文章编号:1000-2367(2020)05-0001-08 DOI:10.16366/j.cnki.1000-2367.2020.05.001

PBKZ 算法及其在格挑战中的应用

孙明豪,屈龙江,李超

(国防科技大学 数学系,长沙 410073)

摘 要:格是欧氏空间 $\mathbf{R}^n$ 中的离散加法子群,格上的许多计算问题被证明是 NP-hard,常用来作为公钥密码体制的底层困难问题.目前基于量子计算机模型的量子算法也难以高效求解格上的困难问题,因此后量子时代下格密码学受到了越来越多的关注.最短向量问题(Shortest Vector Problem,SVP)是格上的计算困难问题,格基约化算法是求解 SVP 问题的一个有效算法,该算法可以找到格中的一些短向量.YOSHINORI 等人在 2016 年欧洲密码年会上提出 Progressive BKZ 算法,是目前格基约化算法中最为高效的算法之一.详细介绍了 PBKZ 算法,分析了它的运行机理及其内在特点,然后在 Linux 系统下成功调试了 PBKZ 算法库,并针对 Darmstadt 格挑战展开了一系列实验,最终在 600 维和 725 维两种情形下取得了目前国际上最好结果.

关键词:格;格基约化算法;SVP 问题;PBKZ 算法;Darmstadt 格挑战

中图分类号:TP309.7 **文献标志码**:A

随着量子计算算法与硬件实现的快速发展,能够抵抗量子计算攻击且支持数据隐私保护和安全处理的密码算法是目前的迫切需求.基于格的密码体制被普遍认为能够抵抗量子计算攻击,所以其研究近十年来发展迅速.从格算法应用于分析非格公钥密码体制如背包密码体制、RSA 体制的安全性,发展到基于格计算困难问题的密码体制设计,再到基于格计算困难问题设计全同态加密体制.由此可见,格计算问题的算法研究在密码学尤其是后量子密码学中是至关重要的[1].

针对格中 SVP 问题的求解,第一个格基约化算法是由文献[2]在 1982 年提出的 LLL 算法.该算法可以在多项式时间内,输出近似因子为 $((1+\varepsilon)\sqrt{4/3})^{(n-1)/2}$ 的短向量,其中 ε 为一非负常数,n 为所求格的维数.LLL 算法也是至今唯一可以证明是多项式时间运行的格基约化算法.LLL 算法的提出对格理论的研究,特别是公钥密码算法分析起到了很大的推动作用,另外 LLL 算法在计算代数和计算数论等领域也有广泛的应用,已被公认为是 20 世纪最重要的算法之一.1987 年文献[3]利用分块约化的方法推广了 LLL 算法,算法需要多项式次调用求解 k 维子格 SVP 问题的算法,近似因子降为 $O((6k^2)^{n/k})$,使得近似因子和时间复杂度实现了部分的平衡.随后文献[4]在 1994 年提出了 BKZ 算法.BKZ 算法在求解格困难问题中具有重要地位,其后格基约化算法的发展大都基于 BKZ 算法进行改进.BKZ 算法的运行效率与算法设置的分块大小有关,分块越大,得到的短向量模长越小,但运行时间也就越长.在开始的 BKZ 算法中,分块大小一般不大于 30,超过 30 之后算法的时耗过长.2011 年,文献[5]提出 BKZ 2.0 算法,在实际中大大提高了格基约化算法的效率,并且给出了精确的模拟算法,可以估计高维情况下算法输出格基的性质和运行时间.BKZ 2.0 算法最大的改进在于使用了极致剪枝技术[6],极大地提高了枚举算法的运行效率,减少了实际中算法的时耗.BKZ 2.0 算法被成功地应用于 Darmstadt 格挑战中的 q 元格挑战中,目前该挑战大部分纪录由 CHEN 和 NGUYEN 通过 BKZ 2.0 算法得到,这也反映了 BKZ 2.0 算法的优越性.2016 年文献[7]在欧密会上提出了 Progressive BKZ (PBKZ)算法.PBKZ 算法仍是基于 BKZ 算法进行改进得到的,其最大的特点在于分块大小并非固定的,而是先从较小的值开始逐步增加.BKZ 2.0 算法和 PBKZ 算法是目前实际运行效率最高的格基约化算法,经过

收稿日期:2020-03-19;**修回日期**:2020-06-08.

基金项目:国家自然科学基金重点项目(11531002)

作者简介(通信作者):屈龙江(1980—),男,河南信阳人,国防科技大学教授,博士生导师,国家优青,主要从事编码密码理论及其应用研究,E-mail:ljqu_happy@hotmail.com.

改进之后它们的分块大小可以设置在 100 以上,极大地提升了 BKZ 算法的效率,在求解格困难问题中具有重要应用.

Darmstadt 格挑战是由文献[8]在 2008 年发起的一项国际公开挑战,其目的是为了测试各种格基约化算法求解格困难问题的能力.Darmstadt 格挑战通过特殊机制生成的困难问题实例在理论上既可以保证生成实例的随机性,同时也保证实例的困难性.Darmstadt 格挑战发起后就受到国内外密码学者的广泛关注,目前相对于国外来说国内团队取得的结果较少.格挑战的更多内容将在第 3 节进行介绍.

