格子密码

Lattice-based Cryptography

刘卓

格子密码是在结构本身或在安全证明中涉及格 (Lattice) 的加密图元构造的通用术语。基于晶格的构造目前是后量子密码学的重要候选者。

1 格

定义 1. 让 $\mathbf{v_1}, \mathbf{v_2}, \cdots, \mathbf{v_n}$ 是在 n 维欧几里得空间 (Euclidean space) R^n 中的独立向量 (independent vectors)。格 (lattice) 记作 L，它是由 $\{\mathbf{v_1}, \mathbf{v_2}, \cdots, \mathbf{v_n}\}$ 的线性组合的集合，$\mathbf{v_1}, \mathbf{v_2}, \cdots, \mathbf{v_n}$ 的系数为整数:

$$L = \{a_1\mathbf{v_1} + a_2\mathbf{v_2} + \cdots + a_n\mathbf{v_n}, \mathbf{a_i} \in \mathbb{Z}\}$$

备注:L 的基是生成 L 的 n 个独立向量的任意集合。格子有无穷多个基。

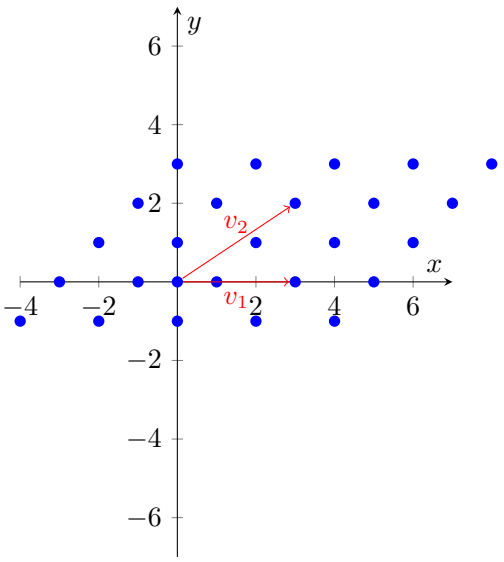
定义 2. 设 L 是维数为 n 的格，B $= \{\mathbf{v_1}, \mathbf{v_2}, \cdots, \mathbf{v_n}\}$ 是 L 的一个基。L 的基本域 (或基本平行六面体) 对应于这个基 B 是集合:

$$\mathbb{F}(\mathbf{v_1}, \mathbf{v_2}, \cdots, \mathbf{v_n}) = \{t_1\mathbf{v_1} + t_2\mathbf{v_2} + \cdots + t_n\mathbf{v_n}, 0 \le t_i < 1\}$$

引理 1.1. 设 $L \subset \mathbb{R}^{\times}$ 是 n 维的格，F 是 L 的一个基本域。则每一个向量 $\mathbf{w} \in \mathbb{R}^{\times}$ 可以写作:

$$\mathbf{w} = \mathbf{t} + \mathbf{v}$$

有唯一的 $\mathbf{t} \in \mathbb{F}$ 和有唯一的 $\mathbf{v} \in L$。



寻找格规约算法 (An algorithm for finding reduced basis):

1. while $\|\mathbf{b_1}\| > \|\mathbf{b_2}\|$
2. swap $\mathbf{b}_1, \mathbf{b}_2$
3. $u := \left\lceil \frac{b_1 \cdot b_2}{b_1 \cdot b_1} \right\rfloor$ ($[u]$ 取最接近的整数)
4. if $u = 0$ then return $(\mathbf{b}_1, \mathbf{b}_2)$
5. else $\mathbf{b}_2 := \mathbf{b}_2 - u \cdot \mathbf{b}_1$

例 1. $\mathbf{b}_1 = (90, 123), \mathbf{b}_2 = (56, 76)$
寻找格规约 (reduced basis)

解. 第一步:

$$\begin{matrix} \|b_1\| = 3\sqrt{2581} \\ \|b_2\| = 4\sqrt{557} \end{matrix} \quad \|b_1\| > \|b_2\|$$

交换 $\mathbf{b}_1, \mathbf{b}_2$, $\mathbf{b}_1 = (56, 76), \mathbf{b}_2 = (90, 123)$

$$u := \left\lceil \frac{b_1 \cdot b_2}{b_1 \cdot b_1} \right\rfloor = \left\lceil \frac{56 \cdot 90 + 76 \cdot 123}{56^2 + 76^2} \right\rfloor = \left\lceil \frac{3597}{2228} \right\rfloor = 2 \neq 0$$

$$\mathbf{b}_2 = (90, 123) - 2(56, -76) = (-22, -29)$$

第二步:
$\mathbf{b}_1 = (56, 76), \mathbf{b}_2 = (-22, -29)$

$$\begin{matrix} \|b_1\| = 4\sqrt{557} \\ \|b_2\| = 5\sqrt{53} \end{matrix} \quad \|b_1\| > \|b_2\|$$

