x^4+62x^3+106x^2+119x+15\}$

Now we assume that the received vector is

$$\boldsymbol{r}=(24,29,87,42,99,57,25,97,49,64,58,31,97,8,120,122,34,36,64,95).$$

Then we can construct a nonzero polynomial $H(T)\in\mathcal{L}(l\mathcal{O})[T]$ using the **pole basis**, all the **zero basis** for each P_i and $\boldsymbol{r}$, where
$H(T)=(52x^{14}+56x^{13}+44x^{12}+94x^{11}+107x^{10}+75x^9+96x^8+35x^7+23x^6+77x^5+118x^4+3x^3+61x^2+27x+89)y+112x^{16}+119x^{15}+83x^{14}+102x^{13}+4x^{12}+8x^{11}+111x^{10}+74x^9+13x^8+90x^7+33x^6+110x^5+51x^4+116x^3+111x^2+18x+77+((43x^{12}+93x^{11}+118x^{10}+92x^9+8x^8+61x^7+25x^6+91x^5+25x^4+88x^3+109x^2+119x+82)y+81x^{14}+47x^{13}+61x^{12}+47x^{10}+27x^9+50x^8+36x^7+55x^6+x^5+31x^4+60x^3+87x^2+65x+90)T+((81x^{10}+82x^9+90x^8+35x^7+114x^6+62x^5+124x^4+35x^3+29x^2+57x+10)y+22x^{12}+115x^{11}+124x^{10}+59x^9+104x^8+27x^7+112x^6+63x^5+113x^4+71x^3+122x^2+x+72)T^2+((76x^8+17x^7+78x^6+80x^5+106x^4+123x^3+71x^2+92x+23)y+5x^10+24x^9+45x^8+5x^7+46x^6+84x^5+87x^4+13x^3+96x^2+56x+19)T^3+((125x^6+59x^5+79x^4+80x^3+113x^2+3x+55)y+35x^8+74x^7+100x^6+49x^4+x^3+74x^2+124x+88)T^4+((37x^4+35x^3+37x^2+74x+36)y+119x^6+52x^5+125x^4+73x^3+119x^2+67x+52)T^5+((41x^2+49x+80)y+61x^4+5x^3+55x^2+44x+115)T^6+(59y+21x^2+119x+53)T^7.$

In the **Root finding** step, we obtain two roots of $H(T)$, which are $68+8x+23y+66x^2$ and $81+102x+100y+37x^2$ respectively. Consequently, the decoding results are

$$\boldsymbol{c}_1:=(24,67,87,90,99,72,25,43,49,112,78,85,97,8,91,122,52,36,64,95)$$

$$\boldsymbol{c}_2:=(24,29,46,42,38,57,91,97,49,64,58,31,97,37,120,81,34,97,84,95)$$

It is easy to verify that they are both valid codewords, and the distance of $\boldsymbol{c}_1$ and $\boldsymbol{r}$ is 7, and distance of $\boldsymbol{c}_2$ and $\boldsymbol{r}$ is 9. Meanwhile,

$$\boldsymbol{c}_1-\boldsymbol{c}_2=(0,38,41,48,61,15,61,73,0,48,20,54,0,98,98,41,18,66,107,0)$$

is a minimum-weight codeword.

4 New Approach to ECDLP

4.1 A Warm-Up

Let us first discuss the relation between ECDLP and minimum-weight codewords of EC code. Let $\mathcal{E}$ be an elliptic curve defined over $\mathbb{F}_q$, $\mathcal{E}(\mathbb{F}_q)$ be the elliptic curve group, $P_1,P_2,\ldots,P_n\in\mathcal{E}(\mathbb{F}_q)$, and G be a divisor of degree k. Let $\mathcal{C}(G,D)$ be the EC code determined by divisors G and $D=P_1+P_2+\ldots+P_n$. We know that the minimum distance d of $\mathcal{C}(G,D)$ is either $n-k$ or $n-k+1$. If $d=n-k+1$, then the EC code is a MDS code, otherwise the EC code is an *almost* MSD (AMDS) code. Whether $\mathcal{C}(G,D)$ is an MDS code or an AMDS code depends on whether there exist $\{P_{i_1},P_{i_2},\ldots,P_{i_k}\}\subseteq(P_1,P_2,\ldots,P_n)$ such that $P_{i_1}+P_{i_2}+\ldots+P_{i_k}-G$ is a principal divisor.