1 基础知识

1.1 格的基本概念

定义 1 格 L 是 n 维欧氏空间 $\mathbf{R}^n$ 中一组线性无关向量 $b_1,b_2,\cdots,b_m$ 的全体整系数线性组合构成的集合,即 $L=\{x_1b_1+x_2b_2+\cdots+x_mb_m \mid x_1,x_2,\cdots,x_m\in\mathbf{Z}\}$.

向量组 $b_1,b_2,\cdots,b_m$ 称为格 L 的一组基.特别地,当 $m=n$ 时,称格 L 为满秩格,通常只考虑满秩格(以下不加特殊说明格 L 均为满秩格).

定义 2 矩阵 $\boldsymbol{B}=[b_1b_2\cdots b_n]$ 称为格 L 的格基矩阵,即 $\boldsymbol{B}$ 的列向量为格 L 的一组基.

格 L 的基不是唯一的,事实上格基矩阵 $\boldsymbol{B}$ 乘以 $\mathbf{Z}^{n\times n}$ 上的任意幺模矩阵 $\boldsymbol{U}$(即 $|\boldsymbol{U}|=\pm1$)得到的矩阵均为格 L 的格基矩阵,其列向量组均为格 L 的一组基.格基约化算法的实质就是通过对初始格基矩阵 $\boldsymbol{B}$ 不断乘以特殊的幺模矩阵,使得 $\boldsymbol{B}$ 的第一个列向量模长越来越小,从而在保证矩阵 $\boldsymbol{B}$ 经过约化算法处理之后仍为格 L 的格基矩阵的基础上得到格 L 中的短向量.

定义 3 格基矩阵 $\boldsymbol{B}$ 的行列式绝对值称为格 L 的行列式,也称为格 L 的体积,记为 $\det(L)$.

格 L 的体积 $\det(L)$ 是格的重要不变量之一,其取值与格基矩阵 $\boldsymbol{B}$ 无关,在求解格困难问题中具有重要应用.

Gram-Schmidt 正交化:给定一组格基 $b_1,b_2,\cdots,b_n$,经过如下处理可以得到一组相互正交的基 $b_1^*,\cdots,b_n^*$(不必为格向量),

$$\begin{cases} b_1^* = b_1 & i=1,\\ b_i^* = b_i-\sum_{j=1}^{i-1}u_{i,j}b_j^* & i=2,3,\cdots,n, \end{cases} \text{其中 } u_{i,j}=\begin{cases} \dfrac{\langle b_i,b_j^*\rangle}{\langle b_j^*,b_j^*\rangle} & i\geq j,\\ 0 & i<j. \end{cases}$$

格 L 上映射 L_i 定义如下:对于格向量 $\boldsymbol{v}$(其在 GS 正交基 $b_1^*,\cdots,b_n^*$ 下表示为 $\boldsymbol{v}=v_1b_1^*+v_2b_2^*+\cdots+v_nb_n^*$,其中 $v_1,\cdots,v_n$ 不必为整数),$L_i(\boldsymbol{v})=v_ib_i^*+\cdots+v_nb_n^*$.由此可知 $L_i(b_i,\cdots,b_j)$ 为格 L 中基为 $b_i,\cdots,b_j$ 的子格的投影映射格,也称为垂直格.

1.2 格上求解困难问题

最短向量问题(Shortest Vector Problem,SVP):给定格 L,找一个非零格向量 $\boldsymbol{v}$,满足对任意非零向量 $\boldsymbol{\mu}\in L$,均有 $\|\boldsymbol{v}\|\leq\|\boldsymbol{\mu}\|$.

最近向量问题(Closest Vector Problem,CVP):给定格 L 和目标向量 $\boldsymbol{t}\in\mathbf{R}^n$,找一个非零格向量 $\boldsymbol{v}$,满足对任意非零向量 $\boldsymbol{u}\in L$,均有 $\|\boldsymbol{v}-\boldsymbol{t}\|\leq\|\boldsymbol{u}-\boldsymbol{t}\|$.

SVP 问题和 CVP 问题是格中 worst-case 下两大基础的困难问题,它们衍生出许多变体问题,这里不再赘述.下面介绍两类 average-case 下的困难问题:

小整数解问题(Small Integer Solutions problem,SIS):给定整数 m,n,q,随机选取矩阵 $\boldsymbol{A}\in\mathbf{Z}_q^{n\times m}$ 和适当的界定参数 β,求解一个非零整数向量 $\boldsymbol{z}\in\mathbf{Z}^m\backslash\{0\}$,使得 $\boldsymbol{Az}=0 \bmod q$ 且 $\|\boldsymbol{z}\|\leq\beta$.