交换 $\mathbf{b}_1, \mathbf{b}_2$,$\mathbf{b}_1 = (-22, -29), \mathbf{b}_2 = (56, 76)$

$$u := \left\lceil \frac{b_1 \cdot b_2}{b_1 \cdot b_1} \right\rfloor = \left\lceil \frac{(-22) \cdot 56 + (-29) \cdot 76}{(-22)^2 + (-29)^2} \right\rfloor = \left\lceil -\frac{3436}{1325} \right\rfloor = -3 \neq 0$$

$$\mathbf{b}_2 = \mathbf{b}_2 + 3\mathbf{b}_1 = (-10, -11)$$

第三步:
$\mathbf{b}_1 = (-22, -29), \mathbf{b}_2 = (-10, -11)$

$$\begin{matrix} \|b_1\| = 5\sqrt{53} \\ \|b_2\| = \sqrt{221} \end{matrix} \quad \|b_1\| > \|b_2\|$$

交换 $\mathbf{b}_1, \mathbf{b}_2$,$\mathbf{b}_1 = (-10, -11), \mathbf{b}_2 = (-22, -29)$

$$u := \left\lceil \frac{b_1 \cdot b_2}{b_1 \cdot b_1} \right\rfloor = \left\lceil \frac{539}{221} \right\rfloor = 2 \neq 0$$

$$\mathbf{b}_2 = \mathbf{b}_2 - 2\mathbf{b}_1 = (-2, -7)$$

第四步：

$\mathbf{b}_1 = (-10, -11), \mathbf{b}_2 = (-2, -7)$

$$\begin{matrix} \|b_1\| = \sqrt{221} \\ \|b_2\| = \sqrt{53} \end{matrix} \quad \|b_1\| > \|b_2\|$$

交换 $\mathbf{b}_1, \mathbf{b}_2$,$\mathbf{b}_1 = (-2, -7), \mathbf{b}_2 = (-10, -11)$

$$u := \left\lceil \frac{b_1 \cdot b_2}{b_1 \cdot b_1} \right\rfloor = \left\lceil -\frac{97}{53} \right\rfloor = 1.8 = 2 \neq 0$$

$$\mathbf{b}_2 = \mathbf{b}_2 - 2\mathbf{b}_1 = (-6, 3)$$

第五步：

$\mathbf{b}_1 = (-2, -7), \mathbf{b}_2 = (-6, 3)$

$$\begin{matrix} \|b_1\| = \sqrt{53} \\ \|b_2\| = \sqrt{39} \end{matrix} \quad \|b_1\| > \|b_2\|$$

交换 $\mathbf{b}_1, \mathbf{b}_2$,$\mathbf{b}_1 = (-6, 3), \mathbf{b}_2 = (-2, -7)$

$$u := \left\lceil \frac{b_1 \cdot b_2}{b_1 \cdot b_1} \right\rfloor = \left\lceil \frac{-9}{39} \right\rfloor = 0$$

结束，返回 $\mathbf{b}_1 = (-6, 3), \mathbf{b}_2 = (-2, -7)$

□

2 最短向量问题

最短向量问题 (The Shortest Vector Problem,SVP) 是指在格 L 中找到一个最短的非零向量，即找到一个非零向量 $v \in L$，使欧几里得范数 $||\mathbf{v}|| = \sqrt{v_1^2 + v_2^2 + \cdots + v_n^2}$ 最小。

最近向量问题 (The Closest Vector Problem ,CVP) 是一个向量 $\mathbf{w} \in \mathbb{R}$ 不在 L 中，然后找到一个向量 $\mathbf{v} \in \mathbf{L}$ 与 $\mathbf{w}$ 最接近。他们之间的欧氏距离范数为 $||\mathbf{w} - \mathbf{v}||$。

SVP 和 CVP 都是深奥的问题，随着格的维度的增加，这两个问题的计算都变得越来越困难。另一方面，SVP 和 CVP 的近似解在纯数学和应用数学的不同领域都有令人惊讶的应用。

如果维数 n 很大，例如 $n > 100$ 那么已知的算法在求解 SVP 时并不是很有效，但 CVP 问题却很容易。这允许在密码结构中使用格，几种基于格的密码系统已经被提出。

3 Goldreich, Goldwasser 和 Halevi (GGH) 公钥密码系统

1. *Alice* 在整数域选择一个基 (basis) $\mathbf{v} = \{\mathbf{v_1}, \mathbf{v_2}, \cdots, \mathbf{v_n}\}$。设 $\mathbf{V}$ 是矩阵它的行是向量 v_i，L 是向量 v_i 生成的格；

2. *Alice* 选择一个 $n \times n$ 的矩阵 U,U 都是整数项，并且 $det(U) = \pm 1$。计算 $W = UV$。然后公开 W 的行向量 $\mathbf{w_i}$；