It is easy to find a codeword of minimum weight for a MDS-EC code. As for AMDS-EC code, we show that finding codewords of minimum weight is closely related to solving ECDLP problems (see the following theorem).

Theorem 4. *Let $\mathcal{E}$ be an elliptic curve over $\mathbb{F}_q$. Let P be a point over group $\mathcal{E}(\mathbb{F}_q)$. Suppose the order of P is a prime p. Let $n = \lceil \log_2 p \rceil$. For any point Q from subgroup $\langle P \rangle$, define an elliptic code $\mathcal{C}(G, D)$ with divisor $G := Q + (k-1)\mathcal{O}$ and divisor $D := P + 2P + \ldots + 2^{n-1}P$. If there exists an algorithm $\mathcal{A}$ who can find a codeword of minimum weight in $\mathcal{C}(G, D)$ with probability ϵ, then another algorithm $\mathcal{B}$ can be constructed to solve the ECDLP problem $s := \log_P Q$ with probability $\epsilon/(n+1)$.*

Proof. Suppose that $\mathcal{B}$ has an ECDLP instance $(\mathbb{F}_q, \mathcal{E}(\mathbb{F}_q), \ell, Q, P)$ where P is a generator of subgroup $\mathbb{G}$ of prime order p and $Q \in \mathbb{G}$. $\mathcal{B}$ aims to determine $s \in \mathbb{Z}_p$ such that $Q = sP$.

Express $s \in \mathbb{Z}_p$ with binary string $s = (s_1, s_2, \ldots s_n)$ with $n = \lceil \log_2 p \rceil$. $\mathcal{B}$ constructs an EC code to solve the ECDLP as follows.

Algorithm $\mathcal{B}$
Input: $(\mathbb{F}_q, \mathcal{E}(\mathbb{F}_q), p, Q, P)$
Output: s'

```
(1) $k \leftarrow \{0, 1, \ldots, n\}$.
(2) Define divisor $G = Q + (k-1)\mathcal{O}$ of degree $k$.
(3) Let $P_i = 2^{i-1}P$ for $i = 1, 2, \ldots, n$;
(4) Define divisor $D = P_1 + P_2 + \ldots + P_n$;
(5) Construct an EC code $\mathcal{C}(G, D)$;
(6) Invoke algorithm $\mathcal{A}$ to find a codeword $\boldsymbol{c} = (c_1, c_2, \ldots, c_n)$ of minimum
    weight for the EC code $\mathcal{C}(G, D)$.
(7) Suppose the nonzero components in $\boldsymbol{c}$ are $c_{i_1}$, $c_{i_2}$, …, $c_{i_k}$. Then compute
    $s' = \sum_{j=1}^{k} 2^{i_j - 1}$.
(8) Return($s'$)
```

Note that k is randomly chosen from $\{0, 1, \ldots, n\}$. Obviously, the probability that $k = \mathsf{wt}(s)$ is $1/(n+1)$.

Now we assume that the event $k = \mathsf{wt}(s)$ happens, or equivalently, the event that $\mathcal{C}(G, D)$ constructed by $\mathcal{B}$ is an AMDS-EC code happens. Since $s = (s_1, s_2, \ldots s_n)$, we have $Q = \sum_{i=1}^{n} s_i P_i$. If $s_{i_1} = s_{i_2} = \ldots = s_{i_k} = 1$ and $s_j = 0$ for $j \notin \{i_1, i_2, \ldots, i_k\}$, then $Q = \sum_{j=1}^{k} s_{i_j} P_{i_j}$.

Define a principal divisor $div(f) := P_{i_1} + P_{i_2} + \ldots + P_{i_k} - Q - (k-1)\mathcal{O}$. It is easy to see that $div(f) \in \mathcal{L}(G)$ since $div(f) + G$ is effective. Consequently, $\boldsymbol{c} := (f(P_1), f(P_2), \ldots, f(P_n))$ is a codeword of $\mathcal{C}(G, D)$, and $f(P_i) = 0$ iff $i \in \{i_1, i_2, \ldots, i_k\}$.