记 $\Lambda_q^{\perp}(\boldsymbol{A})=\{z\in\mathbf{Z}^m \mid \boldsymbol{Az}=0 \bmod q\}$,则集合 $\Lambda_q^{\perp}(\boldsymbol{A})$ 为格,一般称其为 q 元格.求解上述 SIS 问题等价于在 q 元格 $\Lambda_q^{\perp}(\boldsymbol{A})$ 中求模长不大于 β 的短向量.

带错学习问题(Learning With Errors problem,LWE):给定矩阵 $\boldsymbol{A}\in\mathbf{Z}_q^{m\times n}$ 和向量 $\boldsymbol{v}$,$\boldsymbol{v}=\boldsymbol{As}+e$,其中矩阵 $\boldsymbol{A}$ 和秘密向量 $\boldsymbol{s}$ 分别在 $\mathbf{Z}_q^{m\times n}$ 和 $\mathbf{Z}_q^n$ 中均匀随机选取,扰动变量 e 在 $\mathbf{Z}_q^m$ 上服从某种公开分布.判定版本的 LWE 是指区分真正的 LWE 实例中的 v 和在 $\mathbf{Z}_q^m$ 中按均匀分布随机选择的 $\boldsymbol{v}$,搜索版本的 LWE 要求恢复 $\boldsymbol{s}$.

SIS 问题是 1996 年 AJTAI[9] 提出的,AJTAI 在该篇开创性论文中首次提出了格的陷门单向函数思想,创造性地给出了格中困难问题的 worst-case 到 average-case 的归约证明[10],即格问题在最坏情况下的困难性可以归约为一类随机格中问题的困难性,使得基于格的密码体制具有了可证明安全的性质.作为格密码学中两类重要的 average-case 问题之一,SIS 问题广泛地应用于格密码方案中,尤其在格签名方案中具有重要地位.LWE 问题是 REGEV[11] 提出的,他证明了在量子归约下,LWE 问题至少与 worst-case 的近似因子为 $\widetilde{O}(N/\alpha)$ 的 SVP 变体一样困难,其中 α 是 LWE 实例中与扰动分布的方差有关的参数.LWE 问题的提出在基于格的密码学中引起了广泛关注,尤其是在格密码体制的设计中有重要应用.

1.3　格基约化算法

(1)LLL 算法

定义 4(LLL 约化基)　如果一组格基 $b_1,b_2,\cdots,b_n$ 在进行 Gram-Schmidt 正交化过程后满足下面两个条件,则称其为 LLL 约化基.

1)系数规约:$|\mu_{i,j}|<\frac{1}{2},\forall i>j$.

2)Lovasz 条件:$\|b_{i+1}^*+\mu_{i+1,i}b_i^*\|^2\geq\frac{3}{4}\|b_i^*\|^2$.

对于任意一组格基 $b_1,b_2,\cdots,b_n$,通过多项式时间运行 LLL 算法,最终可以得到一组 LLL 约化基.对于格 L 的一组 LLL 约化基 $b_1,b_2,\cdots,b_n$,有 $\|b_1\|\leq 2^{(n-1)/2}\lambda_1(L)$ 成立.虽然 LLL 算法为多项式时间运行,但是其输出的 LLL 约化基性质较差,往往得不到足够短的格向量.

算法 1　LLL 算法

输入　格基 $b_1,b_2,\cdots,b_n$;

输出　LLL 约化基

1 Start:计算 Gram-Schmidt 正交基 $b_1^*,b_2^*,\cdots,b_n^*$;

2　for $i=2$ to n do

3　　for $j=i-1$ to 1 do

4　　$b_i\leftarrow b_i-\mu_{i,j}b_j$

5　　end

6　end

7　if $\exists i$ s.t., $\|b_{i+1}^*+\mu_{i+1,i}b_i^*\|^2<\frac{3}{4}\|b_i^*\|^2$, then

8　$b_i\leftrightarrow b_{i+1}$;

9　goto start;

10　Output $b_1,b_2,\cdots,b_n$;

(2)BKZ 算法

定义 5(BKZ 约化基)　如果一组格基 $b_1,b_2,\cdots,b_n$ 在进行 Gram-Schmidt 正交化过程后满足下面两个条件,则称其为 BKZ 约化基.

1)系数规约:$|\mu_{i,j}|<\frac{1}{2},\forall i>j$.

2)对于 $\forall i,1\leq i<n$,有下式成立:$\|b_i^*\|\leq\lambda_1(L_i(b_i,\cdots,b_{\min(i+\beta-1,n)}))$,其中 β 为 BKZ 算法的分块大小,$2\leq\beta\leq n$.当 $\beta=2$ 时,BKZ 算法即为 LLL 算法.