3. 假设 *Bob* 想要发送信息 $\mathbf{m} = (m_{1,2}, \cdots, m_n)$，将 $\mathbf{m}$ 转化为一个整数域的向量。然后随机选择一个小向量 $\mathbf{r}$，计算

$$\mathbf{e} = m_1\mathbf{w_1} + m_2\mathbf{w_2} + \cdots + m_n\mathbf{w_n} + \mathbf{r}$$

将 $\mathbf{e}$ 发送给 *Alice* ；

4. *Alice* 解决最近向量问题 (CVP)，找到一个向量 $\mathbf{d} \in L$ 与 $\mathbf{e}$ 最接近。然后计算

$$\mathbf{d} \cdot W^{-1} = (m \cdot W + r)\, W^{-1} = m \cdot U \cdot W \cdot W^{-1} + r \cdot W^{-1} = m \cdot U + r \cdot W^{-1}$$

$\mathbf{m} = \mathbf{m} \cdot U \cdot U^{-1}$；

例 2. 令

$$V = \begin{pmatrix} 11 & 0 \\ 0 & 7 \end{pmatrix} \qquad U = \begin{pmatrix} 7 & 5 \\ 4 & 3 \end{pmatrix}$$

使用 $\mathbf{r} = (1, 1)$ 加密信息 $\mathbf{m} = (12, 13)$

解. 验证 $det(U) = 7 \cdot 3 - 4 \cdot 5 = 1$ 满足条件，可以使用

$$V^{-1} = \begin{pmatrix} \frac{1}{11} & 0 \\ 0 & \frac{1}{7} \end{pmatrix} \qquad U^{-1} = \begin{pmatrix} 3 & -5 \\ -4 & 7 \end{pmatrix}$$

$$W = UV = \begin{pmatrix} 7 & 5 \\ 4 & 3 \end{pmatrix} \begin{pmatrix} 11 & 0 \\ 0 & 7 \end{pmatrix} = \begin{pmatrix} 77 & 35 \\ 44 & 21 \end{pmatrix}$$

$$\mathbf{e} = \mathbf{m}W + \mathbf{r} = (1497, 694)$$

Alice 收到 $\mathbf{e}$ 后进行解密：

$$\mathbf{e}V^{-1} = (1497, 694) \begin{pmatrix} \frac{1}{11} & 0 \\ 0 & \frac{1}{7} \end{pmatrix} = \left(\frac{1497}{11}, \frac{694}{7} \right) \approx (136, 99)$$

$$\mathbf{m} = (136, 99)U^{-1} = (136, 99) \begin{pmatrix} 3 & -5 \\ -4 & 7 \end{pmatrix} = (12, 13)$$

□

NTRU 公钥密码系统是由 Hoffstein, Pipher 和 Silverman 发明。由一种多项式的卷积所构成。

多项式卷积算法是

$$f * g = \sum_{k=0}^{N-1} c_k x^k$$

其中

$$c_k = \sum_{i+j\equiv k \pmod N} f_i g$$

例 3. 计算当 $N = 3$ 时，$f * g$, 其中

$$f = \sum_{i=0}^{N-1} f_i x^i = x^2 + 4x - 1$$
$$g = \sum_{i=0}^{N-1} g_i x^i = 5x^2 - x + 3$$

解. f 的多项式为 $f = [f_0, f_1, f_2] = [-1, 4, 1]$
g 的多项式为 $g = [g_0, g_1, g_2] = [3, -1, 5]$
$c = f * g$ 的多项式是

$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} f_0 & f_2 & f_1 \\ f_1 & f_0 & f_2 \\ f_2 & f_1 & f_0 \end{bmatrix} \begin{bmatrix} g_0 \\ g_1 \\ g_2 \end{bmatrix} = \begin{bmatrix} -1 & 1 & 4 \\ 4 & -1 & 1 \\ 1 & 4 & -1 \end{bmatrix} \begin{bmatrix} 3 \\ -1 \\ 5 \end{bmatrix} = \begin{bmatrix} 16 \\ 18 \\ -6 \end{bmatrix}$$

$$f * g = 16 + 18x - 6x^2$$

因此通常情况 $f * g$ 通项是:

$$\begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{N-1} \end{bmatrix} = \begin{bmatrix} f_0 & f_{N-1} & f_{N-2} & \cdots & f_1 \\ f_1 & f_0 & f_{N-1} & \cdots & f_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f_{N-1} & f_{N-2} & f_{N-3} & \cdots & f_0 \end{bmatrix} \begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_{N-1} \end{bmatrix} = \begin{bmatrix} p_0 \\ p_1 \\ \vdots \\ p_{N-1} \end{bmatrix}$$

□

3.1 NTRU

NTRU 是一种非对称密码系统，它基于格上的最短向量问题。1996 年由 Hoffstein, piphher 和 Silverman 发现。2009 年，NTRU 密码系统已经通过了电气和电子工程师协会 (IEEE) 的标准化认证。与 RSA 和 ECC 相比，这是一种更有效的加密和解密方法: 更快的密钥生成 (使用一次性密钥) 和低内存使用 (因此它可以用于移动设备和智能卡)。

NTRU 的消息为多项式形式，其系数为整数 mod p。

Alice 作为发送者以及 *Bob* 接受者想使用 NTRU，那么他们该如何使用呢?