If $\mathcal{A}$ successfully outputs a codeword of minimum weight, then the codeword $\boldsymbol{c}$ must be of weight $n - k$. According to Theorem 2, the principal divisor $div(f)$ suggests that $P_{i_1} + P_{i_2} + \ldots + P_{i_k} - Q - (k-1)\mathcal{O}$ is $\mathcal{O}$ when "+" and "−" are implemented with the elliptic curve group operation. As a result, $Q = P_{i_1} + P_{i_2} + \ldots + P_{i_k}$ holds on the group of $\mathcal{E}(\mathbb{F}_q)$, hence $Q = \sum_{j=1}^{k} 2^{i_j - 1}P$.

In this way, $\mathcal{B}$ solves the ECDLP probem by invoking algorithm $\mathcal{A}$ with probability $\epsilon/(n+1)$. ▮

Note that in Section 3, we construct algorithm MinWeightCode which outputs codewords of minimum weight with probability $1 - \left(1 - \frac{\mu \cdot \binom{n-k}{u}}{\binom{n}{u}(q-1)^u}\right)^{T_m}$. Hence we can construct $\mathcal{B}$ to solve the ECDLP problem with probability

$$\frac{1}{n+1} \cdot \left(1 - \left(1 - \frac{\mu \cdot \binom{n-k}{u}}{\binom{n}{u}(q-1)^u}\right)^{T_m}\right).$$

Take the example in subsection 3.3. Let $P = P_1$, $Q = P_9$. Let $r_1 = 1$ and choose $r_2 = 29, r_3 = 93, r_4 = 49, r_5 = 5, r_6 = 98, r_7 = 54, r_8 = 10, r_9 = 103, r_{10} = 59, r_{11} = 15, r_{12} = 108$. Then $P_{10} = 29Q, P_{11} = 93Q, P_{12} = 49Q, P_{13} = 5Q, P_{14} = 98Q, P_{15} = 54Q, P_{16} = 10Q, P_{17} = 103Q, P_{18} = 59Q, P_{19} = 15Q, P_{20} = 108Q$. The output of MinWeiCodeword is the minimum-weight codeword

$$(0, 38, 41, 48, 61, 15, 61, 73, 0, 48, 20, 54, 0, 98, 98, 41, 18, 66, 107, 0).$$

Therefore, $P_1 + P_9 + P_{13} + P_{20} = \mathcal{O}$, which means $1 + s + 5s + 108s \equiv 0 \mod 137$. This immediately leads to a correct output of $s = 6$ since $Q = 6P$.

4.2 The Algorithm of Solving ECDLP

In the proof of the theorem in the previous subsection, algorithm $\mathcal{B}$ wins only if it correctly guesses the Hamming weight of $s (= \log_P Q)$. Hence the security reduction suffers from a security loss of factor $(n+1)$. In this subsection, we try to decrease the security loss factor. We have two observations.

(1) For a random $s \in \mathbb{Z}_p$, the hamming weight of s belongs to $\{0, 1, \ldots, \lceil \log_2 p \rceil)\}$ and it takes the value of $\lceil (\log_2 p + 1)/2 \rceil$ with the maximal probability.

(2) If we increase the number of elements in the support of D and add random elements in the support of divisor D, it is possible for us to improve the probability that $\exists i_1, i_2, \ldots, i_k$ such that $P_{i_1} + P_{i_2} + \ldots + P_{i_k} - G$ is principal divisor. Hence, the probability of $\mathcal{C}(G, D)$ being an AMDS-EC code will be greatly increased.

Based on the above observations, we present a probabilistic algorithm of solving ECDLP by constructing AMDS-EC codes and finding codewords of minimum weight with help of list decoding.

Let P be a point of prime order p in the group $\mathcal{E}(\mathbb{F}_q)$, where $\mathcal{E}$ is an elliptic curve defined over $\mathbb{F}_q$. Given P and $Q(= sP) \in \langle P \rangle$ and the parameter of $\mathcal{E}$, the following algorithm SolveECDLP aims to compute s $(= \log_P Q)$ by invoking MinWeiCodeword which aims to find minimum weight codeword of elliptic codes.