算法 2　BKZ 算法

输入　格基 $B:b_1,b_2,\cdots,b_n;\beta$

输出　BKZ 约化基

1　$B\leftarrow$LLL(B)

2　$z=0$; $j=0$;

3　While $z<n-1$ do

4　　$j=j+1$;$k=\min(j+\beta-1,n)$;

5　　if $j=n$ then

6　　$j=1$; $k=\beta$;

7　　ENUM($b_j,\cdots,b_k$);

8　　$h=\min(k+1,m)$;

9　　if $L_j(b_h^{\text{new}})\leq L_j(b_j)$

10　　then LLL($b_1,\cdots,b_{j-1},b_j^{new},b_j,\cdots,b_h$);$z=0$;

11　　else LLL($b_1,\cdots,b_h$);$z=z+1$;

12　end while;

13 Output $b_1,b_2,\cdots,b_n$;

枚举算法 ENUM($b_j,\cdots,b_k$) 的作用是在分块($b_j,\cdots,b_k$) 中找到非零格向量 $b_j^{\text{new}}=\sum_{s=j}^{k}u_sb_s$,使得 $L_j(b_j^{\text{new}})=\lambda_1(L_j(b_j,\cdots,b_k))$,其中 $u_s\in\mathbf{Z}$.

2 PBKZ 算法

本节将详细介绍 PBKZ 算法的运行机理及其内在特点,尤其是相对于其他 BKZ 类算法而言,PBKZ 算法具有的独特优势.

2.1 算法主要流程

PBKZ 算法是 YOSHINORI 等人[7]在 2016 年欧密会上提出的,通过逐步增加分块大小来提升 BKZ 算法的运行效率,其与 BKZ 2.0 算法是目前 BKZ 类格基约化算法中两个最高效的改进算法,在求解格困难问题上表现优异.

算法 3 PBKZ 算法

输入 n 维格基矩阵 $\boldsymbol{B}$,分块策略 $(\beta_j^a,\beta_j^{\text{goal}})_{j=1,\cdots,D}$

输出 β_D^{goal} — 约化基 $\boldsymbol{B}$

1 $\boldsymbol{B}\leftarrow \text{LLL}(\boldsymbol{B})$;
2 for $j=1$ to D do
3 While $FEC(\boldsymbol{B}) > Sim-FEC(n,\beta_j^{\text{goal}})$ do
4 for $i=1$ to $n-1$
5 在分块大小为 β 的分块 B_i 中计算 Gram-Schmidt 变量 $\|b_i^*\|,u_{i,j}$;其中 $\beta=\min(\beta_j^a,n-i+1)$;
6 设置对分块 B_i 执行枚举算法的最佳参数;
7 执行枚举算法 ENUM(B_i),输出 $v_1,\cdots,v_h$;
8 更新基向量:$b_i,\cdots,b_{i-1},v_{i_1},\cdots,v_{i_g},b_i,\cdots,b_{i+\beta-1}$;
9 运行 LLL 算法处理上述基向量:LLL$(b_1,\cdots,b_{i-1},v_{i_1},\cdots,v_{i_g},b_i,\cdots,b_{i+\beta-1})$;
10 end for
11 end While
12 end for

步骤 3 中函数 $FEC(\boldsymbol{B})$ 表示对格基 $\boldsymbol{B}$ 执行完全枚举算法所需要的耗时估计,Sim − FEC$(n,\beta_j^{\text{goal}})$ 表示对一个 n 维 β_j^{goal}- 约化基执行完全枚举算法的最小耗时估计,通过比较运行完全枚举算法的耗时来判断格基 $\boldsymbol{B}$ 是否满足进入下一分块进程的条件.

PBKZ 算法最大的改进就在于使用分块策略 $(\beta_j^a,\beta_j^{\text{goal}})_{j=1,\cdots,D}$ 来"智能"地逐步增加分块大小,最终输出 β_D^{goal}- 约化基 $\boldsymbol{B}$.这里首先解释一下分块策略$(\beta_j^a,\beta_j^{\text{goal}})_{j=1,\cdots,D}$:

$$\text{LLL}\xrightarrow{\beta_1^a}\beta_1^{\text{goal}}\xrightarrow{\beta_2^a}\beta_2^{\text{goal}}\rightarrow\cdots\xrightarrow{\beta_D^a}\beta_D^{\text{goal}}.$$

上述分块策略意为先用 LLL 算法处理初始格基 $\boldsymbol{B}$,再多次运行分块大小为 β_1^a 的 BKZ 算法继续处理格基 $\boldsymbol{B}$,直到格基 $\boldsymbol{B}$ 为 β_1^{goal}- 约化基;然后多次运行分块大小为 β_2^a 的 BKZ 算法处理 β_1^{goal}- 约化基 $\boldsymbol{B}$,直至 $\boldsymbol{B}$ 为 β_2^{goal}- 约化的;依次递推下去直到格基 $\boldsymbol{B}$ 为 β_D^{goal}- 约化基,然后输出格基 $\boldsymbol{B}$.

事实上,在 PBKZ 算法之前有许多学者提出过多种分块策略,如 GAMA[6,12],SCHNORR[13]等.他们提出的分块策略大都带有经验性的猜测,缺乏理论依据,而 PBKZ 算法中一个重要的结果就是先对算法进行时间估计,然后将如何选取最优分块策略问题归纳到解决使运行时间最小的最优化问题上面.