1. *Alice* 和 *Bob* 共同决定一个指数 N，使得多项式的次数最多为 $N-1$，系数为整数。以及再共同决定两个整数 p, q，使得 $gcd(p,q)=1$;

2. *Bob* 选择两个多项式

$$f = \sum_{k=0}^{N-1} f_i x^i$$

$$g = \sum_{k=0}^{N-1} g_i x^i$$

然后计算 $f_p := f^{-1} \pmod p$ 和 $f_q := f^{-1} \pmod q$，保留 f_p, f_q;

3. *Bob* 计算 $h = f_q * g \pmod q$，并公开;

4. *Alice* 把信息转换成一个多项式 m, 其系数是整数介于 $-p/2$ 和 $p/2 \bmod p$ 之间;

5. 然后 *Alice* 随机选择另一个小的多项式 r(以掩盖信息)。保留 r 不公开，然后将 $e = pr * h + m \pmod q$ 发送给 *Bob*;

6. *Bob* 为了还原信息，他需要计算 $a = f * e \pmod q$ 和 $b = a \pmod p$;

7. 最后，*Bob* 计算 $f_p * b \pmod p$ 还原信息;

注意 $f * f_p = 1 \pmod p, f * f_q = 1 \pmod q, h = f_q * g \pmod q$

$$a = f * e = f * (pr * h + m) = pr * (f * h) + f * m = pr * g + f * m \pmod q$$

$$b = a \pmod p = pr * g + f * m \pmod p = f * m \pmod p$$

因此，

$$f_p * b = f_p * f * m = m \pmod p$$

还原成功。

例 4. 密钥生成 *Alice* 和 *Bob* 共同决定了 $N = 7, p = 3, q = 41, d = 2$。

1. *Bob* 选择了两个多项式:

$$f(x) = x^6 - x^4 + x^3 + x^2 - 1$$

$$g(x) = x^6 + x^4 - x^2 - x$$

2.

$$Fq(x) = f(x)^{-1} \pmod q = 8x^6 + 26x^5 + 31x^4 + 21x^3 + 40x^2 + 2x + 37 \pmod{41}$$

$$Fp(x) = f(x)^{-}1 \pmod p = x^6 + 2x^5 + x^3 + x^2 + x + 1 \pmod 3$$

保留为私钥;

3.

$$h(x) = p * (Fq) * g \pmod q = 20x^6 + 40x^5 + 2x^4 + 38x^3 + 8x^2 + 26x + 30 \pmod{41}$$

并公开;

4. 假设 *Alice* 把信息转化为 $m(x) = -x^5 + x^3 + x^2 - x + 1$ 并用 $r(x)x^6 - x^5 + x - 1$ 进行加密

$$e(x)\ 31x^6 + 19x^5 + 4x^4 + 2x^3 + 40x^2 + 3x + 25 \pmod{41}$$

5. *Bob* 计算

$$a = f * e \pmod{q} = x^6 + 10x^5 + 33x^4 + 40x^3 + 40x^2 + x + 40 \pmod{41}$$

$$b = a \pmod{p} = x^6 + 10x^5 - 8x^4 - x^3 - x^2 + x - 1 \pmod{3}$$

6. *Bob* 计算 $f_p * b \pmod{p} = 2x^5 + x^3 + x^2 + 2x + 1 \pmod{3} = m(x) = -x^5 + x^3 + x^2 - x + 1$ ，还原成功

□

当 $N = 503, p = 3, q = 256$，就拥有非常高的安全性了。

4 Zero Knowledge Protocol

假设有一个双面隧道，隧道两边有一扇门。*Alice* 想向 *Bob* 证明她能开门。然而，她不想让 *Bob* 知道她是怎么做到的。该如何做呢?

1. *Bob* 秘密地随机选择 k 个命令序列，“对”、“错”、“左”、“右” 之类的;
2. *Alice* 进入隧道 (*Bob* 没有看到她从哪边进入)，并站在门中等待 *Bob* 的命令;
3. *Bob* 发出命令，并观察 *Alice* 离开隧道的一侧;
4. 如果 *Alice* 每次都能从正确的一边出来，那么 *Bob* 就接受她可以打开门。