Algorithm SolveECDLP$(\mathcal{E}(\mathbb{F}_q), P, Q, p)$**:**
Input: An elliptic curve group $\mathcal{E}(\mathbb{F}_q)$, a generator P of prime order p, and an element $Q \in \langle P \rangle$.
Output: $s \in \mathbb{Z}_p$ (such that $Q = sP$).

```
1. Define $\theta := \lceil \log_2 p \rceil$, $n := 2\theta$, $k := \lfloor (\theta+1)/2 \rfloor$. If $\mathsf{wt}(p) = k$, then $k := \lfloor (\theta+1)/2 \rfloor + 1$.
2. Define divisor $G := k\mathcal{O}$ and Define $P_i := 2^{i-1}P$ for $i = 1, 2, \ldots, \theta$.
3. Randomly choose $r_2, r_3, \ldots, r_{n-\theta}$ from $\mathbb{Z}_p$. Set $r_1 := 1$ and define $P_{\theta+j} := r_j Q$ for $j = 1, 2, \ldots, n-\theta$.
4. Construct an Elliptic code $\mathcal{C}(G, D)$ where divisor $D = P_1 + P_2 + \ldots + P_n$.
5. Set $t_m = n - \sqrt{nk}$ and $T_m = O(poly(n))$.
6. Invoke $\boldsymbol{c} \leftarrow \mathsf{MinWeiCodeword}(\mathcal{C}(G, D), \{n-k\}, t_m, T_m)$;
7. If $\boldsymbol{c} = \perp$, goto Step 3.
8. If $\boldsymbol{c} \neq \perp$, then $\mathsf{wt}(c) = n - k$. Parse $\boldsymbol{c} = (c_1, c_2, \ldots, c_n)$. Suppose the zero components of $\boldsymbol{c}$ are $c_{i_1}, c_{i_2}, \ldots, c_{i_k}$.
9. Suppose $i_{j-1} \leq \theta$ and $i_j > \theta$, then compute $s' :\equiv -(r_{i_j-\theta} + r_{i_{j+1}-\theta} + \ldots + r_{i_k-\theta})^{-1}(2^{i_1-1} + 2^{i_2-1} + \ldots + 2^{i_{j-1}-1}) \mod p$.
10. If $Q = s'P$ then Return($s'$); else Return($\perp$).
```

In the above algorithm, it is possible for us to choose the parameters n and k flexibly as to optimize the algorithm.

4.3 Analysis of Algorithm SolveECDLP

Now we analyze the success probability of Algorithm SolveECDLP via the proof of the following theorem.

Theorem 5. *Let $\mathcal{E}$ be an elliptic curve over $\mathbb{F}_q$. Let P be a point over group $\mathcal{E}(\mathbb{F}_q)$. Suppose the order of P is a prime p. Let $\theta := \lceil \log_2 p \rceil$, $n := 2\theta$, $k := \lfloor (\theta+1)/2 \rfloor$, $u \leq (n-k)/2$ and $u \geq n - k - t_m$. then algorithm SolveECDLP successfully solves the ECDLP problem with probability*

$$\left(1 - \left(1 - \frac{\binom{\theta}{k-1}}{2^\theta}\right)^{n-\theta}\right) \cdot \left(1 - \left(1 - \frac{\lambda \cdot \binom{n-k}{u}}{\binom{n}{u}(q-1)^{u-1}}\right)^{T_m}\right) \cdot \left(1 - \frac{1}{p}\right),$$

where λ denotes the number of subsets $\mathcal{J} = \{i_1, i_2, \ldots, i_k\} \subseteq \{1, 2, \ldots, n\}$ such that $G - \sum_{j \in \mathcal{J}} P_j$ is a principal divisor.

Proof. Note that the Elliptic Code $\mathcal{C}(G,D)$ is an AMDS code iff there exists a principal divisor $div(f) := P_{i_1}+P_{i_2}+\ldots+P_{i_k}-k\mathcal{O}\in\mathcal{L}(G)$. Equivalently, there exist $P_{i_1},P_{i_2},\ldots,P_{i_k}$ such that $P_{i_1}+P_{i_2}+\ldots+P_{i_k}+k\mathcal{O}=\mathcal{O}$, i.e,

$$P_{i_1}+P_{i_2}+\ldots+P_{i_k}=\mathcal{O},$$

where the addition is over the elliptic group $\mathcal{E}(\mathbb{F}_q)$. There are three cases all together.