许多格中困难问题的求解最终都归结于如何在给定格中求解足够短的格向量,PBKZ 算法相对于其他 BKZ 算法的另一个优势在于:除了设置分块大小参数之外,PBKZ 算法还可以设置模长阈值.如果某一分块进程 $(\beta_j^a,\beta_j^{\text{goal}})$ 结束后得到的 β_j^{goal}- 约化基 $\boldsymbol{B}$ 中模长最短的首列向量的模长小于设置的模长阈值,则此时结束程序并输出该列向量.这样便得到了模长小于设置阈值的短格向量,并且提前终止算法,减少了运行时间.由于 BKZ 类的格基约化算法是概率算法,具有很强的随机性,因此在运行 PBKZ 算法的过程中往往不需要求得最终的 β_D^{goal}-约化基就可以得到需要的足够短的格向量,一定程度上提高了 BKZ 算法运行的效率.

2.2 主要技术改进

2.2.1 分块策略的最优选取

PBKZ 算法与之前算法最明显的区别就在于先对算法的运行时间进行估计,得到算法时耗关于分块策略的函数,将求解最优分块策略问题归于如何选取分块策略使得时耗取最小值的问题上.步骤 7 中的枚举算法是 PBKZ 算法的主要耗时部分,因此耗时估计中忽略步骤 1 中 LLL 算法和步骤 5 中 GS 正交化的时间消

耗,主要对枚举算法的耗时进行估计.

PBKZ 算法的总耗时可以表示如下[7]:

$$\sum_{i=1}^{D} T_{\mathrm{BKZ}}(n,\beta_{i-1}^{\mathrm{goal}} \rightarrow \beta_i^{\mathrm{goal}}),$$

其中 $T_{\mathrm{BKZ}}(n,\beta_{i-1}^{\mathrm{goal}} \rightarrow \beta_i^{\mathrm{goal}})$ 表示分块进程$(\beta_{i-1}^{\mathrm{goal}},\beta_i^{\mathrm{goal}})$ 的耗时.通过曲线拟合技术和数值实验,算法耗时近似地表示如下[7].

当 $n<400$ 时,耗时为: $\sum_{\beta^{\mathrm{start}}}^{\beta^{\mathrm{goal}}} \sum_{t=1}^{\#\mathrm{tours}} [A_1 \cdot \beta^2 n^3 + W_1 \cdot \beta \cdot \sum_{i=1}^{n-1} \mathrm{Enumcost}(B_i)]$,其中 $A_1 = 1.5 \times 10^{-10}$,$W_1 = 1.5 \times 10^{-8}$.

当 $n \geq 400$ 时,耗时为: $\sum_{\beta^{\mathrm{start}}}^{\beta^{\mathrm{goal}}} \sum_{t=1}^{\#\mathrm{tours}} [A_2 \cdot \frac{n-\beta}{H-\beta} \cdot Hn^2 + W_2 \cdot \beta \cdot \sum_{i=1}^{n-1} \mathrm{Enumcost}(B_i)]$,其中 $A_2 = 10^{-6}$,$W_2 = 3.0 \times 10^{-8}$,$H = 250$.

之所以算法的耗时估计在 n 取值 400 前后不同是因为 $n \geq 400$ 时算法需要精确度更高的数据类型.有了上述的算法耗时估计,选取最优的分块策略就转化为找到使算法耗时最小的分块策略问题.

2.2.2 枚举算法中参数的最佳选取

运行枚举算法需要设置参数 α,p.α 表示枚举算法的搜索半径为 $\alpha \cdot GH(B_i)$,其中 $GH(B_i)$ 表示格 B_i 的 Gaussian Heuristic,是对 $\lambda_1(B_i)$ 的有效估计;p 表示枚举算法的成功概率.PBKZ 算法选取满足如下条件的 α,p.在搜索半径 $\alpha \cdot GH(B_i)$ 内约有 $\frac{\alpha^{\beta}}{2}$ 对关于原点对称的格点,其中 β 为分块大小,则枚举算法至少找到一个格点 $\|v\| \leq \alpha \cdot GH(B_i)$ 的概率为 $1-(1-p)^{\frac{\alpha^{\beta}}{2}} \approx p \cdot \frac{\alpha^{\beta}}{2}$,当 $p = \frac{2}{\alpha^{\beta}}$ 时,枚举算法至少找到一个格点 $\|v\| \leq \alpha \cdot GH(B_i)$ 的概率约为 1.因此,PBKZ 算法中参数 α,p 满足条件 $p = \frac{2}{\alpha^{\beta}}$,此时枚举算法以近似 1 的概率输出搜索半径内的短向量.