Case I: $i_1\le\theta$ **and** $i_k>\theta$**.** In this case, suppose that $i_{j-1}\le\theta$ and $i_j>\theta$, then $(2^{i_1-1}+2^{i_2-1}+\ldots+2^{i_{j-1}-1})P+(r_{i_j-\theta}+r_{i_{j+1}-\theta}+\ldots+r_{i_k-\theta})Q=\mathcal{O}$, that is, $-(r_{i_j-\theta}+r_{i_{j+1}-\theta}+\ldots+r_{i_k-\theta})s\equiv(2^{i_1-1}+2^{i_2-1}+\ldots+2^{i_{j-1}-1})\mod p$.

Case II: $i_1>\theta$**.** In this case, $(r_{i_1-\theta}+r_{i_2-\theta}+\ldots+r_{i_k-\theta})Q=\mathcal{O}$, i.e., $r_{i_1-\theta}+r_{i_2-\theta}+\ldots+r_{i_k-\theta}\equiv 0\mod p$.

Case III: $i_k\le\theta$**.** In this case, $(2^{i_1-1}+2^{i_2-1}+\ldots+2^{i_k-1})P=\mathcal{O}$, i.e., $(2^{i_1-1}+2^{i_2-1}+\ldots+2^{i_k-1})\equiv 0\mod p$. Recall that $\theta:=\lceil\log_2 p\rceil$, and $k=\lfloor\theta/2\rfloor+1$. Then $(2^{i_1-1}+2^{i_2-1}+\ldots+2^{i_k-1})<2p$.

Clearly, Case II happens with probability $1/p$ when r_ℓ's are randomly chosen, $\ell\in\{2,3,\ldots,n-\theta\}$, and Case III never happens since $\mathsf{wt}(p)\neq k$.

Now we consider the probability that $\mathcal{C}(G,D)$ is an AMDS code, when $s,r_2,\ldots r_{n-\theta}$ are randomly chosen from $\mathbb{Z}_p$.

$$\begin{aligned}
&\Pr[\mathcal{C}(G,D)\text{ is AMDS}]\\
&=\Pr[\exists i_1,\ldots,i_k\in[n]\text{ s.t. divisor }P_{i_1}+P_{i_2}+\ldots+P_{i_k}-G\text{ is principal}]\\
&=\Pr[\exists i_1,\ldots,i_k\in[n]\text{ s.t. }P_{i_1}+P_{i_2}+\ldots+P_{i_k}=\mathcal{O}]\,(\text{addition is over }\mathcal{E}(\mathbb{F}_q))\\
&=\Pr[\exists i_1,\ldots,i_k\in[n]\text{ s.t. Case I happens}]+\Pr[\exists i_1,\ldots,i_k\in[n]\text{ s.t. Case II happens}]\\
&\quad+\Pr[\exists i_1,\ldots,i_k\in[n]\text{ s.t. Case III happens}]\\
&=1/p+\Pr[\exists i_1,\ldots,i_k\in[n]\text{ s.t. Case II happens}]+0 \qquad (7)\\
&\ge\Pr[\exists i_1,\ldots,i_k\text{ s.t. Case II happens}]\\
&=\Pr\left[\begin{matrix}\exists i_1,\ldots,i_k\in[n]\\ \exists i_{j-1}\le\theta,i_j>\theta\end{matrix}:-(r_{i_j-\theta}+\ldots+r_{i_k-\theta})s=(2^{i_1-1}+\ldots+2^{i_{j-1}-1})\mod p\right]\\
&\ge\Pr\left[\begin{matrix}\exists i_1,\ldots,i_k\in[n]\\ \exists i_k>\theta\end{matrix}:-r_{i_k-\theta}s=(2^{i_1-1}+\ldots+2^{i_{k-1}-1})\mod p\right]\\
&=1-\Pr\left[\nexists i_k,i_k\in[n],i_k>\theta\text{ s.t. }-r_{i_k-\theta}s=(2^{i_1-1}+\ldots+2^{i_{k-1}-1})\mod p\right]\\
&=1-\left(1-\frac{\binom{\theta}{k-1}}{2^\theta}\right)^{n-\theta}. \qquad (8)
\end{aligned}$$