2.2.3 终止策略

在算法 3 中从步骤 2 即开始按照输入的分块策略对格基 $\boldsymbol{B}$ 运行 BKZ 算法进行处理,在每个 $(\beta_j^a,\beta_j^{\mathrm{goal}})$ 进程中多次运行分块大小为 β_j^a 的 BKZ 算法直到格基 $\boldsymbol{B}$ 为 β_j^{goal}- 约化基.那么究竟如何判断格基 $\boldsymbol{B}$ 是否为 β_j^{goal}- 约化基? PBKZ 算法引入函数 FEC($\boldsymbol{B}$) 进行判断,即步骤 3 中格基 $\boldsymbol{B}$ 是否为 β_j^{goal}- 约化基的判断条件:若 $\mathrm{FEC}(\boldsymbol{B}) > \mathrm{Sim}-\mathrm{FEC}(n,\beta_j^{\mathrm{goal}})$ 则说明当前格基 $\boldsymbol{B}$ 性质不够好,还不是 β_j^{goal}- 约化基,继续执行算法进行处理;反之,则说明格基 $\boldsymbol{B}$ 已是 β_j^{goal}- 约化基,执行下一进程$(\beta_j^a,\beta_{j+1}^{\mathrm{goal}})$.

3 算法实现及格挑战实验

本节先介绍了 Darmstadt 格挑战,然后通过对 PBKZ 算法实现的有效控制和处理,并用之来求解 q 元格挑战,最终在 600 维和 725 维情形下取得了目前国际最好结果.

3.1 Darmstadt 格挑战

Darmstadt 格挑战是文献[8]在 2008 年发起的关于求解一些特定格困难问题的公开挑战,其目的是为了测试格困难问题求解算法的效率.Darmstadt 格挑战自发起以来就受到了格研究领域国内外学者的广泛关注.发展到现在,Darmstadt 格挑战分为 4 个部分,即 q 元格挑战、SVP 挑战、LWE 挑战及理想格挑战.上述4 个挑战分别针对格上不同的困难问题:q 元格挑战是测试求解 SIS 问题的能力,SVP 挑战是测试求解 SVP 问题的能力,LWE 挑战是测试求解 LWE 问题的能力,理想格挑战是测试求解理想格中短向量的能力.这里只针对 q 元格挑战进行求解.

q 元格挑战是在不同维数下的随机 q 元格中求解模长小于 q 的尽量短的格向量.挑战维数从 500 维开始,以 25 递增,目前最高在 825 维下得到符合要求的短向量.q 元格挑战中不同维数的格是按照文献[8]的理论生成的,可以保证构造的这类 q 元格是足够困难的,如果能对这类格求出它的短向量,那么对于更低维数的所有格都可以求出其短向量.这是由格的特殊性质[10]——从 worst case 到 average case 的规约保证的.因

此求解 q 元格挑战对测试格基约化算法求解 SIS 问题的能力具有非常重要的意义.

到目前为止,Darmstadt 格挑战官方网站上已经收录了许多求解纪录[14].q 元格挑战的大多数纪录由著名国际密码学家 Nguyen 团队保持,主要使用工具是 BKZ 2.0 算法.SVP 挑战在高维情况下的纪录大多由 Eamonn 团队和 Yamaguchi 团队得到.国内研究团队方面,清华大学王小云院士团队在 2014－2016 年在 52、54、56、58、62、99、105、113、121 维情况下取得一些最好结果,主要工具是 BKZ 算法.LWE 挑战的高维纪录也主要是由 Eamonn 团队在 2018－2019 年运用筛法得到.理想格挑战的纪录分为 SVP 和 Approx-SVP 两个版本,大部分由 Yoshinori 团队和 Deneuville 取得.综上,2019 年 11 月前国内仅王小云院士团队有纪录,国内团队在格挑战上的研究与国际上还有较大的差距,需要加强研究.

3.2 实验及结果

q 元格挑战原有的纪录大都由 Chen 和 Nguyen 运用 BKZ 2.0 算法得到,因此考虑将 PBKZ 算法运用到求解 q 元格挑战中.为了打破已有纪录,运用 PBKZ 算法做了大量的实验.与 BKZ2.0 算法不同,PBKZ 算法的代码是公开的,2016 年 PBKZ 算法提出之后作者公开了它的算法库—Progressive BKZ library[15].由于 Progressive BKZ library 是基于 Linux 系统的,在调试 Progressive BKZ library 的过程中,先在虚拟机上安装 ubuntu19.0 系统,然后在该系统下安装调试 Progressive BKZ library,并成功运行相应程序.分别使用了笔记本、服务器运行了 PBKZ 算法,其中笔记本的处理器数据为 Intel(R)Core(TM)i5-7300HQ CPU@2.50 GHz 2.50 GHz,安装内存(RAM)为 8 GB;服务器的处理器数据为 Intel(R)Xeon(R)CPUE5-2690v3@2.60 GHz 2.60 GHz,安装内存(RAM)为 256 GB.

在实验中,还采取了文献[16]中提出的删列思想,通过在给定格的子格中求解短向量将问题的难度降低.对于一个 m 维的随机 q 元格 $\Lambda_q^{\perp}(A)$ 来说,通过格基约化算法求得的短向量模长近似为:

$$\min\{q, q^{\frac{n}{m}}\delta^m\},$$

其中 δ 为一常数,其大小只与格基约化算法有关,通常在 1.01 左右.若把 $q^{n/m}\delta^m$ 看作关于 m 的函数,可知当 $m=\sqrt{n\lg q/\lg\delta}$ 时式 $q^{n/m}\delta^m$ 取最小值.因此,在删列实验中,在给定的 m 维 q 元格 $\Lambda_q^{\perp}(A)$ 中保留 $\sqrt{n\lg q/\lg\delta}$ 维的子格,并在子格中求解短向量.

通过实验,发现删列处理之后运行 PBKZ 算法求解 q 元格挑战的效率在一定程度上得到提升.仅以 600 维q 元格挑战为例,在其 180 维的子格中求解短向量,表 1 给出了运用 PBKZ 算法在参数设置相同的情况下分别在 600 维 q 元格和其 180 维子格中求解短向量的实验数据.

由表 1 可知,运用相同的 PBKZ 算法求解,在同一分块进程中删列后得到的短向量模长要更短,且算法耗时也更小.这说明删列处理对 PBKZ 算法的求解效率有一定的提高.

表 1 600 维 q 元格挑战删列实验

Tab.1 Delete-column Experiment in 600-dimension q-ary lattice

分块进程:$(\beta^{\text{start}},\beta^{\text{goal}})$	未删列实验		删列实验	
	模长	耗时/s	模长	耗时/s
(43,49)	83.588	104.739	78.460	79.896
(45,51)	81.498	136.895	77.097	125.847
(46,52)	77.168	162.692	74.659	154.814
(55,61)	72.952	381.640	66.196	262.536
(60,66)	68.366	552.048	61.862	436.568
(74,80)	57.887	1 907.140	53.805	1 510.060
(92,98)	50.348	153 794.000	48.579	118 866.000

通过大量实验,总结了关于 PBKZ 算法实际运行中优化控制的几个关键点.

首先,在调用 Progressive BKZ library 中函数 ProgressiveBKZ 求解格困难问题时,参数设置很关键.ProgressiveBKZ 函数对初始格基的后 k 列进行 LLL 处理,尽管库中 k 设置为 180,但经过实验发现对整个初始格基进行 LLL 处理后算法的运行效果最好.而在模长阈值设置时,大量实验结果提示:阈值越小,算法

运行的效率越高.

其次，虽然文献[7]给出了如何选取最佳分块策略的方法，但是由于其实现过于复杂，因此在 Progressive BKZ library 中分块策略仍是人为地进行控制.在本文的实验中，这种设置具有较高的运行效率：分块大小从 70 开始，以 6 递增，到 125 终止.当然，后续如果做更多的实验，可能还能发现其他更好的设置.

最后，需要指出的是，Progressive BKZ library 中 PBKZ 算法具有很强的随机性，即使设置完全相同的参数，算法运行的实际表现也可能差距很大.例如，在 600 维 q 元格挑战求解中，运行参数设置完全相同的 PBKZ 算法求解程序，有一次在 18 万 s 左右就得到了一个模长小于 41 的格向量，但另一次运行了超过 200 万s 也没有得到模长小于 42 的格向量.

通过实验，取得了 Darmstadt 格挑战的两个新纪录.一是针对 600 维 q 元格挑战，在其 180 维的子格中运行 PBKZ 算法求解，最终求得一个模长为 40.02 的短向量，运行时间 189 076 s.二是针对 725 维 q 元格挑战，在其 210 维的子格中进行求解，最终求得一个模长为 79.31 的短向量，运行时间 1 711 350 s.这两个结果均已被 Darmstadt 格挑战官方网站收录[17]，目前都是相应维数下的最高纪录.

4　总结及展望

本文首先简单回顾了格基约化算法的发展历程，然后介绍了格的一些基础知识及格上的困难求解问题.格的线性结构使得格密码体制具有高效性，同时格上许多困难问题是 NP-hard 保证了基于格求解困难问题设计的密码体制能够抗量子攻击，这些都使得格在后量子时代的地位越来越重要，成了国际密码学研究的前沿和热点.接下来介绍了格基约化算法的相关知识，重点总结了 PBKZ 算法的运行机理及其采用的特殊技术.运用 PBKZ 算法，成功地在 q 元格挑战的 600 维和 725 维实例下取得了目前国际的最优结果[17].据笔者从 Darmstadt 格挑战网站纪录查阅结果来看，这是继清华大学王小云院士团队之后第二个有刷新纪录结果的国内研究团队.目前我们的关于其他维实例的求解程序仍在继续运行，有些维数的输出结果与已有纪录已经比较接近.

格基约化算法作为最有效的格攻击手段之一在破解格密码体制、评估密码方案如格签名算法、格加密算法等的安全性上发挥重要作用，但是由于其自身的复杂性以及研究过程中需要许多经验性的假设，研究格基约化算法的密码学者远没有研究基于格设计的密码方案的学者多，然而这并不能否认格求解算法在格密码中的核心地位.

PBKZ 算法在 2016 年提出，提出之后关于 PBKZ 算法的实际运行效率及其应用于求解格困难问题的研究较少，我们认为对 Progressive BKZ 的进一步认识和改进是值得研究的.对于未来的研究方向，我们认为主要有以下几点.