Given that $\mathcal{C}(G,D)$ is an AMDS elliptic code, then the minimum distance of $\mathcal{C}(G,D)$ is $d=n-k$. According to (6), $\mathsf{MinWeiCodeword}(\mathcal{C}(G,D),\{n-k\},t_m,T_m)$ successfully outputs a codeword $\boldsymbol{c}=(c_1,c_2,\ldots,c_n)$ of minimum weight with probability

$$1 - \left(1 - \frac{\lambda \cdot \binom{n-k}{u}}{\binom{n}{u}(q-1)^{u-1}}\right)^{T_m}.$$

Suppose that the zero components of the minimum weight codeword $\boldsymbol{c}$ are given by $c_{i_1}, c_{i_2}, \ldots, c_{i_k}$. Then it must hold that $P_{i_1} + P_{i_2} + \ldots + P_{i_k} = \mathcal{O}$. Similarly, there are three cases: $i_1 \leq \theta$ and $i_k > \theta$; $i_1 > \theta$; $i_k \leq \theta$. As analyzed before, the second case happens with probability $1/p$ and the third case never happens. Therefore, the first case happens with probability $1 - 1/p$. Meanwhile the first case means that $\exists i_{j-1} \leq \theta, i_j > \theta$, so that

$$(2^{i_1-1} + 2^{i_2-1} + \ldots + 2^{i_{j-1}-1})P + (r_{i_j-\theta} + r_{i_{j+1}-\theta} + \ldots + r_{i_k-\theta})Q = \mathcal{O},$$

i.e.,

$$s \equiv -(r_{i_j-\theta} + r_{i_{j+1}-\theta} + \ldots + r_{i_k-\theta})^{-1}(2^{i_1-1} + 2^{i_2-1} + \ldots + 2^{i_{j-1}-1}) \mod p.$$

In this case, Algorithm SolveECDLP successfully solves the ECDLP problem.

Consequently,

$$\begin{aligned}
& \Pr[\text{SolveECDLP succeeds}] && (9)\\
= & \Pr[\mathcal{C}(G,D) \text{ is AMDS} \wedge \text{MinWeiCodeword succeeds} \wedge \text{Case I happens}] && (10)\\
= & \Pr[\mathcal{C}(G,D) \text{ is AMDS}] && (11)\\
& \cdot \Pr[\text{MinWeiCodeword succeeds} \mid \mathcal{C}(G,D) \text{ is AMDS}] && (12)\\
& \cdot \Pr[\text{Case I happens} \mid \text{MinWeiCodeword succeeds}, \mathcal{C}(G,D) \text{ is AMDS}] && (13)\\
= & \left(1 - \left(1 - \frac{\binom{\theta}{k-1}}{2^\theta}\right)^{n-\theta}\right) \cdot \left(1 - \left(1 - \frac{\lambda \cdot \binom{n-k}{u}}{\binom{n}{u}(q-1)^{u-1}}\right)^{T_m}\right) \cdot (1 - \frac{1}{p}), && (14)
\end{aligned}$$

where λ denotes the number of subsets $\mathcal{J} = \{i_1, i_2, \ldots, i_k\} \subseteq \{1, 2, \ldots, n\}$ such that $G - \sum_{j \in \mathcal{J}} P_j$ is a principal divisor. ▮

Remark. The probability of (8) only shows a lower bound of the probability that $\mathcal{C}(G,D)$ is an AMDS code. Even if it is only a lower bound, (8) is already close to 1 (as compared with the loss factor $1/(n+1)$ in the previous subsection). For instance, now we choose the ECC curve **ECCp-131** over a prime field of 131-bit to construct a EC code, then $\theta = 131, k = 66$. Take $n = 262$, then the probability is at least 0.99992.