(1)深入研究 PBKZ 算法的性质和性能提升，一是研究算法的并行实现，二是深入理解并利用随机性；

(2)将 PBKZ 算法中调用的枚举算法替换成筛法.筛法在渐进意义下具有比枚举算法更小的算法复杂度，但是由于在实际中运行效率过低因此很少使用.经过不断的优化，筛法的运行效率已经有了很大的提升，可以考虑将筛法运用到求解 SVP 问题上；

(3)将 PBKZ 算法应用于 SVP 挑战、LWE 挑战、理想格挑战的求解中.目前我们使用 PBKZ 算法求解 q 元格挑战，后面可以使用 PBKZ 算法求解其他三个挑战或者在 PBKZ 算法的基础上设计求解其他挑战的有效算法.

参　考　文　献

[1]　王小云，刘明洁.格密码学研究[J].密码学报，2014，1(1)：22-36.
WANG X Y，LIU M J.Survey of lattice-based cryptography[J].Journal of Cryptologic Research，2014，1(1)：13-27.

[2]　LENSTRA A K，LENSTRA H W，LOVASZ L.Factoring polynomials with rational coefficients[J].Math Ann，1982，261：515-534.

[3]　SCHNORR C P.A hierarchy of polynomial time lattice basis reduction algorithms[J].Theoretical Computer Science，1987，53(2)：201-224.

[4]　SCHNORR C P，EUCHNER M.Lattice basis reduction：improved practical algorithms and solving subset sum problems[J].Mathematical

Programming,1994,66(1/2/3):181-199.
[5] CHEN Y,NGUYEN P Q.BKZ 2.0:better lattice security estimates[C]//Advances in Cryptology-ASIACRYPT 2011.Berlin:Springer,2011:1-20.
[6] GAMA N,NGUYEN P Q,REGEV O.Lattice enumeration using extreme pruning[C]//Advances in Cryptology-EUROCRYPT 2010.Berlin:Springer,2010:257-278.
[7] AONO Y,WANG Y,HAYASHI T,et al.Improved progressive BKZ algorithms and their precise cost estimation by sharp simulator[C]//Advances in Cryptology—EUROCRYPT 2016.Berlin:Springer,2016:789-819.
[8] BUCHMANN J,LINDNER R,RUCKERT M.Explicit Hard Instances of the Shortest Vector Problem[M].Berlin:Springer,2008:79-94.
[9] AJTAI M.Generating hard instances of lattice problems[C]// Proceedings of the 28th Annual ACM Symposium on Theory of Computing.New York:ACM,1996:99-108.
[10] AJTAI M,DWORK C.A public-key cryptosystem with worst-case/average-case equivalence[C]//Proceedings of the 29th Annual ACM Symposium on Theory of Computing.New York:ACM,1997:284-293.
[11] REGEV O.On lattices,learning with errors,random linear codes,and cryptography[J].Journal of the ACM(JACM),2009,56(6):34.
[12] GAMA N,NGUYEN P Q.Predicting lattice reduction[C]//Advances in Cryptology-EUROCRYPT 2008.Berlin:Springer,2008:31-51.
[13] SCHNORR C P,SHEVCHENKO T."Solving subset sum problems of density close to 1 by "randomized" BKZ-reduction"[EB/OL].[2020-03-01].http://eprint.iacr.org/2012/620.pdf.
[14] TU Darmstadt Lattice Challenge[EB/OL].[2020-03-10].http://www.latticechallenge.org/.
[15] PBKZ Library[EB/OL].[2020-03-10].https://www2.nict.go.jp/security/pbkzcode/.
[16] MICCIANCIO D,REGEV O.Lattice-Based Cryptography[C]//Advances in Cryptology-CRYPTO 2006.Berlin:Springer,2006.
[17] HALL OF FAME[EB/OL].[2020-03-10].http://www.latticechallenge.org/halloffame.php.

PBKZ algorithm and its application in lattice challenge

Sun Minghao,Qu Longjiang,Li Chao

(Department of Mathematics,National University of Defense Technology,Changsha 410073,China)

Abstract:Lattice is a discrete addition subgroup in Euclidean space.Many computational problems in lattices have been proved to be NP-hard,so they can be used as the underlying difficult problems of cryptosystems.At the same time,quantum algorithms based on quantum computer models cannot efficiently solve difficult problems in lattices.Therefore,lattice-based cryptography has received more and more attentions in the post-quantum cryptography.The Shortest Vector Problem(SVP)is one of computational problems in lattices,and the lattice reduction algorithm is one of the effective algorithms for solving SVP problems.YOSHINORI et. al proposed the Progressive BKZ algorithm at Eurocrypt 2016,which is one of the most efficient lattice reduction algorithms.In this paper we first introduce the PBKZ algorithm detailedly,and then its operating mechanism and its inherent characteristics.Then we successfully debug the PBKZ algorithm library under Linux system.A series of experiments were carried out in response to the Darmstadt lattice challenge.Finally,the best known results were obtained in 600-dimension and 725-dimension respectively.

Keywords:lattice;lattice reduction algorithm;SVP problem;Progressive BKZ;Darmstadt lattice challenge

[责任编校 陈留院 赵晓华]