Recall that in algorithm MinWeiCodeword$(\mathcal{C}(G,D), \{n-k\}, t_m, T_m)$, there are T_m times of invocations of FindCodeword$(\mathcal{C}(G,D), u, d, t_m)$. For each invocation, FindCodeword succeeds in finding a codeword of minimum weight via the

Guruswami-Sudan list decoding with probability

$$\Pr\left[\hat{\boldsymbol{c}} \leftarrow \mathsf{FindCodeword}(\mathcal{C}, u, d, t_m) : \mathsf{wt}(\hat{\boldsymbol{c}}) = d\right] \approx \frac{\lambda \cdot \binom{d}{u}}{\binom{n}{u}(q-1)^{u-1}}. \tag{15}$$

Therefore, the times T_m of invocations of FindCodeword should be of order

$$\frac{\binom{n}{u}(q-1)^{u-1}}{\lambda \cdot \binom{d}{u}} \tag{16}$$

for MinWeiCodeword to succeed. Algorithm FindCodeword is dominated by the list decoding algorithm. Recall that list decoding algorithms, either the Guruswami-Sudan or Shokrollahi-Wasserman algorithm [30], are polynomial-time algorithms in the codeword length. However, the probability (15) is too small to make T_m a polynomial. Therefore, the algorithm SolveECDLP is not efficient, and it is even not of square-root time algorithm. To decrease the computational complexity of SolveECDLP, a possible way is to increase the error bound t_m of the list decoding. Recall that $u + t = d$ and $t \leq t_m$. A larger t_m implies that we can take a small value of u, so the probability in (15) will be improved which in turn to decrease the computational complexity of SolveECDLP. On the other hand, for a concrete security parameter κ such that $q \approx 2^\kappa$, a more efficient list decoding algorithm will also help us to improve the efficiency of SolveECDLP.

5 Conclusion

We proposed a new method to solve the ECDLP problem. For any ECDLP, we first construct an Elliptic Code, then resort to techniques of List Decoding to find codewords of minimum weight. With such a codeword, we are able to solve the ECDLP problem. Our method of solving ECDLP is still not efficient enough, due to the small probability of finding minimum-weight codeword. Nevertheless, this is a totally new approach and we believe the efficiency can be improved with the new development of List Decoding.

References

[1] P. Beelen and K. Brander, Efficient list decoding of a class of algebraic-geometrycodes. Advances in Mathematics of Communications, Vol. 4, No. 4, 2010, pp.485-518.

[2] E. R. Berlekamp, R. J. McEliece, and H. C. van Tilborg, On the inherent intractability of certain coding problems, IEEE Trans. Inf. Theory, vol. IT-24, no. 3, pp. 384-386, 1978.

[3] Certicom. Certicom ECC Challenge. http://www.certicom.com/images/pdfs/cert ecc challenge.pdf, 1997.

[4] Q. Cheng, Hard problems of algebraic geometry codes, IEEE Transactions on Information Theory, vol. 54, pp. 402-406, 2008.

[5] Q. Cheng, D. Wan, On the list and bounded distance decodability of Reed-Solomon codes (extended abstract), in: FOCS, 2004, pp. 335-341.

[6] Driencourt Y., Michon J.F.: Elliptic codes over fields of characteristics 2. J. Pure Appl. Algebra 45(1), 15-39 (1987)

[7] I. Dumer, D. Micciancio, and M. Sudan, Hardness of approximating the minimum distance of a linear code, IEEE Trans. Inf. Theory, vol. 49, no. 1, pp. 22-37, 2003.

[8] P. Elias, List decoding for noisy channels, in: 1957-IRE WESCON Convention Record, 1957, pp. 94-104.

[9] S.D., Galbraith and P. Gaudry, Recent progress on the elliptic curve discrete logarithm problem, Des. Codes Cryptogr. (2016) pp.78-51.

[10] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing, pages 25-32, 1989.

[11] V.D. Goppa, Codes on algebraic curves, Soviet Math. Dokl. **24** (1981) No. 1, 170-172.

[12] V. Guruswami and M. Sudan, Improved decoding of reed-solomon and algebraic-geometry codes, IEEE Trans. Inf. Theory, vol. 45, no. 6, pp. 1757-1767, 1999.

[13] V. Guruswami and M. Sudan, On Representations of Algebraic-Geometric Codes for List Decoding, IEEE Trans. Inf. Theory, vol. 47, no. 4, pp. 1610-1613, 2001.

[14] V. Guruswami and A. Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. IEEE Transactions on Information Theory, 54(1):135-150, 2008.

[15] V. Guruswami and C. Xing. List decoding reed-solomon, algebraic-geometric, and gabidulin subcodes up to the singleton bound. In Proceedings of the 45th annual ACM symposium on Theory of Computing (STOC), pages 843-852. ACM, 2013.

[16] T. HøHOLDT, J. H. VAN LINT AND R. PELLIKAAN. Algebraic Geometry Codes. In Handbook of Coding Theory, (V.S. Pless, W.C. Huffamn and R.A. Brualdi Eds.), Elsevier

[17] A. Kiayias and M. Yung, Cryptographic Hardness based on the Decoding of Reed Solomon Codes, In the Proceedings of the 29th International Colloquium in Algorithms, Languages and Programming, 2002.

[18] N. Koblitz, Elliptic curve cryptosystems, Mathematics of Computation, 48, pp. 203-209, 1987.

[19] MAGMA Computational Algebra System, http://magma.maths.usyd.edu.au/magma/

[20] R. J. McEliece. On the average list size for the Guruswami-Sudan decoder. In 7th International Symposium on Communications Theory and Applications (ISCTA), July 2003.

[21] V. Miller, Use of elliptic curves in cryptography, Advances in Cryptology-Crypto'85, LNCS 218, pp. 417-426, New York: Springer-Verlag, 1986.

[22] V. Miller. Short programs for functions on curves. unpublished manuscript, 1986.

[23] C. Moreno, Algebraic curves over finite fields, Cambridge Tracts in Mathematics, Cambridge University Press, No. 97 (1991).

[24] Parvaresh, F., Vardy, A. Correcting Errors Beyond the Guruswami-Sudan Radius in Polynomial Time. In: 46th Annual IEEE Symposium on Foundations of Computer Science. pp. 285-294 (2005)

[25] J. M. Pollard, "Monte Carlo methods for index computation mod p", Mathematics of Computation, 32, pp. 918-924, 1978.

[26] M. Sudan, Decoding of Reed Solomon Codes Beyond the Error-Correction Bound, Journal of Complexity, vol. 13, pp. 180-193, 1998.

[27] M. Sudan. List decoding: Algorithms and applications. Lecture Notes in Computer Science, 2013, 31(1):25-41.

[28] Joseph H. Silverman, John Tate, Rational Points on Elliptic Curves, Springer-Verlag 1992

[29] M. Amin Shokrollahi, Minimum Distance of Elliptic Codes, Advances in Mathematics, 93, pp.251-281, 1992.

[30] M. Amin Shokrollahi and H. Wasserman, List Decoding of Algebraic-Geometric Codes, IEEE Transaction of Information Theory, Volume 45, No. 2, 1999.

[31] H. Stichtenoth, Algebraic Function Field and Codes, Springer-Verlag 1993

[32] J.H. Silverman, *The arithmetic of elliptic curves.* Springer-Verlag, New York, 1986.

[33] M.A. Tsfasman and S.G. Vlădut, Algebraic-geometric codes, Kluwer Academic Publishers, 1991.

[34] A. Vardy, The intractability of computing the minimum distance of a code, IEEE Trans. Inf. Theory, vol. 43, no. 6, pp. 1757-1766, 1997.

[35] L. Washington, Elliptic Curves: Number Theory and Cryptography, Chapman and Hall/CRC, 2003

[36] J. M. Wozencraft, List decoding, Quarterly Progress Report, Research Laboratory of Electronics, MIT, 1958, 48 pp. 90-95.

[37] I. A. Semaev. Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p. Math. Comp., 67(221):353-356, 1998

[38] N.P.Smart, The discrete logarithm problem on elliptic curves of trace one. Journal of Cryptology, 12(3):193-196,1999.

[39] T. Satoh and K. Araki, Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves, Comm. Math. Pauli, Vol.47, No.1, pp81-92, 1998.

[40] A. Menezes, T. Okamoto and S. Vanstone, Reducing elliptic curve logarithms to logarithms in a finite field, IEEE Trans on Information Theory , 1993. 39(2). pp. 1639-1646.

[41] G. Frey and H.Rck, A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves, Mathematics of Computation, 62, pp.865-874, 1994.

[42] P. Gaudry, F. Hess, and N. P. Smart. Constructive and destructive facets of Weil descent on elliptic curves. Journal of Cryptology, 15(1):19-46, 2002.